ZPL II, ZBI 2, Set-Get-Do, Mirror, WML

Programming Guide

ZEBRA

P1134473-06EN Rev A

Copyright

2025/01/31

ZEBRA and the stylized Zebra head are trademarks of Zebra Technologies Corporation, registered in many jurisdictions worldwide. All other trademarks are the property of their respective owners. ©2024 Zebra Technologies Corporation and/or its affiliates. All rights reserved.

Information in this document is subject to change without notice. The software described in this document is furnished under a license agreement or nondisclosure agreement. The software may be used or copied only in accordance with the terms of those agreements.

For further information regarding legal and proprietary statements, please go to:

SOFTWARE: zebra.com/informationpolicy.
COPYRIGHTS: zebra.com/copyright.
PATENTS: ip.zebra.com.
WARRANTY: zebra.com/warranty.
END USER LICENSE AGREEMENT: zebra.com/eula.

Terms of Use

Proprietary Statement

This manual contains proprietary information of Zebra Technologies Corporation and its subsidiaries (“Zebra Technologies”). It is intended solely for the information and use of parties operating and maintaining the equipment described herein. Such proprietary information may not be used, reproduced, or disclosed to any other parties for any other purpose without the express, written permission of Zebra Technologies.

Product Improvements

Continuous improvement of products is a policy of Zebra Technologies. All specifications and designs are subject to change without notice.

Liability Disclaimer

Zebra Technologies takes steps to ensure that its published Engineering specifications and manuals are correct; however, errors do occur. Zebra Technologies reserves the right to correct any such errors and disclaims liability resulting therefrom.

Limitation of Liability

In no event shall Zebra Technologies or anyone else involved in the creation, production, or delivery of the accompanying product (including hardware and software) be liable for any damages whatsoever (including, without limitation, consequential damages including loss of business profits, business interruption, or loss of business information) arising out of the use of, the results of use of, or inability to use such product, even if Zebra Technologies has been advised of the possibility of such damages. Some jurisdictions do not allow the exclusion or limitation of incidental or consequential damages, so the above limitation or exclusion may not apply to you.

Contents

Introduction **47**
Firmware 47
Who Should Use This Document 47

ZPL Commands **48**
How Commands Are Documented 48
Basic ZPL Exercises and Examples 49
Exercise 1: Specify a Location for an Entered Name 50
Exercise 2: Boxes and Lines 53
Exercise 3: Bar Codes — ^B3 Code 39 Barcode 54
Exercise 4: ^SN — Serial Number Command 55
Exercise 5: Saving a template to memory. ^IS and image save and image load 56
Exercise 6: ^DF and ^XF — Download Format and Recall Format 57
Exercise 7: Asian and Unicode Encodings 58
Allowed Characters in File Names 58
^A 60
^A@ 62
^B0 64
^B1 66
^B2 68
^B3 70
^B4 74
^B5 78
^B7 79
^B8 83

Contents

^B9....85
^BA....87
^BB....90
^BC....94
^BD....106
^BE....109
^BF....111
^BI....114
^BJ....116
^BK....118
^BL....120
^BM....122
^BO....124
^BP....126
^BQ....128
^BR....135
^BS....137
^BT....140
^BU....142
^BX....144
^BY....148
^BZ....150
^CC ~CC....152
^CD ~CD....153
^CF....154
^CI....155
^CM....160
^CN....162
^CO....163
^CP....165
^CT ~CT....166
^CV....167
^CW....168
~DB....170

~DE....172
^DF....174
~DG....175
~DN....177
~DS....178
~DT....179
~DU....180
~DY....181
~EG....185
^FB....186
^FC....189
^FD....190
^FE....191
^FH....193
^FL....195
^FM....197
^FN....200
^FO....201
^FP....202
^FR....203
^FS....204
^FT....205
^FV....207
^FW....208
^FX....209
^GB....210
^GC....212
^GD....213
^GE....214
^GF....215
^GS....217
~HB....218
~HD....219
^HF....220

^HG..... 221
^HH..... 222
~HI..... 224
~HM..... 225
~HQ..... 226
~HS..... 233
^HT..... 237
~HU..... 238
^HV..... 239
^HW..... 240
^HY..... 242
^HZ..... 243
^ID..... 245
^IL..... 247
^IM..... 248
^IS..... 249
~JA..... 251
^JB..... 252
~JB..... 253
~JC..... 254
~JD..... 255
~JE..... 256
~JF..... 257
~JG..... 258
^JH..... 259
^JI..... 263
~JI..... 265
^JJ..... 266
~JL..... 268
^JM..... 269
~JN..... 270
~JO..... 271
~JP..... 272
~JQ..... 273

~JR....274
^JS....275
~JS....276
^JT....277
^JU....278
^JW....279
~JX....280
^JZ....281
~KB....282
^KD....283
^KL....284
^KN....285
^KP....287
^KV....288
^LF....292
^LH....293
^LL....294
^LR....295
^LS....296
^LT....297
^MA....298
^MC....300
^MD....301
^MF....302
^MI....303
^ML....304
^MM....305
^MN....307
^MP....308
^MT....310
^MU....311
^MW....313
^PA....314
^PF....315

^PH ~PH 316
~PL 317
^PM 318
~PM 319
^PN 320
^PO 321
^PP ~PP 322
^PQ 323
~PR 324
^PR 325
~PS 327
^PW 328
~RO 329
^SC 331
~SD 332
^SE 333
^SF 334
^SI 337
^SL 338
^SN 340
^SO 342
^SP 343
^SQ 345
^SR 347
^SS 348
^ST 350
^SX 351
^SZ 353
~TA 354
^TB 355
^TO 356
~WC 358
^WD 360
~WQ 362

^XA....369
^XB....370
^XF....371
^XG....372
^XS....373
^XZ....374
^ZZ....375

ZPL Network Commands....376
^KC....377
^NB....378
^NC....379
~NC....380
^ND....381
^NI....383
^NN....384
^NP....385
~NR....386
^NS....387
^NT....389
~NT....390
^NW....391
^WA....392
^WE....393
^WL - Set Leap....395
~WL - Print Network....396
^WP....398
^WR - Set Transmit....399
~WR - Reset Wireless....400
^WS....401
^WX....403

ZPL RFID Commands....411

^HL or ~HL....412
^HR....414
^RB....423
^RF....425
^RL....429
^RS....432
^RU....437
^RW....439

ZBI Commands....443
Introduction to Zebra Basic Interpreter (ZBI)....443
Printers, ZBI Keys, & ZBI Versions....443
Command and Function Reference Format....444
Function Rules....444
Command/Function NAME....444
Section Organization....445
Writing ZBI Programs....446
Editing Commands....446
NEW....448
REM....449
! (EXCLAMATION MARK)....450
LIST....451
AUTONUM....452
RENUM....453
ECHO....454
Running and Debugging Commands....455
RUN....457
CTRL-C....458
RESTART....459
STEP....460
DEBUG....461
TRACE....462
BREAK....463
ADDBREAK....464

DELBREAK 465
ZPL 466
Base Types and Expressions 467
Variable Names 467
Variable Declarations 468
Constants 468
Arrays 468
Assignment 469
LET 470
Numeric Expressions 470
String Concatenation (&) 471
Sub-strings 472
Boolean Expressions 473
Combined Boolean Expressions 475
Control and Flow 476
IF Statements 476
DO Loops 477
FOR Loops 478
GOTO/GOSUB 480
SUB 481
EXIT 482
END 483
Input and Output 484
OPEN 486
CLOSE 487
DATAREADY 488
SERVERSOCKET 489
SERVERCLOSE 490
CLIENTSOCKET 491
ACCEPT 492
Reading and Writing 493
INPUT 494
PRINT 496
OUTBYTE 497

INBYTE....498
READ....499
WRITE....500
SEARCHTO$....501
Port Usage Examples....503
Physical Ports (Serial, Parallel, USB, Bluetooth®)....503
ZPL Parser....503
TCP Client....504
TCP Server....504
UDP Client....505
UDP Server....505
E-mail....506
File System....508
Runtime Access....508
STORE....510
LOAD....511
DIR....512
DELETE....513
Comma Separated Values (CSV)....514
CSVLOAD....515
CSV File Information....516
CSVSTORE....517
TXTLOAD....518
TXTSTORE....519
Events....520
ZBI Key Names....522
REGISTEREVENT....525
UNREGISTEREVENT....527
HANDLEEVENT....528
TRIGGEREVENT....530
Systems....530
ISERROR....531
ISWARNING....532
SLEEP....533

SETERR..... 534
CLRERR..... 535
ON ERROR..... 536
Applicator Functions..... 537
AUXPORT_STEALPIN..... 538
AUXPORT_SETPIN..... 539
AUXPORT_GETPIN..... 540
AUXPORT_RELEASEPIN..... 541
String Functions..... 542
LCASE$..... 543
CHR$..... 544
LTRIM$..... 545
REPEAT$..... 546
RTRIM$..... 547
SPLIT..... 548
SPLITCOUNT..... 552
UCASE$..... 553
EXTRACT$..... 554
ORD..... 556
POS..... 557
LEN..... 558
Math Functions..... 559
STR$..... 560
MAX..... 561
MIN..... 562
MAXNUM..... 563
MOD..... 564
VAL..... 565
INTTOHEX$..... 566
HEXTOINT..... 568
Array Functions..... 570
REDIM..... 571
INSERTROW..... 572
DELROW..... 573

ROWSIZE....574
COLUMNSIZE....575
FIND....576
Time and Date Functions....578
DATE$....579
TIME$....580
DATE....581
TIME....582
Set/Get/Do Interactions....583
SETVAR....584
GETVAR$....585
Example Programs....586
Array Program....586
CSV Program....587
DPI Conversion Program....590
Email Program....593
Extraction 1 Program....594
Extraction 2 Program....595
Front Panel Control....597
Recall Program....599
Scale Program....600

About SGD Printer Commands....602
Overview....603
setvar Command....603
getvar Command....603
do Command....603
Command Structure....604
How to Send Multiple SGD Commands....604
JSON (JavaScript Object Notation)....604
Configuring JSON Usage for Communications....605
Get an SGD Branch....606
Get an allvalues Report....606
Get an allconfig Report....607

SGD Command Reference **608**
alerts.add 609
alerts.conditions 611
alerts.configured 612
alerts.destinations 613
alerts.http.authentication.add 614
alerts.http.authentication.entries 616
alerts.http.authentication.remove 617
alerts.http.logging.clear 618
alerts.http.logging.entries 619
alerts.http.logging.max_entries 620
alerts.http.proxy 622
alerts.send_current_status_alerts 624
alerts.tracked_settings.clear_log 625
alerts.tracked_settings.log_tracked 626
alerts.tracked_settings.max_log_entries 627
alerts.tracked_sgds.log 628
alerts.tracked_sgds.max_log_entries 629
alerts.tracked_sgds.zbi_notified 630
apl.enable 631
apl.framework_version 632
apl.settings 633
apl.version 635
appl.bootblock 636
appl.date 637
appl.link_os_version 638
appl.link_os_version_full 639
appl.name 640
appl.option_board_version 641
capture.channel1.count 642
capture.channel1.data.mime 643
capture.channel1.data.raw 644
capture.channel1.delimiter 645
capture.channel1.max_length 646

capture.channel1.port....647
CISDFCRC16 Download Files....648
comm.baud....650
comm.halt....651
comm.mode....652
comm.pnp_option....653
comm.type....654
comm.parity....655
comm.stop_bits....656
cradle.comm.baud....657
cradle.comm.handshake....658
cutter.clean_cutter....659
cutter.clean_reminder_enable....660
cutter.clean_reminder_threshold....661
device.allow_firmware_downloads....662
device.applicator.data_ready....663
device.applicator.end_print....664
device.applicator.feed....665
device.applicator.media_out....666
device.applicator.pause....667
device.applicator.reprint....668
device.applicator.rfid_void....669
device.applicator.ribbon_low....670
device.applicator.ribbon_out....671
device.applicator.service_required....672
device.applicator.start_print....673
device.bluetooth_installed....674
device.command_override.active....675
device.command_override.add....676
device.command_override.clear....677
device.command_override.list....678
device.company_contact....679
device.configuration_number....680
device.cpcl_synchronous_mode....681

device.cutter_installed..... 682
device.download_connection_timeout..... 683
device.download_interactive_mode..... 684
device.epl_legacy_mode..... 685
device.feature.bluetooth_le..... 688
device.feature.mcr..... 689
device.feature.nfc..... 690
device.feature.ribbon_cartridge..... 691
device.feature.802_11ac..... 692
device.feature.802_11ax..... 693
device.feature.head_element_test..... 694
device.friendly_name..... 695
device.frontpanel.feedenabled..... 696
device.frontpanel.key_press..... 697
device.frontpanel.line1..... 699
device.frontpanel.line2..... 700
device.frontpanel.xml..... 701
device.host_identification..... 702
device.host_status..... 703
device.idle_display_format..... 707
device.idle_display_value..... 708
device.internal_wired_setting_location..... 709
device.jobs_print..... 710
device.job_log.total_jobs_logged..... 711
device.languages..... 712
device.light.cover_open_brightness..... 713
device.light.head_open_brightness..... 714
device.location..... 715
device.loader_version..... 716
device.ltu_installed..... 717
device.mcu_communication.revision..... 718
device.mcu_cutter.revision..... 719
device.mcu_cutter.desired_revision..... 720
device.mcu_io_expand_rev..... 721

device.mcu_io_expand.desired_rev..... 722
device.orientation.....723
device.pause..... 724
device.pnp_option.....725
device.pmcu.revision..... 726
device.position.accuracy.....727
device.position.altitude.....728
device.position.latitude.....729
device.position.longitude.....730
device.print_2key.....731
device.print_reprogram_2key..... 732
device.printhead.test.summary.....733
device.printhead.odometer..... 734
device.printhead.test.detail..... 735
device.product_name_submodel.....736
device.prompted_network_reset.....737
device.prompted_default_network.....738
device.prompted_reset..... 739
device.protected_mode.....740
device.protected_mode_allowed.....741
device.reset..... 742
device.reset_button_enable..... 743
device.restore_defaults.....744
device.rewinder_installed.....745
device.save_2key..... 746
device.sensor_select.....747
device.sensor_profile.....748
device.serial_number.option_board_date.....749
device.serial_numbers.control_panel_date..... 750
device.serial_numbers.mlb_date.....751
device.serial_numbers.processor.....752
device.serial_numbers.applicator_option_board_date.....753
device.serial_numbers.wired_ethernet_option_board..... 754
device.serial_numbers.wired_ethernet_option_board_date..... 755

device.serial_numbers.applicator_option_board..... 756
device.serial_numbers.cutter.....757
device.serial_numbers.cutter_date.....758
device.serial_numbers.printhead..... 759
device.serial_numbers.printhead_date..... 760
device.serial_numbers.usb_host_option_board_date.....761
device.serial_numbers.usb_host_option_board.....762
device.serial_numbers.parallel_option_board.....763
device.serial_numbers.parallel_option_board_date.....764
device.set_clock_to_build_date.....765
device.slot_1..... 766
device.slot_2..... 767
device.super_host_status.....768
device.syslog.clear_log..... 769
device.syslog.configuration..... 770
device.syslog.enable..... 772
device.syslog.entries..... 773
device.syslog.log_max_file_size.....774
device.syslog.save_local_file.....775
device.applicator.data_ready_activation..... 776
device.applicator.error_on_pause..... 777
device.applicator.start_print_mode.....778
device.applicator.voltage..... 779
device.unique_id..... 780
device.unpause.....781
device.uptime..... 782
device.user_p1..... 783
device.user_p2..... 784
device.user_vars.set_range..... 785
device.user_vars.create..... 786
device.xml.enable..... 788
device.feature.lighted_arrows..... 789
device.light.ribbon_path_brightness..... 790
device.light.media_path_brightness.....791

device.zuid....792
display.backlight....793
display.backlight_on_time....794
display.batch_counter....795
display.bluetooth.mac....796
display.calibrate....797
display.language....798
display.load_card....800
display.password.level....801
display.root_wml....802
display.text....803
ezpl.head_close_action....804
ezpl.label_length_max....805
ezpl.label_sensor....806
ezpl.manual_calibration....807
ezpl.media_type....808
ezpl.power_up_action....809
ezpl.print_method....810
ezpl.print_width....811
ezpl.reprint_mode....812
ezpl.take_label....813
ezpl.take_label_calibration....814
ezpl.tear_off....815
file.capture_response.begin....816
file.capture_response.destination....817
file.capture_response.end....818
file.cert.expiration....819
file.cert.supported_curves....820
file.delete....821
file.dir....822
file.dir_format....823
file.type....824
file.run....825
formats.cancel_all....826

head.authenticated....827
head.darkness_switch_enable....828
head.darkness_switch....829
head.element_test....830
head.latch....831
head.resolution.in_dpi....832
input.capture....833
log.reboot.code....835
log.reboot.codes....836
log.reboot.reason....837
log.reboot.report....838
mcr.crypt.enabled....839
mcr.cancel....840
mcr.crypt.key_mgmt....841
mcr.crypt.algorithm....842
mcr.out....843
mcr.revision....844
media.bar_location....845
media.cartridge.darkness....846
media.cartridge.labels_remaining....847
media.cartridge.width....848
media.cartridge.total_label_cnt....849
media.cartridge.speed....850
media.cartridge.length....851
media.cartridge.inserted....852
media.cartridge.part_number....853
media.cut_now....854
media.darkness_mode....855
media.draft_mode....856
media.dynamic_length_calibration....857
media.extended_presentation....858
media.feed_skip....859
media.linerless_offset....860
media.media_low.external....861

media.media_low.warning 862
media.part_number 863
media.present.cut_amount 864
media.present.eject 865
media.present.length_addition 866
media.present.loop_length 867
media.present.loop_length_max 868
media.present.cut_margin 869
media.present.present_timeout 870
media.present.present_type 871
media.printmode 872
media.speed 874
media.serial_number 875
media.tof 876
memory.flash_free 878
memory.flash_size 879
memory.ram_free 880
memory.ram_size 881
mqtt.enable 882
mqtt.logging.clear 883
mqtt.logging.entries 884
mqtt.logging.max_entries 885
mqtt.restore_defaults 886
mqtt.conn[1|2].clean_session_flag 887
mqtt.conn[1|2].password 888
mqtt.conn[1|2].ping_interval 889
mqtt.conn[1|2].reset_now 890
mqtt.conn[1|2].reset_required 891
mqtt.conn[1|2].retry_interval_random_max 892
mqtt.conn[1|2].server_address 893
mqtt.conn[1|2].tenant_id 894
mqtt.conn[1|2].username 895
netmanage.avalanche.agent_addr 896
netmanage.avalanche.available_agent 897

netmanage.avalanche.available_port 898
netmanage.avalanche.encryption_type 899
netmanage.avalanche.interval 900
netmanage.avalanche.interval_update 901
netmanage.avalanche.model_name 902
netmanage.avalanche.set_property 903
netmanage.avalanche.startup_update 904
netmanage.avalanche.tcp_connection_timeout 905
netmanage.avalanche.terminal_id 906
netmanage.avalanche.text_msg.beep 907
netmanage.avalanche.text_msg.display 908
netmanage.avalanche.text_msg.print 909
netmanage.avalanche.udp_timeout 910
netmanage.error_code 911
netmanage.state_code 912
netmanage.status_code 913
odometer.cut_marker_count 914
odometer.headclean 915
odometer.headnew 916
odometer.label_dot_length 917
odometer.media_marker_count 918
odometer.media_marker_count1 919
odometer.media_marker_count2 920
odometer.net_media_length 921
odometer.net_ribbon_length 922
odometer.retracts_count 923
odometer.rfid.valid_resettable 924
odometer.rfid.void_resettable 926
odometer.total_cuts 928
odometer.total_print_length 929
odometer.total_label_count 930
odometer.user_label_count 931
odometer.user_total_cuts 932
odometer.user_label_count[1|2] 933

odometer.latch_open_count 934
parallel_port.mode 935
parallel_port.present 936
power.average_current 937
power.battery_led_blink_rate 938
power.battery_led_enable 939
power.battery_led_off_duration 940
power.battery_led_on_duration 941
power.battery_type 942
power.dtr_power_off 943
power.energy_star.enable 944
power.energy_star.timeout 945
power.label_queue.shutdown 946
power.power_on_mode 947
power.shutdown 948
power.voltage 949
power.wake.radio 950
power.current 951
power.temperature 952
power.percent_health 953
power.part_number 954
power.sleep.cradle 955
power.remaining_capacity 956
power.cycle_count 957
print.legacy_compatibility 958
print.tone 959
print.troubleshooting_label_print 960
ribbon.serial_number 961
ribbon.part_number 962
ribbon.cartridge.part_number 963
ribbon.cartridge.length_remaining 964
ribbon.cartridge.length 965
ribbon.cartridge.authenticated 966
ribbon.cartridge.inserted 967

ribbon.coating....968
ribbon.tension....969
rtc.exists....970
rtc.date....971
rtc.time....972
rtc.timezone....973
rtc.unix_timestamp....975
sensor.air_pressure.current_reading....976
sensor.ambient_light.current_reading....977
sensor.battery.in_volts....978
sensor.back_bar.brightness....979
sensor.back_bar.ppr_out_thold....980
sensor.back_bar.cur....981
sensor.cover_open....982
sensor.front_bar.ppr_out_thold....983
sensor.front_bar.cur....984
sensor.front_bar.thold....985
sensor.front_bar.gain....986
sensor.front_bar.brightness....987
sensor.front_bar.offset....988
sensor.back_bar.offset....989
sensor.gap.thold....990
sensor.gap.offset....991
sensor.gap.gain....992
sensor.gap.brightness....993
sensor.head.temp_avg....994
sensor.head.temp_celsius....995
sensor.head.temp....996
sensor.magnetometer.current_reading....997
sensor.object_temperature.current_reading....998
sensor.peel.thold....999
sensor.peel.gain....1000
sensor.paper_supply....1001
sensor.peeler....1002

sensor.peel.brightness....1003
sensor.proximity.current_reading....1004
sensor.width.in_dots....1005
sensor.width.cur....1006
sensor.self_adjusting_enable....1007
usb.device.device_id_string....1008
usb.device.device_unique_id....1009
usb.device.device_version....1010
usb.device.manufacturer_string....1011
usb.device.product_id....1012
usb.device.product_string....1013
usb.device.serial_string....1014
usb.device.vendor_id....1015
usb.halt....1016
usb.host.config_info_to_usb....1017
usb.host.fn_field_data....1018
usb.host.fn_last_field....1019
usb.host.hid_count....1020
usb.host.keyboard_input....1021
usb.host.lock_out....1022
usb.host.mass_storage_count....1023
usb.host.read_list....1024
usb.host.read_list_print_delay....1025
usb.host.template_list....1026
usb.host.template_print_amount....1027
usb.host.write_list....1028
usb.mirror.appl_path....1029
usb.mirror.auto....1030
usb.mirror.enable....1031
usb.mirror.enabled....1032
usb.mirror.error_retry....1033
usb.mirror.feedback.auto....1034
usb.mirror.feedback.odometer....1035
usb.mirror.feedback.path....1036

usb.mirror.fetch 1037
usb.mirror.last_error 1038
usb.mirror.last_time 1039
usb.mirror.path 1040
usb.mirror.reset_delay 1041
usb.mirror.success 1042
usb.mirror.success_time 1043
zbi.control.add_breakpoint 1044
zbi.control.break 1045
zbi.control.clear_breakpoints 1046
zbi.control.delete_breakpoint 1047
zbi.control.line_number 1048
zbi.control.restart 1049
zbi.control.run 1050
zbi.control.step 1051
zbi.control.terminate 1052
zbi.control.variable_name 1053
zbi.control.variable_value 1054
zbi.enable 1055
zbi.key 1056
zbi.last_error 1057
zbi.program_list 1058
zbi.reseller_key 1060
zbi.revision 1061
zbi.running_program_name 1062
zbi.start_info.execute 1063
zbi.start_info.file_name 1064
zbi.start_info.memory_alloc 1065
zbi.state 1066
zpl.calibrate 1067
zpl.format_prefix 1068
zpl.caret 1069
zpl.control_character 1070
zpl.delimiter 1071

zpl.label_length....1072
zpl.label_length_always....1073
zpl.label_orientation....1074
zpl.left_position....1075
zpl.system_error....1076
zpl.system_status....1078
zpl.zpl_mode....1081
zpl.zpl_override....1082
zpl.relative_darkness....1083

SGD Network Commands....1084
bluetooth.address....1085
bluetooth.afh_map....1086
bluetooth.afh_map_curr....1087
bluetooth.afh_mode....1088
bluetooth.allow_non_display_numeric_comparison....1089
bluetooth.authentication....1090
bluetooth.bluetooth_pin....1091
bluetooth.clear_bonding_cache....1092
bluetooth.date....1093
bluetooth.discoverable....1094
bluetooth.enable....1095
bluetooth.enable_reconnect....1096
bluetooth.friendly_name....1097
bluetooth.json_config_channel_enable....1098
bluetooth.power_class....1099
bluetooth.le.controller_mode....1100
bluetooth.le.power_class....1101
bluetooth.le.minimum_security....1102
bluetooth.page_scan_window....1103
bluetooth.local_name....1104
bluetooth.minimum_security_mode....1105
bluetooth.radio_auto_baud....1107
bluetooth.radio_version....1108

bluetooth.short_address....1109
bluetooth.version....1110
card.mac_addr....1111
card.inserted....1112
external_wired.check....1113
external_wired.ip.addr....1114
external_wired.ip.arp_interval....1115
external_wired.ip.default_addr_enable....1116
external_wired.ip.dhcp.cid_all....1117
external_wired.ip.dhcp.cid_enable....1118
external_wired.ip.dhcp.cid_prefix....1119
external_wired.ip.dhcp.cid_suffix....1120
external_wired.ip.dhcp.cid_type....1121
external_wired.ip.gateway....1122
external_wired.ip.netmask....1123
external_wired.ip.port....1124
external_wired.ip.protocol....1125
external_wired.ip.timeout.enable....1126
external_wired.ip.timeout.value....1127
external_wired.ip.v6.addr....1128
external_wired.ip.v6.gateway....1129
external_wired.ip.v6.prefix_length....1130
external_wired.mac_addr....1131
external_wired.mac_raw....1132
interface.network.active.arp_interval....1133
interface.network.active.cable_type....1134
interface.network.active.dhcp_received_host_name....1135
interface.network.active.gateway....1136
interface.network.active.ip_addr....1137
interface.network.active.ipv6.addresses....1138
interface.network.active.ipv6.address_type....1139
interface.network.active.ipv6.dhcp_server_duid....1140
interface.network.active.ipv6.gateways....1141
interface.network.active.mac_addr....1142

interface.network.active.mac_raw 1143
interface.network.active.netmask 1144
interface.network.active.protocol 1145
interface.network.active.protocol_error 1146
interface.network.active.rx_errors 1147
interface.network.active.rx_packets 1148
interface.network.active.server_address 1149
interface.network.active.speed 1150
interface.network.active.tx_errors 1151
interface.network.active.tx_packets 1152
interface.network.active.wins_addr 1153
interface.network.settings_require_reset 1154
internal_wired.8021x.password 1155
internal_wired.8021x.peap.validate_server_certificate 1156
internal_wired.8021x.peap.anonymous_identity 1157
internal_wired.8021x.private_key_password 1158
internal_wired.8021x.security 1159
internal_wired.8021x.ttls_anonymous_identity 1160
internal_wired.8021x.ttls_tunnel 1161
internal_wired.8021x.username 1162
internal_wired.activity_led 1163
internal_wired.auto_switchover 1164
internal_wired.enable 1165
internal_wired.installed 1166
internal_wired.ip.addr 1167
internal_wired.ip.arp_interval 1168
internal_wired.ip.default_addr_enable 1169
internal_wired.ip.dhcp.arp_verify 1170
internal_wired.ip.dhcp.cache_ip 1171
internal_wired.ip.dhcp.cid_all 1172
internal_wired.ip.dhcp.cid_enable 1173
internal_wired.ip.dhcp.cid_prefix 1174
internal_wired.ip.dhcp.cid_suffix 1175
internal_wired.ip.dhcp.cid_type 1176

internal_wired.ip.dhcp.lease.last_attempt..... 1177
internal_wired.ip.dhcp.lease.length.....1178
internal_wired.ip.dhcp.lease.server.....1179
internal_wired.ip.dhcp.lease.time_left.....1180
internal_wired.ip.dhcp.option12.....1181
internal_wired.ip.dhcp.option12_format.....1182
internal_wired.ip.dhcp.option12_value.....1183
internal_wired.ip.dhcp.requests_per_session.....1184
internal_wired.ip.dns.domain.....1185
internal_wired.ip.dns.domain_user_value.....1186
internal_wired.ip.dns.servers.....1187
internal_wired.ip.dns.servers_user_value.....1188
internal_wired.ip.gateway.....1189
internal_wired.ip.netmask.....1190
internal_wired.ip.port.....1191
internal_wired.ip.port_alternate.....1192
internal_wired.ip.port_json_config.....1193
internal_wired.ip.protocol..... 1194
internal_wired.ip.timeout.enable.....1195
internal_wired.ip.timeout.value.....1196
internal_wired.ip.wins.addr..... 1197
internal_wired.ip.wins.permanent_source.....1198
internal_wired.ipv6.addresses.....1199
internal_wired.ipv6.address_type..... 1200
internal_wired.ipv6.dhcp.lease.last_attempt.....1201
internal_wired.ipv6.dhcp.lease.length.....1202
internal_wired.ipv6.dhcp.lease.time_left..... 1203
internal_wired.ipv6.dhcp.option39_enable..... 1204
internal_wired.ipv6.dhcp.option39_format.....1205
internal_wired.ipv6.dhcp.option39_fqdn.....1206
internal_wired.ipv6.dhcp.option39_value..... 1207
internal_wired.ipv6.gateways..... 1208
internal_wired.ipv6.static.addresses.....1209
internal_wired.ipv6.static.gateways..... 1210

internal_wired.mac_addr....1211
internal_wired.mac_raw....1212
ip.active_network....1213
ip.addr....1214
ip.address_mode....1215
ip.arp_interval....1216
ip.bootp.enable....1217
ip.dhcp.arp_verify....1218
ip.dhcp.auto_provision_enable....1219
ip.dhcp.cache_ip....1220
ip.dhcp.cid_all....1221
ip.dhcp.cid_enable....1222
ip.dhcp.cid_prefix....1223
ip.dhcp.cid_suffix....1224
ip.dhcp.cid_type....1225
ip.dhcp.cid_value....1226
ip.dhcp.dhcpv6_duid....1227
ip.dhcp.enable....1228
ip.dhcp.lease.last_attempt....1229
ip.dhcp.lease.server....1230
ip.dhcp.lease.time_left....1231
ip.dhcp.ntp.enable....1232
ip.dhcp.ntp.received_servers....1233
ip.dhcp.option12....1234
ip.dhcp.option12_format....1235
ip.dhcp.option12_value....1236
ip.dhcp.request_timeout....1237
ip.dhcp.requests_per_session....1238
ip.dhcp.session_interval....1239
ip.dhcp.user_class_id....1240
ip.dhcp.vendor_class_id....1241
ip.dns.domain....1242
ip.dns.servers....1243
ip.firewall.authentication.add....1244

ip.firewall.authentication.entries....1245
ip.firewall.authentication.remove....1246
ip.firewall.proxy....1247
ip.firewall.whitelist_in....1248
ip.ftp.enable....1249
ip.ftp.execute_file....1250
ip.ftp.request_password....1251
ip.gateway....1252
ip.http.admin_name....1253
ip.http.admin_password....1254
ip.http.custom_link_name....1255
ip.http.custom_link_url....1256
ip.http.enable....1257
ip.http.faq_url....1258
ip.http.port....1259
ip.https.enable....1260
ip.https.port....1261
ip.lpd.enable....1262
ip.mac_raw....1263
ip.mirror.appl_path....1264
ip.mirror.auto....1265
ip.mirror.error_retry....1266
ip.mirror.feedback.auto....1267
ip.mirror.feedback.freq....1268
ip.mirror.feedback.odometer....1269
ip.mirror.feedback.path....1270
ip.mirror.fetch....1271
ip.mirror.freq....1272
ip.mirror.freq_hours....1273
ip.mirror.interface....1274
ip.mirror.last_error....1275
ip.mirror.last_time....1276
ip.mirror.mode....1277
ip.mirror.password....1278

ip.mirror.path....1279
ip.mirror.reset_delay....1280
ip.mirror.server....1281
ip.mirror.success....1282
ip.mirror.success_time....1283
ip.mirror.username....1284
ip.mirror.version....1285
ip.netmask....1286
ip.ntp.enable....1287
ip.ntp.log....1288
ip.ntp.servers....1289
ip.ping_gateway_interval....1290
ip.pop3.enable....1291
ip.pop3.password....1292
ip.pop3.poll....1293
ip.pop3.print_body....1294
ip.pop3.print_headers....1295
ip.pop3.save_attachments....1296
ip.pop3.server_addr....1297
ip.pop3.username....1298
ip.pop3.verbose_headers....1299
ip.port....1300
ip.port_alternate....1301
ip.port_json_config....1302
ip.port_single_conn....1303
ip.port_single_conn_idle_timeout....1304
ip.primary_network....1305
ip.smtp.domain....1306
ip.smtp.enable....1307
ip.smtp.server_addr....1308
ip.snmp.enable....1309
ip.snmp.get_community_name....1310
ip.snmp.set_community_name....1311
ip.snmp.trap_community_name....1312

ip.snmpv3.admin.auth_protocol.... 1313
ip.snmpv3.admin.name.... 1314
ip.snmpv3.admin.priv_protocol.... 1315
ip.snmpv3.enable.... 1316
ip.snmpv3.monitor.auth_protocol.... 1317
ip.snmpv3.monitor.name.... 1318
ip.snmpv3.monitor.priv_protocol.... 1319
ip.tcp.enable.... 1320
ip.tcp.nagle_algorithm.... 1321
ip.tls.enable.... 1322
ip.tls.port.... 1323
ip.tls.port_json_config.... 1324
ip.udp.enable.... 1325
weblink.cloud_connect.enable.... 1326
weblink.enable.... 1327
weblink.ip.conn[1|2].authentication.add.... 1328
weblink.ip.conn[1|2].authentication.entries.... 1330
weblink.ip.conn[1|2].authentication.remove.... 1331
weblink.ip.conn[1|2].location.... 1332
weblink.ip.conn[1|2].num_connections.... 1333
weblink.ip.conn[1|2].maximum_simultaneous_connections.... 1334
weblink.ip.conn[1|2].proxy.... 1335
weblink.ip.conn[1|2].retry_interval.... 1337
weblink.ip.conn1.test.location.... 1338
weblink.ip.conn[1|2].test.retry_interval.... 1340
weblink.ip.conn1.retry_interval_random_max.... 1341
weblink.ip.conn1.test.test_on.... 1342
weblink.logging.clear.... 1344
weblink.logging.entries.... 1345
weblink.logging.max_entries.... 1346
weblink.printer_reset_required.... 1348
weblink.restore_defaults.... 1349
weblink.zebra_connector.authentication.add.... 1350
weblink.zebra_connector.authentication.entries.... 1351

weblink.zebra_connector.authentication.remove....1352
weblink.zebra_connector.enable....1353
weblink.zebra_connector.proxy....1354
weblink.zebra_connector.version....1356
wlan.11ac.80mhz_enable....1357
wlan.11d.enable....1358
wlan.11n.20mhz_only....1359
wlan.11n.aggregation....1360
wlan.11n.greenfield....1361
wlan.11n.rifs....1362
wlan.11n.short_gi_40mhz....1363
wlan.11n.short_gi_20mhz....1364
wlan.8021x.enable....1365
wlan.8021x.validate_peap_server_certificate....1366
wlan.8021x.peap.anonymous_identity....1367
wlan.8021x.authentication....1368
wlan.8021x.eap.password....1369
wlan.8021x.eap.username....1370
wlan.8021x.eap.privkey_password....1371
wlan.8021x.peap.peap_password....1372
wlan.8021x.peap.privkey_password....1373
wlan.8021x.peap.peap_username....1374
wlan.8021x.ttls_anonymous_identity....1375
wlan.active_channels....1376
wlan.adhoc_last_channel....1377
wlan.authenticated....1378
wlan.authentication_error....1379
wlan.available....1380
wlan.allowed_band....1381
wlan.adhocautomode....1382
wlan.adhocchannel....1383
wlan.associated....1384
wlan.auth_type....1385
wlan.band_preference....1386

wlan.bssid....1387
wlan.channel....1388
wlan.channel_mask....1389
wlan.country_code....1390
wlan.current_tx_rate....1391
wlan.enable....1392
wlan.encryption_index....1393
wlan.encryption_key....1394
wlan.encryption_mode....1396
wlan.essid....1397
wlan.firmware_version....1398
wlan.ip.addr....1399
wlan.ip.arp_interval....1400
wlan.ip.default_addr_enable....1401
wlan.ip.dhcp.arp_verify....1402
wlan.ip.dhcp.cache_ip....1403
wlan.ip.dhcp.cid_all....1404
wlan.ip.dhcp.cid_enable....1405
wlan.ip.dhcp.cid_prefix....1406
wlan.ip.dhcp.cid_suffix....1407
wlan.ip.dhcp.cid_type....1408
wlan.ip.dhcp.lease.last_attempt....1409
wlan.ip.dhcp.lease.length....1410
wlan.ip.dhcp.lease.server....1411
wlan.ip.dhcp.lease.time_left....1412
wlan.ip.dhcp.option12....1413
wlan.ip.dhcp.option12_format....1414
wlan.ip.dhcp.option12_value....1415
wlan.ip.dhcp.request_timeout....1416
wlan.ip.dhcp.requests_per_session....1417
wlan.ip.dhcp.session_interval....1418
wlan.ip.dns.domain....1419
wlan.ip.dns.domain_user_value....1420
wlan.ip.dns.servers....1421

wlan.ip.dns.servers_user_value....1422
wlan.ip.gateway....1423
wlan.ip.netmask....1424
wlan.ip.port....1425
wlan.ip.port_alternate....1426
wlan.ip.port_json_config....1427
wlan.ip.protocol....1428
wlan.ip.timeout.enable....1429
wlan.ip.timeout.value....1430
wlan.ip.wins.addr....1431
wlan.ip.wins.permanent_source....1432
wlan.ipv6.addresses....1433
wlan.ipv6.address_type....1434
wlan.ipv6.dhcp.lease.last_attempt....1435
wlan.ipv6.dhcp.lease.length....1436
wlan.ipv6.dhcp.lease.time_left....1437
wlan.ipv6.dhcp.option39_enable....1438
wlan.ipv6.dhcp.option39_format....1439
wlan.ipv6.dhcp.option39_fqdn....1440
wlan.ipv6.dhcp.option39_value....1441
wlan.ipv6.gateways....1442
wlan.ipv6.static.addresses....1443
wlan.ipv6.static.gateways....1444
wlan.keep_alive.enable....1445
wlan.keep_alive.timeout....1446
wlan.kerberos.kdc....1447
wlan.kerberos.mode....1448
wlan.kerberos.password....1449
wlan.kerberos.realm....1450
wlan.kerberos.username....1451
wlan.leap_mode....1452
wlan.leap_password....1453
wlan.leap_username....1454
wlan.mac_addr....1455

wlan.mac_raw....1456
wlan.operating_mode....1457
wlan.password....1458
wlan.permitted_channels....1459
wlan.pmf....1460
wlan.poor_signal_threshold....1461
wlan.preamble....1462
wlan.private_key_password....1463
wlan.region_code....1464
wlan.roam.interchannel_delay....1465
wlan.roam.interval....1466
wlan.roam.max_chan_scan_time....1467
wlan.roam.max_fail....1468
wlan.roam.monitor....1469
wlan.roam.neighbor_assist....1470
wlan.roam.rssi....1471
wlan.roam.signal....1472
wlan.rts_cts_enabled....1473
wlan.security....1474
Supporting SGDs for Different Security Types....1476
wlan.signal_noise....1483
wlan.signal_quality....1484
wlan.signal_strength....1485
wlan.station_name....1486
wlan.translation_disable_clear....1487
wlan.tx_power....1488
wlan.tx_rate....1489
wlan.user_channel_list....1490
wlan.username....1491
wlan.wep.auth_type....1492
wlan.wep.index....1493
wlan.wep.key1....1494
wlan.wep.key2....1495
wlan.wep.key3....1496

wlan.wep.key4 1497
wlan.wpa.psk 1498
wlan.wep.key_format 1499
wlan.wpa.groupkey_ciphersuite 1500
wlan.wpa.pairwise_ciphersuite 1501
wlan.wpa.timecheck 1502
wlan.wpa.wpa_version 1503

SGD RFID Commands 1504
rfid.antenna_sweep 1505
rfid.country_code 1506
rfid.enable 1507
rfid.error.response 1508
rfid.hop_table_version 1509
rfid.position.program 1510
rfid.reader_1.antenna_port 1512
rfid.reader_1.power.read 1514
rfid.reader_1.power.write 1516
rfid.reader_1.power.single_power 1518
rfid.reader_1.firmware_version 1519
rfid.reader_1.hardware_version 1520
rfid.reader_1.model 1521
rfid.recipe_version 1522
rfid.region_code 1523
rfid.tag.calibrate 1524
rfid.tag.data 1525
rfid.tag.read.content 1526
rfid.tag.read.execute 1527
rfid.tag.read.result_line1 1528
rfid.tag.read.result_line2 1529
rfid.tag.read.result_line1_alternate 1530
rfid.tag.read.result_line2_alternate 1531
rfid.tag.test 1532
rfid.tag.test.content 1533

rfid.tag.test.execute ... 1534
rfid.tag.test.result_line1 ... 1535
rfid.tag.test.result_line2 ... 1536
rfid.tag.type ... 1537
rfid.log.enabled ... 1539
rfid.log.entries ... 1540
rfid.log.clear ... 1541

Zebra Code Pages ... 1542
Zebra Code Page 850 — Latin Character Set ... 1543
Zebra Code Page 1250 — Central and Eastern European Latin Character Set ... 1545
Zebra Code Page 1252— Latin Character Set ... 1547
Zebra Code Page 1253 — Modern Greek Character Set ... 1549
Zebra Code Page 1254 — Turkish Character Set ... 1551
Zebra Code Page 1255 — Hebrew Character Set ... 1553

ASCII ... 1555

Fonts and Barcodes ... 1557
Standard Printer Fonts ... 1557
Proportional and Fixed Spacing ... 1558
Scalable Versus Bitmapped Fonts ... 1560
Font Matrices ... 1560
Barcodes ... 1562
Basic Format for Bar Codes ... 1563
Barcode Field Instructions ... 1564
Bar Code Command Groups ... 1565

Mod 10 and Mod 43 Check Digits ... 1568
Mod 10 Check Digit ... 1568
Mod 43 Check Digit ... 1569

Error Detection Protocol....1571
Introduction....1571
What is a Protocol?....1571
How Protocols Work....1571
Request Packet Formats from the Host Computer....1572
Header Block Fields....1572
Data Block Fields....1573
Response From the Zebra Printer....1573
Zebra Packet Response....1573
Header Block Fields....1574
Data Block Fields....1574
Disguising Control Code Characters....1575
Error Detection Protocol Application....1576
Error Conditions and System Faults....1576
How the Zebra Printer Processes a Request Packet....1577
How the Zebra Printer Responds to Host Status....1578

ZB64 Encoding and Compression....1579
Introduction to B64 and Z64....1579
B64 and Z64 Encoding....1581

Field Interactions....1583

Real Time Clock....1590
Control Panel Programming....1591
Real Time Clock Parameters....1591
Idle Display....1591
RTC Date....1591
RTC Time....1592
RTC General Information....1593
First Day of the Week Affects Calendar Week....1594
Time and Date Precision....1595
ZPL II Samples....1596

ZBI Character Set **1600**

SGD Command Support **1603**
SGDs Supported for Industrial Printers 1603
SGDs Supported for Desktop Printers 1627
SGDs Supported for Mobile Printers 1651

Mirror **1676**
Mirror Overview 1676
Benefits 1676
Professional Services for Mirror Configuration 1677
Requirements 1677
Supported Printers and Print Server Types 1679
How Mirror Works 1680
Mirror Process Summary 1681
Mirror Process Details 1681
Creating ZPL Files for Use in the <update-root>/files Directory 1682
Configuration 1685
Mirror FTP Server Configuration 1685
Mirror Printer Configuration 1685
The Feedback.get File 1686
Example Feedback.get file 1686
How to Set Up and Use Mirror 1687

Wireless Markup Language (WML) **1693**
WML Overview 1693
WML Details 1693
Supported Printers 1694
Professional Services for WML Content Creation 1695
WML Tags 1695
Using WML 1695
Create a Sample index.wml File 1695
Prepare the Printer to Receive WML Content via FTP 1696

Send WML Content to the Printer via FTP..... 1697
Resetting the ip.ftp.execute_file Setting..... 1698
Sending WML Content to the Printer via the CISDFCRC16 Command..... 1698
Retrieving WML Content from the Printer using the file.type Command..... 1699
Using .nrd Files from WML Menus.....1699
Removing WML or .nrd Files from the Printer using the file.delete Command..... 1700
WML Examples..... 1700
Example 1.....1700
Example 2.....1701
Example 3.....1702
Example 4.....1704
Example 5.....1706
Troubleshooting Scenarios.....1710

Using Weblink..... 1712
When Should Weblink be Used?.....1712
Configuring Weblink.....1712
Basic Configuration.....1712
When a Proxy Server is Part of the Network Configuration.....1713
When HTTP Authentication is Necessary..... 1714
Additional Firewall Configuration.....1714
Difference Between Conn1 and Conn2.....1715
Enable Logging.....1715
Navigating the Log Output.....1716
SSL/TLS Certificate Errors.....1716
Other Typical Errors..... 1718
Troubleshooting.....1719
HTTP Messages.....1720

HTTP POST Alerts.....1722
Configuring Alerts Where the Alert Destination is HTTP POST.....1722
How to Parse via PHP..... 1723
Basic Configuration.....1723

When a Proxy Server is Part of the Network Configuration....1724
When HTTP Authentication is Necessary....1725
Enabling Logging....1726
Navigating the Log Output....1726
Understanding Errors in the Alerts HTTP Log....1727
Troubleshooting....1727
HTTP Messages....1728

Advanced Techniques....1729
Special Effects for Print Fields....1729
Serialized Data....1729
Variable Data....1730
Stored Formats....1730
Initialize/Erase Stored Formats....1730
Download Format Command....1730
Field Number Command....1730
Recall Stored Format Command....1731
Control Commands....1731
Test and Setup Commands....1732
Calibration and Media Feed Commands....1733
Cancel/Clear Commands....1733
Printer Control Commands....1733
Set Dots/Millimeter....1734
Host Status Commands....1735
Changing Delimiters and Command Prefixes....1735
Communication Diagnostics Commands....1736
Graphic Commands....1736
Image Move....1737
Working with Label Formats as Graphics....1737
Working with Hex Graphic Images....1737
Alternative Data Compression Scheme for ~DG and ~DB Commands....1737
Recalling a Hexadecimal Graphic Image....1738
Reducing Download Time of Graphic Images....1738
Transferring Object Between Storage Devices....1739

Deleting Graphics from Memory....1739
Defining and Using the AUTOEXEC.ZPL Function....1740
Memory, Flash Cards, and Font Cards....1740
Shortcuts and Alternate Schemes for Writing ZPL II Scripts....1742
Font Shortcuts....1743

PDFium License....1745

Introduction

This guide is the unabridged, alphabetical reference of programming commands supported in the firmware. This includes all ZPL commands and SGD commands.

IMPORTANT: These are important points to note when using ZPL and SGD commands:

- ZPL and SGD commands should be sent to the printer as separate files.
- Certain settings can be controlled by both ZPL and SGD. Configuration changes made in ZPL can affect configuration changes made in SGD.

To contact Zebra or for technical support, visit www.zebra.com/contact.

Firmware

You can find the printer's firmware version by printing a configuration label. For instructions to do so, see your printer's user guide. For firmware upgrades go to: www.zebra.com/firmware.

IMPORTANT: These are important points to note when using a Zebra G-Series printer:

- You can send instructions to the printer using multiple programming languages: EPL, ZPL, or SGD. EPL and ZPL commands configure the printer, print labels, and get device status information. SGD commands set and get configuration details. These three languages can be used without the need to send the printer instructions to switch from one language to another.
- EPL, ZPL, and SGD commands must be sent to the printer as separate files. They cannot be used together in one format, or set of commands. For example, if you send a series of SGD commands to the printer and they are followed by a printable format, this needs to be done using separate files.

Many text editors and word processors can recreate most examples in this guide in ASCII format. However, for other encodings such as Unicode, a text editor such as Microsoft Notepad is needed.

Who Should Use This Document

This is for programmers who are familiar working with programming languages.

ZPL Commands

This section contains the complete alphabetical listing of ZPL II commands.

How Commands Are Documented

Description: The first paragraph(s) provides an explanation of how the command is used, what it is capable of, and any defining characteristics it has.

Format: The format explains how the command is syntactically arranged and what parameters it contains. For example, the ^B8 command prints a EAN-8 bar code. The format of the ^B8 command is: ^B8o,h,f,g. It is arranged with the caret symbol (^), the command code (B8), and the parameters and are replaced with supported values.

Parameters: In the parameters table, if a command has values that can be defined to make its function more specific, these are outlined as parameters.

Still using the ^B8 example, the h parameter is defined as:

h = bar code height (in dots)	**Values:** 1 to 32000 **Default:** value set by ^BY

If the command has no parameters – for example ~JA (Cancel All) – the parameter section is removed, indicating that the format of the command (~JA) is acceptable ZPL II code.

Examples: When the command is best clarified in context, an example of the ZPL II code is provided. Text indicating exact code entered is printed in an easily recognizable Courier font. An example of code using the ^B8 command looks like this:

```
^XA
^FO50,50
^B8N,100,Y,N
^FD1234567^FS
^XZ
```

Notice that the ^B8 parameter letters have been replaced with real values that apply to the command. In this example N,100,Y,N have been entered.

Comments: A Comments section (if used) will show notes that are of value to a programmer, warnings of potential command interactions, or command-specific information that should be taken into consideration.

Comments are also included next to parameters if they apply directly to a particular setting.

Basic ZPL Exercises and Examples

The purpose of these exercises is to introduce basic ZPL commands to novice ZPL users.

Make sure this checklist is complete

- Load the printer with labels that are big enough to give you ample space to work with.
- Print a configuration label.
- Look at the configuration label and make sure that the `LEFT POSITION` is set to `000` and `LABEL TOP` is set to `000`.
- Determine the printer's resolution. It is listed on the configuration label. 8/MM = 200 dpi, `12/MM` = 300 dpi and `24/MM` = 600 dpi.

Tips

These are some tips when using ZPL:

- Use the DOS text editor to write ZPL files.
- Save the file as a `.txt` file and copy it to the printer from DOS command line.

Before You Begin

Some things that are important to understand before you begin are:

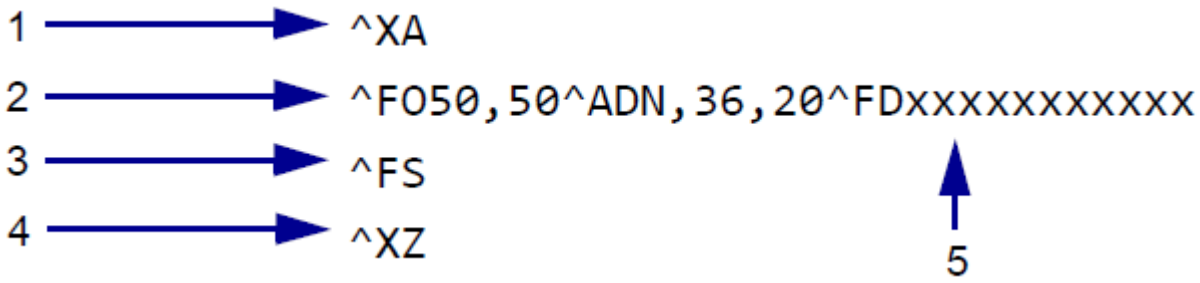
- 200 dpi means the resolution of the printhead is 200 dots per inch. If you program the printer to draw a line 100 dots long that equals a half inch. 100 dots on a 300 dpi printer prints a line 1/3 inch long.
- The home position that all your coordinates are referencing is at the left-hand trailing edge of the label as the label comes out of the printer. (There are some exceptions to this.)

Exercises

The exercises start simple and gradually progress to give you an opportunity to try a variety of commonly used ZPL commands. Not all commands are covered, but this should be a good core of commands to learn. Some commands may not be supported due to the firmware version in your printer.

Exercise 1: Specify a Location for an Entered Name

1. Print your name on the label.
2. Start by printing just your name on the label. Use this format as a model:
3. Send this format to the printer:



1	Every format starts with the ^XA command
2	^FO (field origin) command
3	^FS (field separator) command
4	Every format ends with the ^XZ command
5	^FD (field data) command. Insert your name in place of the xxxxxxxxxxx shown.

4. When the label prints correctly, alter the first number after the `^FOx.` See how that change affects the print position. Alter the second number after the `^FO50,x` and see how that the print position.

Font instruction

`^ADN`

1. Alter the numbers after the `^ADN,x,x` command.
   - 18,10 is the smallest size you can make the `D` font.
   - The first number is the height of the font in dots. The second number is the width in dots.
   - You can use direct multiples up to ten times that size as a maximum.

     180,100 is the largest you can make the `D` font.
   - 25,18 would not be a valid size. The printer rounds to the next recognizable size.
2. Check the font matrices tables for other fonts to try. See Fonts and Barcodes on page 1557.
3. Try the zero scalable font `^A0N,x,x`.

   This font is scalable, and you can choose any height and width.

Rotation Commands

1. Change `^ADN` to `^ADR`, and then `^ADI`, and then `^ADB`.

   See how the print position changes.
2. Add more fields.

3. Add two more fields to print directly under your name using the ^ADN,36,20 font and size:

   Your street address

   Your city, state, zip

4. You must add two more lines of code that start off with:

```
^XA
^FO50,50^ADN,36,20^FDxxxxxxxxxxx^FS
^FO    (fill in the rest)
^FO    (fill in the rest)
^XZ
```

   Make sure all these fields print in the same font and size and left side of fields has same vertical alignment.

   Your name

   1200 W Main Street

   Anytown, Il 60061

Reverse Printing a Field

Write the following format and send to the printer:

```
^XA
^PR1
^FO100,100
^GB70,70,70,,3^FS
^FO200,100
^GB70,70,70,,3^FS
^FO300,100
^GB70,70,70,,3^FS
^FO400,100
^GB70,70,70,,3^FS
^FO107,110^CF0,70,93
^FR^FDREVERSE^FS
^XZ
```

To see the effects, remove:

```
^FR^FDREVERSE^FS
```

To see the effects, try removing one of the ^GB lines of code.

Label Reverse Print

Write the following format and send to the printer:

```
^XA^LRY
^FO100,50
^GB195,203,195^FS
```

```
^FO180,110^CFG
^FDLABEL^FS
^FO130,170
^FDREVERSE^FS
^XZ
```

To see the effects, remove:

```
^GB195,203,195^FS
```

Mirror Image of Label

Write the following format and send to the printer:

```
^XA^PMY
^FO100,100
^CFG
^FDMIRROR^FS
^FO100,160
^FDIMAGE^FS
^XZ
```

To see the effects, in the first line of code change ^PMY to ^PMN.

Print Orientation

Write the following format and send to the printer:

```
^XA^CFD
^POI
^LH330,10
^FO50,50
^FDZEBRA TECHNOLOGIES^FS
^FO50,75
^FDVernon Hills, IL^FS
^XZ
```

To see the effects, in the second line of code change ^POI to ^PON.

Exercise 2: Boxes and Lines

Use the address format from <X-refBlue>Exercise .

Add this new line to your existing format:

```
^FO50,200^GB200,200,2^FS
```

This prints a box one wide by one inch long and the thickness of the line is 2 dots.

Reposition and resize the square so that it goes around the name and address uniformly.

Print a line by adding:

```
^FO50,300^GB400,1,4,^FS
```

This prints a horizontal line two inches wide by 4 dots thick.

Print a vertical line using this code:

```
^FO100,50^GB1,400,4^FS
```

Exercise 3: Bar Codes — ^B3 Code 39 Barcode

Write the following format and send to the printer:

```
^XA
^FO50,50^B3N,N,100,Y,N^FD123456^FS
^XZ
```

Try changing each of the parameters in the ^B3 string so you can see the effects.

```
^B3o,e,h,f,g
^BY
```

For valid parameter choices, see ^B3.

Insert the ^BY command just before the ^B3 to see how the narrow bar width can be altered.

```
^FO50,50^BY2^B3..etc ^BYx
```

Acceptable values for x are 1 through 10

Alter the ratio of the narrow to wide bar.

```
^FO50,50^BY2,3^B3..etc ^BY2,x
```

Acceptable values for x are 2.1 through 3 in .1 increments

Print out a ^B3 bar code with the interpretation line on top of the bar code and the bar code rotated 90 degrees.

Add a ^PQ just before the ^XZ to print several labels.

```
^PQ4
^XZ
^PR Print rate (in inches per second)
```

Add a ^PR command after the ^XA at the beginning of the format to change the print rate (print speed).

```
^XA
^PR4
```

then try ^PR6. ^PRx acceptable values for x are 2 through 12 (check printer specs)

See how the print speed affects the print quality of the bar code. You may need to increase the printer darkness setting at higher print speeds.

Exercise 4: ^SN — Serial Number Command

Send this format to the printer:

```
^XA
^FO100,100^ADN,36,20^SN001,1,Y^FS
^PQ3
^XZ
```

To vary the ^SNv,n,z to exercise increment/decrement and leading zeros functions, consult this guide.

If your serial number contains alpha and numeric characters, you can increment or decrement a specific segment of the data even if it is in the middle, as this sample sequence shows:

```
ABCD1000EFGH, ABCD1001EFGH, ABCD1002EFGH
```

Send this file to the printer and to see how it increments the serial number. The ^SF command can also work with alpha characters.

```
^XA
^FO100,100^ADN,36,20^FDABCD1000EFGH^SF%%%%dddd%%%%,10000^FS
^PQ15
^XZ
```

Notice how the field data character position aligns with the ^SF data string:

^	F	D	A	B	C	D	1	0	0	0	E	F	G	H
^	S	F	%	%	%	%	d	d	d	d	%	%	%	%
										1	0	0	0	0
										2	0	0	0	0
										3	0	0	0	0

And on through...

							1	0	1	4	0	0	0	0

The last label prints **ABCD1014EFGH**.

The % is placed in positions that you do not want to increment or decrement, `d` = decimal, `10000` = increment value.

For more details on ^SF, see ^SF.

Exercise 5: Saving a template to memory. ^IS and image save and image load.

This exercise helps you troubleshoot your code against the errors you see on your labels.

Send this format to the printer:

```
^XA
^FO20,30^GB750,1100,4^FS
^FO20,30^GB750,200,4^FS
^FO20,30^GB750,400,4^FS
^FO20,30^GB750,700,4^FS
^FO20,226^GB325,204,4^FS
^FO30,40^ADN,36,20^FDShip to:^FS
^FO30,260^ADN,18,10^FDPart number #^FS
^FO360,260^ADN,18,10^FDDescription:^FS
^FO30,750^ADN,36,20^FDFrom:^FS
^ISR:SAMPLE.GRF^FS
^XZ
```

Send this format:

```
^XA
^ILR:SAMPLE.GRF^FS
^FO150,125^ADN,36,20^FDAcme Printing^FS
^FO60,330^ADN,36,20^FD14042^FS
^FO400,330^ADN,36,20^FDScrew^FS
^FO70,480^BY4^B3N,,200^FD12345678^FS
^FO150,800^ADN,36,20^FDMacks Fabricating^FS
^XZ
```

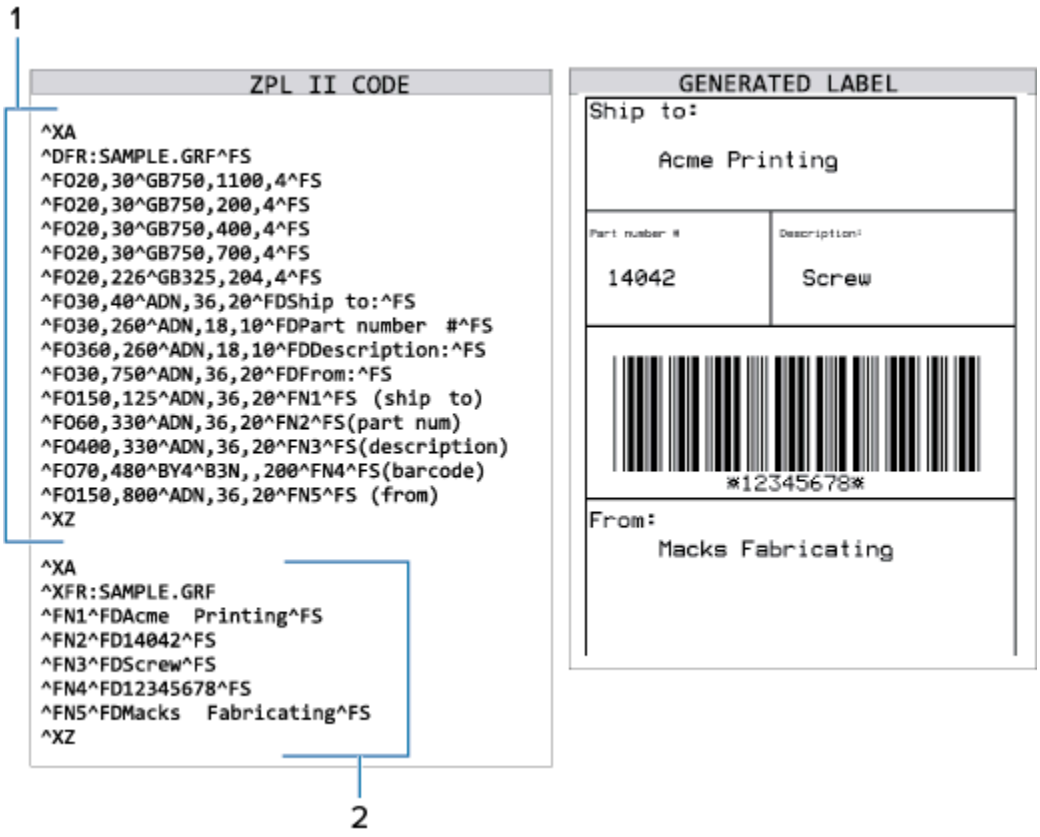
In this way the template only needs to be sent one time to the printer's memory. Subsequent formats can be sent recalling the template and merging variable data into the template. In this exercise, the file was saved in the printers `R:` memory, which is volatile.

Exercise 6: ^DF and ^XF — Download Format and Recall Format

Similar concept to ^IS and ^IL command. ^IS and ^IL processes faster in the printer than ^DF and ^XF.

This is how the ^DF and ^XF format structure produces a label similar to the ^IS/^IL sample you just tried.

Figure 1 Download and Recall Format



1	Download format code
2	Recall format call that generates the generated label.

Exercise 7: Asian and Unicode Encodings

.14↑

This exercise works on printers with firmware version V60.14, V50.14, or later.

In each of the following examples, the format is saved in the corresponding encoding before being sent down to the printer and the ZPL script was made in Microsoft Notepad, a basic text editor. The characters were inserted from the character map in Windows or typed from the keyboard.

Example: This is an example of using an Asian encoding, such as UHANGUL, with ASCII text. Using the `CI26` command tells the printer to recognize any byte less than 7F as ASCII text and every byte above as the first byte of UHANGUAL encoding:

ZPL II CODE	GENERATED LABEL
^XA^CW1,B:ANMDK.TTF ^SEB:UHANGUL.DAT^CI26 ^FO100,100^A1,50,50^FDASCII 한국어^FS ^XZ	ASCII한국어

Example: This is an example of using the Unicode encoding, UTF-8:

ZPL II CODE
^XA^CW1,ANMDJ.TTF^CI28 ^FO100,50^A1,30,30^FDENGLISH/日本語/한국어/简体中文/浪體中文^FS ^XZ

GENERATED LABEL
ENGLISH/日本語/한국어/简体中文/浪體中文

Allowed Characters in File Names

Files on the internal printer drives (R:, E:, etc.) can be created or accessed using several different commands (for example, ^DF, ^XF, ^TO, etc.). The names of the file can contain ONLY the characters shown here

Shaded areas indicate characters which cannot be used. The command and control characters (normally ^ and ~) cannot be used unless the control characters are changed to something else using the ^CC~CC ZPL command.

Char.	DEC	OCT	HEX		Char.	DEC	OCT	HEX		Char.	DEC	OCT	HEX
(sp)	32	0040	0x20		@	64	0100	0x40		`	96	0140	0x60
!	33	0041	0x21		A	65	0101	0x41		a	97	0141	0x61
"	34	0042	0x22		B	66	0102	0x42		b	98	0142	0x62
#	35	0043	0x23		C	67	0103	0x43		c	99	0143	0x63
$	36	0044	0x24		D	68	0104	0x44		d	100	0144	0x64
%	37	0045	0x25		E	69	0105	0x45		e	101	0145	0x65
&	38	0046	0x26		F	70	0106	0x46		f	102	0146	0x66
'	39	0047	0x27		G	71	0107	0x47		g	103	0147	0x67
(	40	0050	0x28		H	72	0110	0x48		h	104	0150	0x68
)	41	0051	0x29		I	73	0111	0x49		i	105	0151	0x69
*	42	0052	0x2a		J	74	0112	0x4a		j	106	0152	0x6a
+	43	0053	0x2b		K	75	0113	0x4b		k	107	0153	0x6b
,	44	0054	0x2c		L	76	0114	0x4c		l	108	0154	0x6c
-	45	0055	0x2d		M	77	0115	0x4d		m	109	0155	0x6d
.	46	0056	0x2e		N	78	0116	0x4e		n	110	0156	0x6e
/	47	0057	0x2f		O	79	0117	0x4f		o	111	0157	0x6f
0	48	0060	0x30		P	80	0120	0x50		p	112	0160	0x70
1	49	0061	0x31		Q	81	0121	0x51		q	113	0161	0x71
2	50	0062	0x32		R	82	0122	0x52		r	114	0162	0x72
3	51	0063	0x33		S	83	0123	0x53		s	115	0163	0x73
4	52	0064	0x34		T	84	0124	0x54		t	116	0164	0x74
5	53	0065	0x35		U	85	0125	0x55		u	117	0165	0x75
6	54	0066	0x36		V	86	0126	0x56		v	118	0166	0x76
7	55	0067	0x37		W	87	0127	0x57		w	119	0167	0x77
8	56	0070	0x38		X	88	0130	0x58		x	120	0170	0x78
9	57	0071	0x39		Y	89	0131	0x59		y	121	0171	0x79
:	58	0072	0x3a		Z	90	0132	0x5a		z	122	0172	0x7a
;	59	0073	0x3b		[	91	0133	0x5b		{	123	0173	0x7b
<	60	0074	0x3c		\	92	0134	0x5c			124	0174	0x7c
=	61	0075	0x3d		]	93	0135	0x5d		}	125	0175	0x7d
>	62	0076	0x3e		^	94	0136	0x5e		~	126	0176	0x7e
?	63	0077	0x3f		_	95	0137	0x5f		(del)	127	0177	0x7f

^A

The ^A command specifies the font to use in a text field. ^A designates the font for the current ^FD statement or field. The font specified by ^A is used only once for that ^FD entry. If a value for ^A is not specified again, the default ^CF font is used for the next ^FD entry.

Scalable/Bitmapped Font

Format: ^Afo,h,w

Parameter	Details
f = font name	**Values:** A through Z, and 0 to 9 Any font in the printer (downloaded, EPROM, stored fonts, fonts A through Z and 0 to 9). **IMPORTANT:** Parameter f is required. If f is omitted it defaults to the last value of the ^CF command.
o = field orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from bottom up, 270 degrees **Default:** the last accepted ^FW value or the ^FW default
h = Character Height (in dots)	**Scalable** **Values:** 10 to 32000 **Default:** last accepted ^CF **Bitmapped** **Values:** multiples of height from 1 to 10 times the standard height, in increments of 1 **Default:** last accepted ^CF
w = width (in dots)	**Scalable** **Values:** 10 to 32000 **Default:** last accepted ^CF **Bitmapped** **Values:** multiples of width from 1 to 10 times the standard width, in increments of 1 **Default:** last accepted ^CF

Scalable Font Command

Example: This is an example of a scalable font command:

```
^XA
^FO50,50
^A0,32,25
^FDZEBRA^FS
^FO50,150
^A0,32,25
^FDPROGRAMMING^FS
^FO50,250
^A0,32,25^FDLANGUAGE^FS
^XZ
```

Bitmap Font Command

Example: This is an example of a bitmap font command:

```
^XA
^FO50,50
^ADN,36,20
^FDZEBRA^FS
^FO50,150
^ADN,36,20
^FDPROGRAMMING^FS
^FO50,250
^ADN,36,20^FDLANGUAGE^FS
^XZ
```

Comments:

Fonts are built using a matrix that defines standard height-to-width ratios. If you specify only the height or width value, the standard matrix for that font automatically determines the other value. If the value is not given or a 0 (zero) is entered, the height or width is determined by the standard font matrix.

.14↑

This command interacts with the justification parameters of ^F0 and ^FT and with the field direction parameter of ^FP. For output and examples, see Field Interactions on page 1583.

^A@

The ^A@ command uses the complete name of a font, rather than the character designation used in ^A. Once a value for ^A@ is defined, it represents that font until a new font name is specified by ^A@.

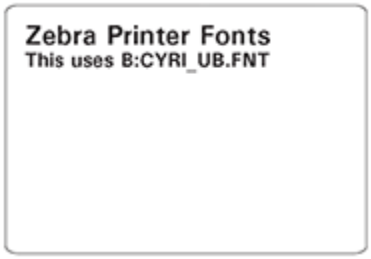
Use Font Name to Call Font

Format: ^A@o,h,w,d:f.x

Parameter	Details
o = field orientation	**Values:** N = normal R = rotates 90 degrees (clockwise) I = inverted 180 degrees B = read from bottom up, 270 degrees **Default:** N or the last ^FW value
h = character height (in dots)	**Default:** Specifies magnification by w (character width) or the last accepted ^CF value. Uses the base height if none is specified. • Scalable - The value is the height in dots of the entire character block. Magnification factors are unnecessary, because characters are scaled. • Bitmapped - The value is rounded to the nearest integer multiple of the font's base height, then divided by the font's base height to give a magnification nearest limit.
w = width (in dots)	**Default:** Specifies magnification by h (height) or the last accepted ^CF value. Specifies the base width is used if none is specified. • Scalable - The value is the width in dots of the entire character block. Magnification factors are unnecessary, because characters are scaled. • Bitmapped - The value rounds to the nearest integer multiple of the font's base width, then divided by the font's base width to give a magnification nearest limit.
d = drive location of font	**Values:** R:, E:, B:, and A: **Default:** R:
f = font name	**Values:** any valid font **Default:** if an invalid or no name is entered, the default set by ^CF is used If no font has been specified in ^CF, font A is used. The font named carries over on all subsequent ^A@ commands without a font name.
x = extension .TTE is only supported in firmware version V60.14.x, V50.14.x, or later.	**Values:** .FNT = font .TTF = TrueType Font .TTE = TrueType Extension

Example: This example identifies the purpose of each line of code for this label:

```
1→  ^XA
2→  ^A2N,50,50,B:CYRI_UB.FNT
3→  ^FO100,100
4→  ^FDZebra Printer Fonts^FS
5→  ^A2N,40,40
6→  ^FO100,150
7→  ^FDThis uses B:CYRI_UB.FNT^FS
8→  ^XZ
```



1	Starts the label format.
2	Searches non-volatile printer memory (`B:`) for `CYRI_UB.FNT`. When the font is found, the ^A@ command sets the print orientation to normal and the character size to 50 dots by 50 dots.
3	Sets the field origin at 100,100.
4	Prints the field data, **Zebra Printer Fonts** on the label.
5	Calls the font again and character size is decreased to 40 dots by 40 dots.
6	Sets the new field origin at 100,150.
7	Prints the field data, **This uses the B:CYRI_UB.FNT** on the label.
8	Ends the label format.

For reference, see Zebra Code Page 850 — Latin Character Set on page 1543, Fonts and Barcodes on page 1557, and ASCII on page 1555.

^B0

The ^B0 command creates a two-dimensional matrix symbology made up of square modules arranged around a bulls-eye pattern at the center.

Aztec Barcode Parameters

NOTE: The Aztec barcode works with firmware version V60.13.0.11A and V50.13.2 or later.

Format: ^B0a,b,c,d,e,f,g

Parameters	Details
a = orientation	**Values:** N = normal R = rotated I = inverted 180 degrees B = read from bottom up, 270 degrees **Default:** current ^FW value
b = magnification factor	**Values:** 1 to 10 **Default:** 1 on 150 dpi printers 2 on 200 dpi printers 3 on 300 dpi printers 6 on 600 dpi printers
c = extended channel interpretation code indicator	**Values:** Y = if data contains ECICs N = if data does not contain ECICs **Default:** N
d = error control and symbol size/type indicator	**Values:** 0 = default error correction level 01 to 99 = error correction percentage (minimum) 101 to 104 = 1 to 4-layer compact symbol 201 to 232 = 1 to 32-layer full-range symbol 300 = a simple Aztec “Rune” **Default:** 0
e = menu symbol indicator	**Values:** Y = if this symbol is to be a menu (bar code reader initialization) symbol N = if it is not a menu symbol **Default:** N
f = number of symbols for structured append	**Values:** 1 through 26 **Default:** 1

Parameters	Details
g = optional ID field for structured append	The ID field is a text string with a 24-character maximum **Default:** no ID

Example

This is an example of the ^B0 command:

ZPL II CODE	GENERA TED LABEL
^XA ^B0R,7,N,0,N,1,0 ^FD 7. This is testing label 7^FS ^XZ	

^B1

The ^B1 command produces the Code 11 bar **code**, also known as USD-8 code. In a Code 11 barcode, each character is composed of three bars and two spaces, and the character set includes 10 digits and the hyphen (-).

Code 11 Barcode

- ^B1 supports print ratios of 2.0:1 to 3.0:1.
- Field data (^FD) is limited to the width (or length, if rotated) of the label.

IMPORTANT: If additional information about this barcode is required, go to aimglobal.org.

Format: ^B1o,e,h,f,g

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from bottom up, 270 degrees **Default: current ^FW value**
e = check digit	**Values:** Y = 1 digit N = 2 digits **Default:** N
h = barcode height (in dots)	**Values:** 1 to 32000 **Default:** Value set by ^BY
f = print interpretation line	**Values:** Y = yes N = no **Default:** Y
g = print interpretation line above code	**Values:** Y = yes N = no **Default:** N

Example: This is an example of the Code 11 barcode:

ZPL II CODE	CODE 11 BARCODE
^XA ^FO100,100^BY3 ^B1N,N,150,Y,N ^FD123456^FS ^XZ	

CODE 11 BARCODE CHARACTERS

0 1 2 3 4 5 6 7 8 9 -

Internal Start/Stop Character: △

When used as a stop character:

△ is used with 1 check digit

△ is used with 2 check digits

^B2

The ^B2 command produces the Interleaved 2 of 5 bar code, a high-density, self-checking, continuous, numeric symbology.

Interleaved 2 of 5 Bar Code

Each data character for the Interleaved 2 of 5 bar code is composed of five elements: five bars or five spaces. Of the five elements, two are wide, and three are narrow. The bar code is formed by interleaving characters formed with all spaces into characters formed with all bars.

- ^B2 supports print ratios of 2.0:1 to 3.0:1.
- Field data (^FD) is limited to the width (or length, if rotated) of the label.

IMPORTANT: If additional information about this bar code is required, go to aimglobal.org.

Format: ^B2o,h,f,g,e,j

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from bottom up, 270 degrees **Default:** current ^FW value
h = bar code height (in dots)	**Values:** 1 to 32000 **Default:** value set by ^BY
f = print interpretation line	**Values:** Y = yes N = no **Default:** Y
g = print interpretation line above code	**Values:** Y = yes N = no **Default:** N
e = calculate and print Mod 10 check digit	**Values:** Y = yes N = no **Default:** N

Example

This is an example of an Interleaved 2 of 5 bar code:

ZPL II CODE	INTERLEAVE D 2 OF 5 BAR CODE
^XA ^FO100,100^BY3 ^B2N,150,Y,N,N ^FD123456^FS ^XZ	

INTERLEAVE D 2 OF 5 BAR CODE CHARACTERS
0 1 2 3 4 5 6 7 8 9 **Start/Stop (internal)**

The total number of digits in an Interleaved 2 of 5 bar code must be even. The printer automatically adds a leading 0 (zero) if an odd number of digits is received.

The Interleaved 2 of 5 bar code uses the Mod 10 check-digit scheme for error checking. For more information on Mod 10 check digits, see Mod 10 Check Digit on page 1568.

^B3

The Code 39 barcode is the standard for many industries, including the US Department of Defense. It is one of three symbologies identified in the American National Standards Institute (ANSI) standard MH10.8M-1983. Code 39 is also known as USD-3 Code and 3 of 9 Code.

Code 39 Barcode

Each character in a Code 39 barcode is composed of nine elements: five bars, four spaces, and an inter-character gap. Three of the nine elements are wide; the six remaining elements are narrow.

- ^B3 supports print ratios of 2.0:1 to 3.0:1.
- Field data (^FD) is limited to the width (or length, if rotated) of the label.
- Code 39 automatically generates the start and stop character (*).
- Asterisk (*) for start and stop character prints in the interpretation line, if the interpretation line is turned on.
- Code 39 is capable of encoding the full 128-character ASCII set.

IMPORTANT: If additional information about this barcode is required, go to aimglobal.org.

Format: ^B3o,e,h,f,g

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from bottom up, 270 degrees **Default:** current ^FW value
e = Mod-43 check digit	**Values:** Y = yes N = no **Default:** N
h = bar code height (in dots)	**Values:** 1 to 32000 **Default:** value set by ^BY
f = print interpretation line	**Values:** Y = yes N = no **Default:** Y
g = print interpretation line above code	**Values:** Y = yes N = no **Default:** N

Example

This is an example of a Code 39 barcode:

ZPL II CODE	CODE 39 BAR CODE
^XA ^FO100,100^BY3 ^B3N,N,100,Y,N ^FD123ABC^FS ^XZ	

CODE 39 BAR CODE CHARACTERS
0 1 2 3 4 5 6 7 8 9 A B C D E F G H I J K L M N O P Q R S T U V W X Y Z - . $ / + % Space

Comments: Extended ASCII is a function of the scanner, not of the barcode. Your scanner must have extended ASCII enabled for this feature to work. To enable extended ASCII in the Code 39, you must first encode +$ in your ^FD statement. To disable extended ASCII, you must encode -$ in your ^FD statement.

Example

This example encodes a carriage return with line feed into a Code 39 barcode:

ZPL II CODE	GENERA TED LABELS
^XA ^FO20,20 ^B3N,N,100,Y ^FDTEST+$$M$J-$^FS ^XZ	

Full ASCII Mode for Code 39

Code 39 can generate the full 128-character ASCII set using paired characters as shown in this table:

Table 1 Code 39 ASCII Set

ASCII	CODE 39		ASCII	Code 39
SP	Space		SOH	$A
!	/A		STX	$B
"	/B		ETX	$C
#	/C		EOT	$D

Table 1 Code 39 ASCII Set (Continued)

ASCII	CODE 39		ASCII	Code 39
$	/D		ENQ	$E
%	/E		ACK	$F
&	/F		BEL	$G
'	/G		BS	$H
(	/H		HT	$I
)	/I		LF	$J
*	/J		VT	$K
+	/K		FF	$L
,	/L		CR	$M
-	-		SO	$N
.	.		SI	$O
/	/O		DLE	$P
0	0		DC1	$Q
1	1		DC2	$R
2	2		DC3	$S
3	3		DC4	$T
4	4		NAK	$U
5	5		SYN	$V
6	6		ETB	$W
7	7		CAN	$X
8	8		EM	$Y
9	9		SUB	$Z
:	/Z		ESC	%A
;	%F		FS	%B
<	%G		FS	%C
=	%H		RS	%D
>	%I		US	%E
?	%J			
@	%V		'	%W
A	A		a	+A
B	B		b	+B
C	C		c	+C
D	D		d	+D

Table 1 Code 39 ASCII Set (Continued)

ASCII	CODE 39		ASCII	Code 39
E	E		e	+E
F	F		f	+F
G	G		g	+G
H	H		h	+H
I	I		i	+I
J	J		j	+J
K	K		k	+K
L	L		l	+L
M	M		m	+M
N	N		n	+N
O	O		o	+O
P	P		p	+P
Q	Q		q	+Q
R	R		r	+R
S	S		s	+S
T	T		t	+T
U	U		u	+U
V	V		v	+V
W	W		w	+W
X	X		x	+X
Y	Y		y	+Y
Z	Z		z	+Z
[	%K		{	%P
\	%L		|	%Q
]	%M		}	%R
^	%N		~	%S
_	%O		DEL	%T, %T

^B4

The ^B4 command creates a multi-row, continuous, variable-length symbology capable of encoding the full 128-character ASCII set. It is ideally suited for applications requiring large amounts of data in a small space.

Code 49 Barcode

The code consists of two to eight rows. A row consists of a leading quiet zone, four symbol characters encoding eight code characters, a stop pattern, and a trailing quiet zone. A separator bar with a height of one module separates each row. Each symbol character encodes two characters from a set of Code 49 characters.

- ^B4 has a fixed print ratio.
- Rows can be scanned in any order.

IMPORTANT: If additional information about this barcode is required, go to aimglobal.org.

Format: ^B4o,h,f,m

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from bottom up, 270 degrees **Default:** current ^FW value
h = height multiplier of individual rows	**Values:** 1 to height of label **Default:** value set by ^BY This number multiplied by the module equals the height of the individual rows in dots. 1 is not a recommended value.
f = print interpretation line	**Values:** N = no line printed A = print interpretation line above code B = print interpretation line below code **Default:** N When the field data exceeds two rows, expect the interpretation line to extend beyond the right edge of the barcode symbol.

Parameters	Details
m = starting mode	**Values:** 0 = Regular Alphanumeric Mode 1 = Multiple Read Alphanumeric 2 = Regular Numeric Mode 3 = Group Alphanumeric Mode 4 = Regular Alphanumeric Shift 1 5 = Regular Alphanumeric Shift 2 A = Automatic Mode. The printer determines the starting mode by analyzing the field data. **Default:** A

Example

This is an example of a Code 49 barcode:

ZPL II CODE	CODE 49 BAR CODE
^XA ^FO150,100^BY3 ^B4N,20,A,A ^FD12345ABCDE^FS ^XZ	

Field Data Set	Unshifted Character Set	Shift 1 Character Set	Shift 2 Character Set
0	0	’	
1	1	ESC	;
2	2	FS	<
3	3	GS	=
4	4	RS	>
5	5	US	?
6	6	!	@
7	7	“	[
8	8	#	\
9	9	&	]
A	A	SOH	a
B	B	STX	b
C	C	ETX	c
D	D	EOT	d
E	E	ENQ	e
F	F	ACK	f
G	G	BEL	g
H	H	BS	h
I	I	HT	I
J	J	LF	j
K	K	VT	k
L	L	FF	l
M	M	CR	m
N	N	SO	n
O	O	SI	o
P	P	DLE	p
Q	Q	DC1	q
R	R	DC2	r
S	S	DC3	s
T	T	DC4	t
U	U	NAK	u
V	V	SYN	v
W	W	ETB	w
X	X	CAN	x
Y	Y	EM	y
Z	Z	SUB	z
-	-	(	_
.	.	)	‘
SPACE	SPACE	Null	DEL
$	$	*	{
/	/	,	|
++	++	:	}
%	%	reserved	~
< (Shift 1)			
> (Shift 2)			
: (N.A.)			
; (N.A.)			
? (N.A.)			
= (Numeric Shift)			

Code 49 Shift 1 and 2 Character Substitutions

Code 49 Field Data Character Set

The ^FD data sent to the printer when using starting modes 0 to 5 is based on the Code 49 Internal Character Set. This is shown in the first column of the Code 49 table on the previous page. These characters are Code 49 control characters:

: ; < = > ?

Valid field data must be supplied when using modes 0 to 5. Shifted characters are sent as a two-character sequence of a shift character followed by a character in the unshifted character set.

> To encode a lowercase a, send a > (Shift 2) followed by an uppercase A. If interpretation line printing is selected, a lowercase **a** prints in the interpretation line. This reflects what the output from the scanner reads. Code 49 uses uppercase alphanumeric characters only.

If an invalid sequence is detected, the Code 49 formatter stops interpreting field data and prints a symbol with the data up to the invalid sequence. These are examples of invalid sequences:

- Terminating numeric mode with any characters other than 0 to 9 or a Numeric Space.
- Starting in Mode 4 (Regular Alphanumeric Shift 1) and the first field data character is not in the Shift 1 set.
- Starting in Mode 5 (Regular Alphanumeric Shift 2) and the first field data character is not in the Shift 2 set.
- Sending Shift 1 followed by a character not in the Shift 1 set.
- Sending Shift 2 followed by a character not in the Shift 2 set.
- Sending two Shift 1 or Shift 2 control characters.

Advantages of Using the Code 49 Automatic Mode

Using the default (Automatic Mode) completely eliminates the need for selecting the starting mode or manually performing character shifts. The Automatic Mode analyzes the incoming ASCII string, determines the proper mode, performs all character shifts, and compacts the data for maximum efficiency.

Numeric Mode is selected or shifted only when five or more continuous digits are found. Numeric packaging provides no space advantage for numeric strings consisting of fewer than eight characters.

^B5

The ^B5 command is supported in all printers as a resident barcode. Accepted barcode characters are 0-9.

Planet Code Barcode

Format: ^B5o,h,f,g

Parameters	Details
o = orientation code	**Values:** N = normal R = rotated I = inverted 180 degrees B = read from bottom up, 270 degrees **Default:** current ^FW value
h = barcode height (in dots)	**Values:** 1 to 9999 **Default:** value set by ^BY
f = interpretation line	**Values:** N = no Y = yes **Default:** N
g = determines if the interpretation line is printed above the barcode	**Values:** N = no Y = yes **Default:** N

Example

This is an example of a Planet Code barcode:

ZPL II CODE	GENERA TED LABEL
^XA ^FO150,100^BY3 ^B5N,100,Y,N ^FD12345678901^FS ^XZ	

^B7

The ^B7 command produces the PDF417 barcode, a two-dimensional, multirow, continuous, stacked symbology. PDF417 is capable of encoding over 1,000 characters per barcode. It is ideally suited for applications requiring large amounts of information at the time the barcode is read.

PDF417 Bar Code

The barcode consists of three to 90 stacked rows. Each row consists of start and stop patterns and symbol characters called **code-words**. A code word consists of four bars and four spaces. A three-code-word minimum is required per row.

The PDF417 barcode is also capable of using the structured append option (^FM), which allows you to extend the field data limitations by printing multiple barcodes. For more information on using structured append, see ^FM.

- PDF417 has a fixed print ratio.
- Field data (^FD) is limited to 3K of character data.

Format: ^B7o,h,s,c,r,t

Parameters	Details
o = orientation	**Values:** N = normalR = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** current ^FW value
h = bar code height for individual rows (in dots)	**Values:** 1 to height of label **Default:** value set by ^BY This number multiplied by the module equals the height of the individual rows in dots. If this number is not specified, the overall barcode height, divided by the number of rows, equals the height of the individual rows in dots, where the overall barcode height is defined by the ^BY command. 1 is not a recommended value.
s = security level	**Values:** 1 to 8 (error detection and correction) **Default:** 0 (error detection only) This determines the number of error detection and correction code words to be generated for the symbol. The default level provides only error detection without correction. Increasing the security level adds increasing levels of error correction and increases the symbol size.
c = number of data columns to encode	**Values:** 1 to 30 **Default:** 1:2 (row-to-column aspect ratio) You can specify the number of code-word columns giving control over the width of the symbol.

Parameters	Details
r = number of rows to encode	**Values:** 3 to 90 **Default:** 1 : 2 (row-to-column aspect ratio) You can specify the number of symbol rows giving control over the height of the symbol. For example, with no row or column values entered, 72 code words would be encoded into a symbol of six columns and 12 rows. Depending on code words, the aspect ratio is not always exact.
t = truncate right row indicators and stop pattern	**Values:** N = no truncation Y = perform truncation **Default:** N

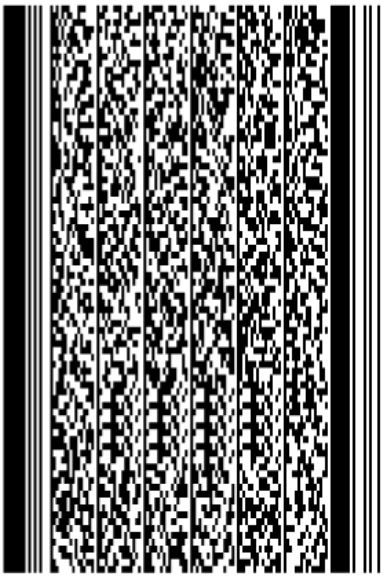
Examples

This is an example of a PDF417 barcode:

ZPL II CODE

```
^XA
^BY2,3
^FO10,10^B7N,5,5,,83,N
^FDZebra Technologies
Corporation strives to be
the expert supplier of
innovative solutions to
speciality demand labeling
and ticketing problems of
business and government.
We will attract and retain
the best people who will
understand our customer's
needs and provide them with
systems, hardware, software,
consumables and service
offering the best value,
high quality, and reliable
performance, all delivered
in a timely manner.
^FS^XZ
```

PDF417 BAR CODE



This is an example of a PDF417 without and with truncation selected:

PDF417 without Truncation being selected

PDF417 with Truncation being selected

This example shows the ^B7 command used with field hex (^FH) characters:

PDF417 without Truncation being selected

PDF417 with Truncation being selected

Comments: Noted in this bulleted list:

- If both columns and rows are specified, their product must be less than 928.
- No symbol is printed if the product of columns and rows is greater than 928.
- No symbol is printed if total code words are greater than the product of columns and rows.

- Serialization is not allowed with this barcode.
- The truncation feature can be used in situations where label damage is not likely. The right row indicators and stop pattern are reduced to a single module bar width. The difference between a non-truncated and a truncated barcode is shown in the previous examples.

Special Considerations for ^BY When Using PDF417

When used with ^B7, the parameters for the ^BY command are:

`w` = **module width (in dots)**

Values: 2 to `10`

Default: 2

`r` = **ratio**

Fixed Value: 3 (ratio has no effect on PDF417)

h = **height of bars (in dots)**

Values: `1` to `32000`

Default: `10`

PDF417 uses this only when row height is not specified in the ^B7 h parameter.

Special Considerations for ^FD When Using PDF417

The character set sent to the printer with the ^FD command includes the full ASCII set, except for those characters with special meaning to the printer.

See Zebra Code Page 850 — Latin Character Set on page 1543, ^CC ~CC on page 152, and ^CT ~CT on page 166.

- CR and LF are also valid characters for all ^FD statements. This scheme is used:

`\&` = carriage return/line feed

`\\` = backslash (\)

- `^CI13` must be selected to print a backslash (\).

^B8

The ^B8 command is the shortened version of the EAN-13 barcode. EAN is an acronym for European Article Numbering. Each character in the EAN-8 barcode is composed of four elements: two bars and two spaces.

EAN-8 Barcode

- ^B8 supports a fixed ratio.
- Field data (^FD) is limited to exactly seven characters. ZPL II automatically pads or truncates on the left with zeros to achieve the required number of characters.
- When using JAN-8 (Japanese Article Numbering), a specialized application of EAN-8, the first two non-zero digits sent to the printer are always 49.

IMPORTANT: If additional information about this bar code is required, go to aimglobal.org.

Format: ^B8o,h,f,g

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** current ^FW value
h = bar code height (in dots)	**Values:** 1 to 32000 **Default:** value set by ^BY
f = print interpretation line	**Values:** N = no Y = yes **Default:** Y
g = print interpretation line above code	**Values:** N = no Y = yes **Default:** N

Example

This is an example of an EAN-8 barcode:

ZPL II CODE	EAN-8 BAR CODE
^XA ^FO100,100^BY3 ^B8N,100,Y,N ^FD1234567^FS ^XZ	

EAN-8 BAR CODE CHARACTERS
0 1 2 3 4 5 6 7 8 9

^B9

The ^B9 command produces a variation of the UPC symbology used for the number system 0. It is a shortened version of the UPC-A barcode, where zeros are suppressed, resulting in codes that require less printing space.

UPC-E Barcode

The 6 dot/mm, 12 dot/mm and 24 dot/mm printheads produce the UPC and EAN symbologies at 100 percent of their size. However, an 8 dot/mm printhead produces the UPC and EAN symbologies at a magnification factor of 77 percent.

Each character in a UPC-E barcode is composed of four elements: two bars and two spaces. The ^BY command must be used to specify the width of the narrow bar.

- ^B9 supports a fixed ratio.
- Field data (^FD) is limited to exactly 10 characters, requiring a five-digit manufacturer's code and a five-digit product code.
- When using the zero-suppressed versions of UPC, you must enter the full 10-character sequence. ZPL II calculates and prints the shortened version.

IMPORTANT: If additional information about this barcode is required, refer to aimglobal.org.

Format: ^B9,h,f,g,e

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** current ^FW value
h = barcode height (in dots)	**Values:** 1 to 32000 **Default:** value set by ^BY
f = print interpretation line	**Values:** N = no Y = yes **Default:** Y
g = print interpretation line above code	**Values:** N = no Y = yes **Default:** N
e = print check digit	**Values:** N = no Y = yes **Default:** Y

Example

This is an example of a UPC-E barcode:

ZPL II CODE	UPC-E BAR CODE
^XA ^FO150,100^BY3 ^B9N,100,Y,N,Y ^FD1230000045^FS ^XZ	

UPC-E BAR CODE CHARACTERS
0 1 2 3 4 5 6 7 8 9

Rules for Proper Product Code Numbers

- If the last three digits in the manufacturer's number are 000, 100, or 200, valid product code numbers are 00000 to 00999.
- If the last three digits in the manufacturer's number are 300, 400, 500, 600, 700, 800, or 900, valid product code numbers are 00000 to 00099.
- If the last two digits in the manufacturer's number are 10, 20, 30, 40, 50, 60, 70, 80, or 90, valid product code numbers are 00000 to 00009.
- If the manufacturer's number does not end in zero (0), valid product code numbers are 00005 to 00009.

^BA

The ^BA command creates a variable length, continuous symbology. The Code 93 barcode is used in many of the same applications as Code 39. It uses the full 128-character ASCII set. ZPL II, however, does not support ASCII control codes or escape sequences. It uses the substitute characters shown below.

Code 93 Barcode

Control Code	ZPL II Substitute
Ctrl $	&
Ctrl %	‘
Ctrl /	(
Ctrl +	)

Each character in the Code 93 barcode is composed of six elements: three bars and three spaces. Although invoked differently, the human-readable interpretation line prints as though the control code has been used.

- ^BA supports a fixed print ratio.
- Field data (^FD) is limited to the width (or length, if rotated) of the label.

IMPORTANT: If additional information about this bar code is required, refer to www.aimglobal.org.

Format: ^BAo,h,f,g,e

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** current ^FW value
h = bar code height (in dots)	**Values:** 1 to 32000 **Default:** value set by ^BY
f = print interpretation line	**Values:** N = no Y = yes **Default:** Y
g = print interpretation line above code	**Values:** N = no Y = yes **Default:** N

Parameters	Details
e = print check digit	**Values:** N = no Y = yes **Default:** N

Example

This is an example of a Code 93 barcode:

ZPL II CODE

```
^XA
^FO100,75^BY3
^BAN,100,Y,N,N
^FD12345ABCDE^FS
^XZ
```

CODE 93 BAR CODE

CODE 93 BAR CODE CHARACTERS

0 1 2 3 4 5 6 7 8 9
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
- . $ / + % & ’ ( )
SPACE

□ Denotes an internal start/stop character that must precede and follow every bar code message.

Comments: All control codes are used in pairs. Code 93 is also capable of encoding the full 128-character ASCII set.

Full ASCII Mode for Code 93

Code 93 can generate the full 128-character ASCII set using paired characters as shown in the following tables.

ASCII	Code 93
NUL	‘U
SOH	&A
STX	&B
ETX	&C
EOT	&D
ENQ	&E
ACK	&F
BEL	&G
BS	&H
HT	&I
LF	&J
VT	&K
FF	&L
CR	&M
SO	&N
SI	&O
DLE	&P
DC1	&Q
DC2	&R
DC3	&S
DC4	&T
NAK	&U
SYN	&V
ETB	&W
CAN	&X
EM	&Y
SUB	&Z
ESC	‘A
FS	‘B
FS	‘C
RS	‘D
US	‘E

ASCII	Code 93
SP	Space
!	(A
“	(B
#	(C
$	(D
%	(E
&	(F
‘	(G
(	(H
)	(I
*	(J
++	++
,	(L
-	-
.	.
/	/
0	O
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
:	(Z
;	‘F
<	‘G
=	‘H
>	‘I
?	‘J

^BB

The ^BB command produces a two-dimensional, multirow, stacked symbology. It is ideally suited for applications that require large amounts of information.

CODABLOCK Barcode

Depending on the mode selected, the code consists of one to 44 stacked rows. Each row begins and ends with a start and stop pattern.

- CODABLOCK A supports variable print ratios.
- CODABLOCK E and F support only fixed print ratios.

IMPORTANT: If additional information about this bar code is required, go towww.aimglobal.org.

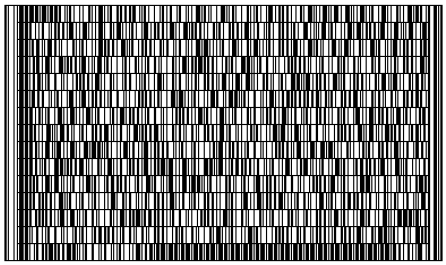
Format: ^BBo,h,s,c,r,m

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** N
h = bar code height for individual rows (in dots)	**Values:** 2 to 32000 **Default:** 8 This number, multiplied by the module, equals the height of the individual row in dots.
s = security level	**Values:** N = no Y = yes **Default:** Y Security level determines whether symbol check-sums are generated and added to the symbol. Checksums are never generated for single-row symbols. This can be turned off only if the parameter m is set to A.
c = number of characters per row (data columns)	**Values:** 2 to 62 characters This is used to encode a CODABLOCK symbol. It gives you control over the width of the symbol.

Parameters	Details
r = number of rows to encode-	**Values:** for CODABLOCK A: 1 to 22 for CODABLOCK E and F: 2 to 4 • If values for c and r are not specified, a single row is produced. • If a value for r is not specified, and c exceeds the maximum range, a single row equal to the field data length is produced. • If a value for c is not specified, the number of characters per row is derived by dividing the field data by the value of r. • If the s parameter is set to the default of Y, then the checksum characters that are included count as two data characters. For example, if c = 6, r is set to 3 and s is set to N, then up to 18 characters can be used (6 x 3). However, if s is set to Y, then only 16 characters can be used. • If the data field contains primarily numeric data, fewer than the specified rows might be printed. If the field data contains several shift and code-switch characters, more than the specified number of rows might be printed.
m = mode	**Values:** A, E, F CODABLOCK A uses the Code 39 character set. CODABLOCK F uses the Code 128 character set. CODABLOCK E uses the Code 128 character set and automatically adds FNC1. **Default:** F

Example

This is an example of a CODABLOCK barcode:

ZPL II CODE	CODABLOCK BAR CODE
^XA ^BY2,3 ^FO10,10^BBN,30,,30,44,E ^FDZebra Technologies Corporation strives to be the expert supplier of innovative solutions to speciality demand labeling and ticketing problems of business and government. We will attract and retain the best people who will understand our customer's needs and provide them with systems, hardware, software, consumables and service offering the best value, high quality, and reliable performance, all delivered in a timely manner.^FS ^XZ	

Special Considerations for the ^BY Command When Using ^BB

The parameters for the ^BY`w,r,h` command, when used with a ^BB code, are as follows:

`w` = **module width (in dots)**

Values: 2 to `10` (CODABLOCK A only)

Default: 2

`r` = **ratio**

Fixed Value: 3 (ratio has no effect on CODABLOCK E or F)

`h` = **height of bars (in dots)**

Values: `1` to `32,32000`

Default: `10`

CODABLOCK uses this as the overall symbol height only when the row height is not specified in the ^BB `h` parameter.

Special Considerations for ^FD Character Set When Using ^BB

The character set sent to the printer depends on the mode selected in parameter m.

CODABLOCK A: CODABLOCK A uses the same character set as Code 39. If any other character is used in the ^FD statement, either no barcode is printed or an error message is printed (if ^CV is active).

CODABLOCK E:

The Automatic Mode includes the full ASCII set except for those characters with special meaning to the printer. Function codes or the Code 128 Subset A <`nul`> character can be inserted using the ^FH command.

<fnc1> = 80 hex	<fnc3> = 82 hex

<fnc2> = 81 hex	<fnc4> = 83 hex
<nul> = 84 hex	

For any other character above 84 hex, either no barcode is printed or an error message is printed (if ^CV is active).

CODABLOCK F: CODABLOCK F uses the full ASCII set, except for those characters with special meaning to the printer. Function codes or the Code 128 Subset A <nul> character can be inserted using the ^FH command.

<fnc1> = 80 hex	<fnc3> = 82 hex
<fnc2> = 81 hex	<fnc4> = 83 hex
<nul> = 84 hex	

^BC

The ^BC command creates the Code 128 barcode, a high-density, variable length, continuous, alphanumeric symbology. It was designed for complexly encoded product identification.

Code 128 Barcode (Subsets A, B, and C)

Code 128 has three subsets of characters. There are 106 encoded printing characters in each set, and each character can have up to three different meanings, depending on the character subset being used. Each Code 128 character consists of six elements: three bars and three spaces.

- ^BC supports a fixed print ratio.
- Field data (^FD) is limited to the width (or length, if rotated) of the label.

IMPORTANT: If additional information about this barcode is required, refer to aimglobal.org.

Format: ^BCo,h,f,g,e,m

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** current ^FW value
h = barcode height (in dots)	**Values:** 1 to 32000 **Default:** value set by ^BY
f = print interpretation line	**Values:** Y (yes) or N (no) **Default:** Y The interpretation line can be printed in any font by placing the font command before the barcode command.
g = print interpretation line above code	**Values:** Y (yes) or N (no) **Default:** N
e = UCC check digit	**Values:** Y (turns on) or N (turns off) Mod 103 check digit is always there. It cannot be turned on or off. Mod 10 and 103 appear together with e turned on. **Default:** N

Parameters	Details
m = mode	**Values:** N = no selected mode U = UCC Case Mode • More than 19 digits in ^FD or ^SN are eliminated. • Fewer than 19 digits in ^FD or ^SN add zeros to the right to bring the count to 19. This produces an invalid interpretation line. A = Automatic Mode This analyzes the data sent and automatically determines the best packing method. The full ASCII character set can be used in the ^FD statement — the printer determines when to shift subsets. A string of four or more numeric digits causes an automatic shift to Subset C. D = UCC/EAN Mode (x.11.x and newer firmware) This allows dealing with UCC/EAN with and without chained application identifiers. The code starts in the appropriate subset followed by FNC1 to indicate a UCC/EAN 128 barcode. The printer automatically strips out parentheses and spaces for encoding but prints them in the human-readable section. The printer automatically determines if a check digit is required, calculates it, and prints it. Automatically sizes the human readable. **Default:** N

Example: This is an example of a Code 128 barcode:

Figure 2 Code 128 Barcode

ZPL II CODE	CODE 128 BAR CODE
^XA ^FO100,100^BY3 ^BCN,100,Y,N,N ^FD123456^FS ^XZ	 123456

Code 128 Subsets

The Code 128 character subsets are referred to as Subset A, Subset B, and Subset C. A subset can be selected in these ways:

- A special Invocation Code can be included in the field data (^FD) string associated with that barcode.
- The desired Start Code can be placed at the beginning of the field data. If no Start Code is entered, Subset B is used.

To change subsets within a barcode, place the Invocation Code at the appropriate points within the field data (^FD) string. The new subset stays in effect until changed with the Invocation Code. For example, in Subset C, >7 in the field data changes the Subset to A.

The following table shows the Code 128 Invocation Codes and Start Characters for the three subsets.

Table 2 Code 128 Invocation Characters

Invocation Code	Decimal Value	Subset A Character	Subset B Character	Subset C Character
><	62			
>0	30	>	>	
>=	94		~	
>1	95	USQ	DEL	
>2	96	FNC 3	FNC 3	
>3	97	FNC 2	FNC 2	
>4	98	SHIFT	SHIFT	
>5	99	CODE C	CODE C	
>6	100	CODE B	FNC 4	CODE B
>7	101	FNC 4	CODE A	CODE A
>8	102	FNC 1	FNC 1	FNC 1
Start Characters				
>9	103	Start CodeA	(Numeric Pairs give Alpha/ Numerics)	
>:	104	Start CodeB	(Normal Alpha/ Numeric)	
>;	105	Stat CodeC	(All Numeric(00 - 99)	

The following table shows the character sets for Code 128:

Table 3 Code 128 Character Sets

Value	CodeA	CodeB	CodeC		Value	CodeA	CodeB	CodeC
0	SP	SP	00		53	U	U	53
1	!	!	01		54	V	V	54
2	"	"	02		55	W	W	55
3	#	#	03		56	X	X	56
4	$	$	04		57	Y	Y	57
5	%	%	05		58	Z	Z	58
6	&	&	06		59	[	[	59
7	'	'	07		60	\	\	60
8	(	(	08		61	]	]	61
9	)	)	09		62	^	^	62
10	*	*	10		63	_	_	63

Table 3 Code 128 Character Sets (Continued)

Value	CodeA	CodeB	CodeC		Value	CodeA	CodeB	CodeC
11	+	+	11		64	NUL	.	64
12	,	,	12		65	SOH	a	65
13	-	-	13		66	STX	b	66
14	.	.	14		67	ETX	c	67
15	/	/	15		68	EOT	d	68
16	0	0	16		69	ENQ	e	69
17	1	1	17		70	ACK	f	70
18	2	2	18		71	BEL	g	71
19	3	3	19		72	BS	h	72
20	4	4	20		73	HT	i	73
21	5	5	21		74	LF	j	74
22	6	6	22		75	VT	k	75
23	7	7	23		76	FF	l	76
24	8	8	24		77	CR	m	77
25	9	9	25		78	SO	n	78
26	:	:	26		79	SI	o	79
27	;	;	27		80	DLE	p	80
28	<	<	28		81	DC1	q	81
29	=	=	29		82	DC2	r	82
30	>	>	30		83	DC3	s	83
31	?	?	31		84	DC4	t	84
32	@	@	32		85	NAK	u	85
33	A	A	33		86	SYN	v	86
34	B	B	34		87	ETB	w	87
35	C	C	35		88	CAN	x	88
36	D	D	36		89	EM	y	89
37	E	E	37		90	SUB	z	90
38	F	F	38		91	ESC	{	91
39	G	G	39		92	FS	|	92
40	H	H	40		93	GS	}	93
41	I	I	41		94	RS	~	94
42	J	J	42		95	US	DEL	95
43	K	K	43		96	FNC3	FNC3	96

Table 3 Code 128 Character Sets (Continued)

Value	CodeA	CodeB	CodeC		Value	CodeA	CodeB	CodeC
44	L	L	44		97	FNC2	FNC2	97
45	M	M	45		98	SHIFT	SHIFT	98
46	N	N	46		99	CodeC	CodeC	99
47	O	O	47		100	CodeB	FNC4	CodeB
48	P	P	48		101	FNC4	CodeA	CodeA
49	Q	Q	49		102	FNC1	FNC1	FNC1
50	R	R	50		103		START (CodeA)	
51	S	S	51		104		START (CodeB)	
52	T	T	52		105		START (CodeC)	

Example: The following figures are examples of identical barcodes.

Figure 3 Subset B with no Start Character

```
^XA
^FO100,75
^BCN,100,Y,N,N
^FDCODE128^FS
^XZ
```

Figure 4 Subset B with Start Character

```
^XA
^FO100,75
^BCN,100,Y,N,N
^FD>:CODE128^FS
^XZ
```

Example: Because Code 128 Subset B is the most commonly used subset, ZPL II defaults to Subset B if no start character is specified in the data string.

This figure is an example of switching from Subset C to B to A.

Figure 5 Switching from Subset C to B to A

```
^XA
^FO50,50
^BY3^BCN,100,Y,N,N
^FD>;382436>6CODE128>752375152^FS
^XZ
```

How ^BC Works Within a ZPL II Script

^XA – the first command starts the label format.

^FO100,75 – the second command sets the field origin at 100 dots across the x-axis and 75 dots down the y-axis from the upper-left corner.

^BCN,100,Y,N,N – the third command calls for a Code 128 barcode to be printed with no rotation (N) and a height of 100 dots. An interpretation line is printed (Y) below the barcode (N). No UCC check digit is used (N).

^FDCODE128^FS (Figure A) ^FD>:CODE128^FS (Figure B) – the field data command specifies the content of the barcode.

^XZ – the last command ends the field data and indicates the end of the label.

The interpretation line prints below the code with the UCC check digit turned off.

The ^FD command for Figure A does not specify any subset, so Subset B is used. In Figure B, the ^FD command specifically calls Subset B with the >: Start Code. Although ZPL II defaults to Code B, it is good practice to include the Invocation Codes in the command.

Code 128 – Subset B is programmed directly as ASCII text, except for values greater than 94 decimal and a few special characters that must be programmed using the invocation codes. Those characters are:

```
^ > ~
```

Code 128 – Subsets A and C

Code 128, Subsets A and C are programmed in pairs of digits, 00 to 99, in the field data string. For details, see Table 2 Code 128 Invocation Characters on page 96.

In Subset A, each pair of digits results in a single character being encoded in the barcode; in Subset C, characters are printed as entered. Figure E below is an example of Subset A (>9 is the Start Code for Subset A).

Nonintegers programmed as the first character of a digit pair (D2) are ignored. However, nonintegers programmed as the second character of a digit pair (2D) invalidate the entire digit pair, and the pair is ignored. An extra unpaired digit in the field data string just before a code shift is also ignored.

The figures below are examples of Subset C. Notice that the barcodes are identical.

Figure 6 Subset C with Normal Data

```
^XA
^FO100,75^BY3
^BCN,100,Y,N,N
^FD>;382436^FS
^XZ
```

Figure 7 Subset C with Ignored Alpha Character

```
^XA
^FO100,75^BY3
^BCN,100,Y,N,N
^FD>;38D2436^FS
^XZ
```

In the program code for the following figure, the D is ignored, and the 2 is paired with the 4.

Figure 8 Subset A

```
^XA
^FO100,75^BY3
^BCN,100,Y,N,N
^FD>935473637171824^FS
^XZ
```

The UCC/EAN-128 Symbology

The symbology specified for the representation of Application Identifier data is UCC/EAN-128, a variant of Code 128, exclusively reserved for EAN International and the Uniform Code Council (UCC).

NOTE: It is not intended to be used for data to be scanned at the point of sale in retail outlets.

UCC/EAN-128 offers several advantages. It is one of the most complete, alphanumeric, one-dimensional symbologies available today. The use of three different character sets (A, B, and C), facilitates the encoding of the full 128 ASCII character set. Code 128 is one of the most compact linear barcode symbologies. Character set C enables numeric data to be represented in a double-density mode. In this mode, two digits are represented by only one symbol character saving valuable space. The code is concatenated. That means that multiple AIs and their fields may be combined into a single barcode. The code is also very reliable. Code 128 symbols use two independent self-checking features which improves printing and scanning reliability.

UCC/EAN-128 barcodes always contain a special non-data character known as function 1 (FNC 1), which follows the start character of the barcode. It enables scanners and processing software to auto-discriminate between UCC/EAN-128 and other barcode symbologies, and subsequently only process relevant data.

The UCC/EAN-128 barcode is made up of a leading quiet zone, a Code 128 start character A, B, or C, an FNC 1 character, Data (Application Identifier plus data field), a symbol check character, a stop character, and a trailing quiet zone.

UCC/EAN and UCC/128 are a couple of ways you'll hear someone refer to the code. This just indicates that the code is structured as dictated by the application identifiers that are used.

SSCC (Serial Shipping Container Code) formatted following the data structure layout for Application Identifier 00. See Table 4 UCC/EAN Application Identifier on page 103. It could be 00 which is the SSCC code. The customer needs to let us know what application identifiers are used for their barcode so we can help them.

There are several ways of writing the code to print the code to the Application Identifier '00' structure.

Using N for the Mode (m) Parameter

This example shows with application identifier 00 structure:

Figure 9 N for the M Parameter

ZPL II CODE	N FOR THE M PARAMETER
^XA ^FO90,200^BY4 ^BCN,256,Y,N,Y,N ^FD>;>8001234512345123451234512^FS ^XZ	

- >;>8' sets it to subset C, function 1
- '00' is the application identifier followed by '17 characters', the check digit is selected using the 'Y' for the (e) parameter to automatically print the 20th character.
- you are not limited to 19 characters with the mode set to N

Using U for the Mode (m) Parameter

The example shows the application identifier 00 format:

Figure 10 U for the M Parameter

ZPL II CODE	U FOR THE M PARAMETER
^XA ^FO90,200 ^BY4^BC,256,Y,N,,U ^FD001234512345123451234512^FS ^XZ	

UCC Case Mode

- Choosing U selects UCC Case mode. You will have exactly 19 characters available in ^FD.
- Subset C using FNC1 values are automatically selected.
- Check digit is automatically inserted.

Using D for the Mode (m) Parameter

This example shows application identifier 00 format ((x.11.x or later):

Figure 11 D for the M Parameter

ZPL II CODE	D FOR THE M PARAMETER
^XA ^FO50,200^BCN,150,Y,N,,D ^FD(00)10084423 7449200940^FS ^XZ	

(0 at the end of field data is a bogus character that is inserted as a placeholder for the check digit the printer will automatically insert.

- Subset C using FNC1 values are automatically selected.
- Parentheses and spaces can be in the field data. '00' application identifier, followed by 17 characters, followed by a bogus check digit placeholder.
- Check digit is automatically inserted. The printer will automatically calculate the check digit and put it into the barcode and interpretation line.
- The interpretation line will also show the parentheses and spaces but will strip them out from the actual barcode.

Printing the Interpretation Line

This example shows printing the interpretation in a different font with firmware x.11.x or later:

Figure 12 Interpretation Line

ZPL II CODE	INTERPRET ATION LINE
^XA ^FO50,200 ^A0N,40,30^BCN,150,Y,N,Y ^FD>;>8001234512345123451234512^FS ^XZ	

The font command (^A0N,40,30) can be added and changed to alter the font and size of the interpretation line.

With firmware version later than x.10.x

- A separate text field needs to be written.
- The interpretation line needs to be turned off.
- ^A0N,50,40 is the font and size selection for the separate text field.
- You have to make sure you enter the correct check digit in the text field.
- Creating a separate text field allows you to format the interpretation line with parentheses and spaces.

Figure 13 Firmware Older Than X.10.X

ZPL II CODE	FIRMWARE OLDER THAN X.10.X
^XA ^FO25,25 ^BCN,150,N,N,Y ^FD>;>8001234512345123451234512^FS ^FO100,190 ^A0N,50,40 ^FD(00) 1 2345123 451234512 0^FS ^XZ	

Application Identifiers — UCC/EAN APPLICATION IDENTIFIER

An Application Identifier is a prefix code used to identify the meaning and the format of the data that follows it (data field).

There are AIs for identification, traceability, dates, quantity, measurements, locations, and many other types of information.

For example, the AI for batch number is 10, and the batch number AI is always followed by an alphanumeric batch code not to exceed 20 characters.

The UCC/EAN Application Identifiers provide an open standard that can be used and understood by all companies in the trading chain, regardless of the company that originally issued the codes.

NOTE: Table 4 UCC/EAN Application Identifier on page 103 is a partial table showing the application identifiers. For more current and complete information, search the Internet for **UCC Application Identifier**.

Table 4 UCC/EAN Application Identifier

Data Content	AI	Plus The Following Data Structure
Serial Shipping Container Code (SSCC)	00	exactly 18 digits
Shipping Container Code	01	exactly 14 digits
Batch Numbers	10	up to 20 alphanumerics
Production Date (YYMMDD)	11	exactly 6 digits
Packaging Date (YYMMDD)	13	exactly 6 digits
Sell By Date (YYMMDD)	15	exactly 6 digits
Expiration Date (YYMMDD)	17	exactly 6 digits
Product Variant	20	exactly 2 digits
Serial Number	21	up to 20 alphanumerics
HIBCC Quantity, Date, Batch and Link	22	up to 29 alphanumerics
Lot Number	23	up to 19 alphanumerics
Quantity Each	30	
Net Weight (Kilograms)	310	exactly 6 digits
Length, Meters	311	exactly 6 digits
Width or Diameter (Meters)	312	exactly 6 digits
Depths (Meters)	313	exactly 6 digits
Area (Sq. Meters)	314	exactly 6 digits
Volume (Liters)	315	exactly 6 digits
Volume (Cubic Meters)	316	exactly 6 digits
Net Weight (Pounds)	320	exactly 6 digits
Customer PO Number	400	up to 29 alphanumerics
Ship To (Deliver To) Location Code using EAN 13 or DUNS Number with leading zeros	410	exactly 13 digits

Table 4 UCC/EAN Application Identifier (Continued)

Data Content	AI	Plus The Following Data Structure
Bill To (Invoice To) Location Code using EAN 13 or DUNS Number with leading zeros	411	exactly 13 digits
Purchase from	412	exactly 13 digits
Ship To (Deliver To) Postal Code within single postal authority	420	up to 9 alphanumerics
Ship To (Deliver To) Postal Code with 3-digit ISO Country Code Prefix	421	3 digits plus up to 9 alphanumerics
Roll Products - width, length, core diameter, direction and splices	8001	exactly 14 digits
Electronic Serial number for cellular mobile phone	8002	up to 20 alphanumerics
Plus one digit for length indication. Plus one digit for decimal point indication.		

For date fields that only need to indicate a year and month, the day field is set to `00`.

Chaining several application identifiers (firmware x.11.x or later)

The FNC1, which is invoked by >8, is inserted just before the AIs so that the scanners reading the code see the FNC1 and know that an AI follows.

Example: This is an example with the mode parameter set to A (automatic):

```
^XA
^BY2,2.5,193
^FO33,400
^BCN,,N,N,N,A
^FD>;>8020401777300348610000 8535>8910001>837252^FS
^FT33,625^AEN,0,0^FD(02)04017773003486(10)0008535(91)0001(37)252^FS
^XZ
```

Example: This is an example with the mode parameter set to U:

```
^XA
^BY3,2.5,193
^FO33,200
^BCN,,N,N,N,U
^FD>;>80204017773003486>8100008535>8910001>837252^FS
^FT33,455^A0N,30,30^FD(02)04017773003486(10)0008535(91)0001(37)252^FS
^XZ
```

Example: This is an example with the mode parameter set to D*:

```
^XA
^PON
^LH0,0
```

```
^BY2,2.5,145
^FO218,343
^BCB,,Y,N,N,D
^FD(91)0005886>8(10)0000410549>8(99)05^FS
^XZ
```

^BD

The ^BD command creates a two-dimensional, optically read (not scanned) code. This symbology was developed by UPS (United Parcel Service).

UPS MaxiCode Bar Code

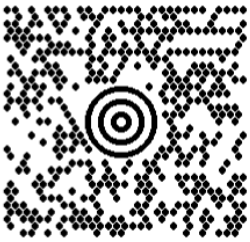
Notice that there are no additional parameters for this code and it does not generate an interpretation line. The ^BY command has no effect on the UPS MaxiCode barcode. However, the ^CV command can be activated.

Format: ^BDm,n,t

Parameters	Details
m = mode	**Values:** 2 = structured carrier message: numeric postal code (U.S.) 3 = structured carrier message: alphanumeric postal code (non-U.S.) 4 = standard symbol, secretary 5 = full EEC 6 = reader program, secretary **Default:** 2
n = symbol number	**Values:** 1 to 8 can be added in a structured document **Default:** 1
t = total number of symbols	**Values:** 1 to 8, representing the total number of symbols in this sequence **Default:** 1

Example

This is an example of the UPS MAXICODE - MODE 2 barcode:

ZPL II CODE	UPS MAXICODE - MODE 2
^XA ^FO50,50 ^CVY ^BD^FH^FD001840152382802 [)>_1E01_1D961Z00004951_1DUPSN_ 1D_06X610_1D159_1D1234567_1D1/1_ 1D_1DY_1D634 ALPHA DR_ 1DPITTSBURGH_1DPA_1E_04^FS ^FO30,300^A0,30,30^FDMode2^FS ^XZ	

Special Considerations for ^FD when Using ^BD

The ^FD statement is divided into two parts: a high-priority message (hpm) and a low-priority message (lpm). There are two types of high-priority messages. One is for a US Style Postal Code; the other is for a non-U.S. Style Postal Code. The syntax for either of these high-priority messages must be exactly as shown or an error message is generated.

Format: ^FD <hpm><lpm>

Parameters	Details
<hpm> = high-priority message (applicable only in Modes 2 and 3)	**Values:** 0 to 9, except where noted U.S. Style Postal Code (Mode 2) <hpm> = aaabbbccccddddd aaa = three-digit class of service bbb = three-digit country zip code ccccc = five-digit zip code dddd = four-digit zip code extension (if none exists, four zeros (0000) must be entered) non-U.S. Style Postal Code (Mode 3) <hpm> = aaabbbcccccc aaa = three-digit class of service bbb = three-digit country zip code ccccc = six-digit zip code (A through Z or 0 to 9)
<lpm> = low priority message (only applicable in Modes 2 and 3)	GS is used to separate fields in a message (0x1D). RS is used to separate format types (0x1E). EOT is the end of transmission characters. **Message Header** []>RS **Transportation Data** **Format Header** 01GS96 **Tracking Number*** <tracking number> **SCAC***GS<SCAC> **UPS Shipper Number** GS<shipper number> **Julian Day of Pickup** GS<day of pickup> **Shipment ID Number** GS<shipment ID number> **Package n/x** GS<n/x> **Package Weight** GS<weight> **Address Validation** GS<validation> **Ship to Street Address** GS<street address> **Ship to City** GS<city> **Ship to State** GS<state> **RS** RS **End of Message** EOT (* Mandatory Data for UPS)

Comments:

- The formatting of <hpm> and <lpm> apply only when using Modes 2 and 3. Mode 4, for example, takes whatever data is defined in the ^FD command and places it in the symbol.
- UPS requires that certain data be present in a defined manner. When formatting MaxiCode data for UPS, always use uppercase characters. When filling in the **fields** in the <lpm> for UPS, follow the data size and types specified in **Guide to Bar Coding with UPS**.

- If you do not choose a mode, the default is Mode 2. If you use non-U.S. Postal Codes, you probably get an error message (invalid character or message too short). When using non-U.S. codes, use Mode 3.
- ZPL II doesn't automatically change your mode based on the zip code format.
- When using special characters, such as GS, RS, or EOT, use the ^FH command to tell ZPL II to use the hexadecimal value following the underscore character (_).

^BE

The ^BE command is similar to the UPC-A barcode. It is widely used throughout Europe and Japan in the retail marketplace.

EAN-13 Barcode

The EAN-13 barcode has 12 data characters, one more data character than the UPC-A code. An EAN-13 symbol contains the same number of bars as the UPC-A but encodes a 13th digit into a parity pattern of the left-hand six digits. This 13th digit, in combination with the 12th digit, represents a country code.

- ^BE supports fixed print ratios.
- Field data (^FD) is limited to exactly 12 characters. ZPL II automatically truncates or pads on the left with zeros to achieve the required number of characters.
- When using JAN-13 (Japanese Article Numbering), a specialized application of EAN-13, the first two non-zero digits sent to the printer must be 49.

NOTE: Use Interleaved 2 of 5 for UCC and EAN 14.

IMPORTANT: If additional information about this barcode is required, refer to www.aimglobal.org.

Format: ^BEo,h,f,g

Parameters	Details
o = orientation	**Value:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** current ^FW value
h = barcode height (in dots)	**Value:** 1 to 32000 **Default:** value set by ^BY
f = print interpretation line	**Value:** N = no Y = yes **Default:** Y
g = print interpretation line above code	**Values:** N = no Y = yes **Default:** N

Example

This is an example of an EAN-13 barcode:

ZPL II CODE	EAN-13 BAR CODE
^XA ^FO100,100^BY3 ^BEN,100,Y,N ^FD12345678^FS ^XZ	

EAN-13 BAR CODE CHARACTERS
0 1 2 3 4 5 6 7 8 9

Comments: The EAN-13 barcode uses the Mod 10 check-digit scheme for error checking. For more information on Mod 10, see Mod 10 Check Digit on page 1568.

^BF

The ^BF command creates a two-dimensional, multi-row, continuous, stacked symbology identical to PDF417, except it replaces the 17-module-wide start and stop patterns and left/right row indicators with a unique set of 10-module-wide row address patterns. These reduce overall symbol width and allow linear scanning at row heights as low as 2X.

MicroPDF417 Barcode

MicroPDF417 is designed for applications with a need for improved area efficiency but without the requirement for PDF417's maximum data capacity. It can be printed only in specific combinations of rows and columns up to a maximum of four data columns by 44 rows.

Field data (^FD) and field hexadecimal (^FH) are limited to:

- 250 7-bit characters
- 150 8-bit characters
- 366 4-bit numeric characters

Format: ^BFo,h,m

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** current ^FW value
h = bar code height (in dots)	**Values:** 1 to 9999 **Default:** value set by ^BY or 10 (if no ^BY value exists).
m = mode	**Values:** 0 to 33 (see Table 5 MicroPDF417 Mode on page 112) **Default:** 0 (see Table 5 MicroPDF417 Mode on page 112)

Example: This is an example of a MicroPDF417 barcode:

ZPL II CODE	MICRO-PDF417 BAR CODE
^XA ^FO100,100^BY6 ^BFN,8,3 ^FDABCDEFGHIJKLMNOPQRSTUV^FS ^XZ	

To encode data into a MicroPDF417 barcode, complete these steps:

1. Determine the type of data to be encoded (for example, ASCII characters, numbers, 8-bit data, or a combination).
2. Determine the maximum amount of data to be encoded within the barcode (for example, number of ASCII characters, quantity of numbers, or quantity of 8-bit data characters).

3. Determine the percentage of check digits that are used within the barcode. The higher the percentage of check digits that are used, the more resistant the barcode is to damage — however, the size of the barcode increases.
4. Use the following table with the information gathered from the questions above to select the mode of the barcode.

Table 5 MicroPDF417 Mode

Mode (M)	Number of Data Columns	Number of Data Rows	% of CWS for EC	Max Alpha Characters	Max Digits
0	1	11	64	6	8
1	1	14	50	12	17
2	1	17	41	18	26
3	1	20	40	22	32
4	1	24	33	30	44
5	1	28	29	38	55
6	2	8	50	14	20
7	2	11	41	24	35
8	2	14	32	36	52
9	2	17	29	46	67
10	2	20	28	56	82
11	2	23	28	64	93
12	2	26	29	72	105
13	3	6	67	10	14
14	3	8	58	18	26
15	3	10	53	26	38
16	3	12	50	34	49
17	3	15	47	46	67
18	3	20	43	66	96
19	3	26	41	90	132
20	3	32	40	114	167
21	3	38	39	138	202
22	3	44	38	162	237
23	4	6	50	22	32
24	4	8	44	34	49
25	4	10	40	46	67
26	4	12	38	58	85
27	4	15	35	76	111
28	4	20	33	106	155

Table 5 MicroPDF417 Mode (Continued)

Mode (M)	Number of Data Columns	Number of Data Rows	% of CWS for EC	Max Alpha Characters	Max Digits
29	4	26	31	142	208
30	4	32	30	178	261
31	4	38	29	214	313
32	4	44	28	250	366
33	4	4	50	14	20

^BI

The ^BI command is a discrete, self-checking, continuous numeric symbology. The Industrial 2 of 5 barcode has been in use the longest of the 2 of 5 family of barcodes. Of that family, the Standard 2 of 5 (^BJ) and Interleaved 2 of 5 (^B2) barcodes are also available in ZPL II.

Industrial 2 of 5 Bar Codes

With Industrial 2 of 5, all of the information is contained in the bars. Two bar widths are employed in this code, the wide bar measuring three times the width of the narrow bar.

- ^BI supports a print ratio of 2.0:1 to 3.0:1.
- Field data (^FD) is limited to the width (or length, if rotated) of the label.

IMPORTANT: If additional information about this barcode is required, refer to www.aimglobal.org.

Format: ^BIo,h,f,g

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** current ^FW value
h = bar code height (in dots)	**Values:** 1 to 32000 **Default:** value set by ^BY
f = print interpretation line	**Values:** N = no Y = yes **Default:** Y
g = print interpretation line above code	**Values:** N = no Y = yes **Default:** N

Example

This is an example of an Industrial 2 of 5 barcode:

ZPL II CODE	CODE 11 BARCODE
^XA ^FO100,100^BY3 ^B1N,N,150,Y,N ^FD123456^FS ^XZ	

CODE 11 BARCODE CHARACTERS

0 1 2 3 4 5 6 7 8 9 -

Internal Start/Stop Character: △

When used as a stop character:

△ is used with 1 check digit

△ is used with 2 check digits

^BJ

The ^BJ command is a discrete, self-checking, continuous numeric symbology.

Standard 2 of 5 Bar Code

With Standard 2 of 5, all of the information is contained in the bars. Two bar widths are employed in this code, the wide bar measuring three times the width of the narrow bar.

- ^BJ supports a print ratio of 2.0:1 to 3.0:1.
- Field data (^FD) is limited to the width (or length, if rotated) of the label.

IMPORTANT: If additional information about this bar code is required, refer to www.aimglobal.org.

Format: ^BJo,h,f,g

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** current ^FW value
h = Barcode height (in dots)	**Values:** 1 to 32000 **Default:** value set by ^BY
f = print interpretation line	**Values:** N = no Y = yes **Default:** Y
g = print interpretation line above code	**Values:** N = no Y = yes **Default:** N

Example

This is an example of a Standard 2 of 5 barcode:

ZPL II CODE	STANDARD 2 OF 5 BAR CODE
^XA ^FO100,100^BY3 ^BJN,150,Y,N ^FD123456^FS ^XZ	 123456

STANDARD 2 OF 5 BAR CODE CHARACTERS
0 1 2 3 4 5 6 7 8 9 **Start/Stop (automatic)**

^BK

The ANSI Codabar barcode is used in a variety of information processing applications such as libraries, the medical industry, and overnight package delivery companies. This barcode is also known as USD-4 code, NW-7, and 2 of 7 code. It was originally developed for retail price labeling.

ANSI Codabar Bar Code

Each character in this code is composed of seven elements: four bars and three spaces. Codabar bar codes use two character sets, numeric and control (start and stop) characters.

- ^BK supports a print ratio of 2.0:1 to 3.0:1.
- Field data (^FD) is limited to the width (or length, if rotated) of the label.

IMPORTANT: If additional information about this barcode is required, refer to www.aimglobal.org.

Format: ^BKo,e,h,f,g,k,l

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** current ^FW value
e = check digit	**Fixed Value:** N
h = Barcode height (in dots)	**Values:** 1 to 32000 **Default:** value set by ^BY
f = print interpretation line	**Values:** N = no Y = yes **Default:** Y
g = print interpretation line above code	**Values:** Y = no Y = yes **Default:** N
k = designates a start character	**Values:** A, B, C, D **Default:** A

Parameters	Details
l = designates stop character	**Values:** A, B, C, D **Default:** A

Example

This is an example of an ANSI Codabar barcode:

ZPL II CODE	ANSI CODABAR BAR CODE
^XA ^FO100,100^BY3 ^BKN,N,150,Y,N,A,A ^FD123456^FS ^XZ	

ANSI CODABAR BAR CODE CHARACTERS

0 1 2 3 4 5 6 7 8 9

Control Characters

- : . $ / +

Start/Stop Characters

A B C D

^BL

The ^BL command is a special application of Code 39 used by the Department of Defense. LOGMARS is an acronym for Logistics Applications of Automated Marking and Reading Symbols.

LOGMARS Barcode

- ^BL supports a print ratio of 2.0:1 to 3.0:1.
- Field data (^FD) is limited to the width (or length, if rotated) of the label. Lowercase letters in the ^FD string are converted to the supported uppercase LOGMARS characters.

IMPORTANT: If additional information about this barcode is required, refer to www.aimglobal.org.

Format: ^BLo,h,g

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** current ^FW value
h = Barcode height (in dots)	**Values:** 1 to 32000 **Default:** value set by ^BY
g = print interpretation line above code	**Values:** N = no Y = yes **Default:** N

Example

This is an example of a LOGMARS barcode:

ZPL II CODE	LOGMARS BAR CODE
^XA ^FO100,75^BY3 ^BLN,100,N ^FD12AB^FS ^XZ	

LOGMARS BAR CODE CHARACTERS
0 1 2 3 4 5 6 7 8 9
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
- . $ / + %
SPACE

Comments: The LOGMARS barcode produces a mandatory check digit using Mod 43 calculations. For further information on the Mod 43 check digit, go toMod 10 Check Digit on page 1568.

^BM

The ^BM command is a pulse-width modulated, continuous, non-self-checking symbology. It is a variant of the Plessey barcode (^BP).

MSI Bar Code

Each character in the MSI barcode is composed of eight elements: four bars and four adjacent spaces.

- ^BM supports a print ratio of 2.0:1 to 3.0:1.
- For the barcode to be valid, field data (^FD) is limited to 1 to 14 digits when parameter e is B, C, or D. ^FD is limited to 1 to 13 digits when the parameter e is A, plus a quiet zone.

IMPORTANT: If additional information about this bar code is required, refer to www.aimglobal.org.

Format: ^BMo,e,h,f,g,e2

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** current ^FW value
e = check digit selection	**Values:** A = no check digits B = 1 Mod 10 C = 2 Mod 10 D = 1 Mod 11 and 1 Mod 10 **Default:** B
h = Barcode height (in dots)	**Values:** 1 to 32000 **Default:** value set by ^BY
f = print interpretation line	**Values:** N = no Y = yes **Default:** Y
g = print interpretation line above code	**Values:** N = no Y = yes **Default:** N

Parameters	Details
e2 = inserts check digit into the interpretation line	**Values:** N = no Y = yes **Default:** N

Example

This is an example of an MSI barcode:

ZPL II CODE	MSI BAR CODE
^XA ^FO100,100^BY3 ^BMN,B,100,Y,N,N ^FD123456^FS ^XZ	

MSI BAR CODE CHARACTERS
1 2 3 4 5 6 7 8 9

^BO

The ^B0 command creates a two-dimensional matrix symbology made up of square modules arranged around a bulls-eye pattern at the center.

Aztec Bar Code Parameters

NOTE: The Aztec barcode works with firmware version V60.13.0.11A and V50.13.2 or later.

Format: ^B0a,b,c,d,e,f,g

Parameters	Details
a = orientation	**Values:** N = normal R = rotated I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** current ^FW value
b = magnification factor	**Values:** 1 to 10 **Default:** 1 on 150 dpi printers 2 on 200 dpi printers 3 on 300 dpi printers 6 on 600 dpi printers
c = extended channel interpretation code indicator	**Values:** Y = if data contains ECICs N = if data does not contain ECICs. **Default:** N
d = error control and symbol size/type indicator	**Values:** 0 = default error correction level 01 to 99 = error correction percentage (minimum) 101 to 104 = 1 to 4-layer compact symbol 201 to 232 = 1 to 32-layer full-range symbol 300 = a simple Aztec “Rune” **Default:** 0
e = menu symbol indicator	**Values:** Y = if this symbol is to be a menu (barcode reader initialization) symbol N = if it is not a menu symbol **Default:** N
f = number of symbols for structured append	**Values:** 1 through 26 **Default:** 1

Parameters	Details
g = optional ID field for structured append	The ID field is a text string with a 24-character maximum**Default:** no ID

Example

This is an example of the ^B0 command:

ZPL II CODE	GENERA TED LABEL
^XA ^B0R,7,N,0,N,1,0 ^FD 7. This is testing label 7^FS ^XZ	

^BP

The ^BP command is a pulse-width modulated, continuous, non-self-checking symbology.

Plessey Barcode

Each character in the Plessey barcode is composed of eight elements: four bars and four adjacent spaces.

- ^BP supports a print ratio of 2.0:1 to 3.0:1.
- Field data (^FD) is limited to the width (or length, if rotated) of the label.

IMPORTANT: If additional information about this bar code is required, refer to www.aimglobal.org.

Format: ^BPo,e,h,f,g

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** current ^FW value
e = print check digit	**Values:** N = no Y = yes **Default:** N
h = barcode height (in dots)	**Values:** N = no Y = yes **Default:** N
f = print interpretation line	**Values:** N = no Y = yes **Default:** Y
g = print interpretation line above code	**Values:** N = no Y = yes **Default:** N

Example

This is an example of a Plessey barcode:

ZPL II CODE	PLESSEY BAR CODE
^XA ^FO100,100^BY3 ^BPN,N,100,Y,N ^FD12345^FS ^XZ	

PLESSEY BAR CODE CHARACTERS
0 1 2 3 4 5 6 7 8 9 A B C D E F

^BQ

The ^BQ command produces a matrix symbology consisting of an array of nominally square modules arranged in an overall square pattern. A unique pattern at three of the symbol's four corners assists in determining barcode size, position, and inclination.

QR Code Barcode

A wide range of symbol sizes is possible, along with four levels of error correction. User-specified module dimensions provide a wide variety of symbol production techniques.

QR Code Model 1 is the original specification, while QR Code Model 2 is an enhanced form of the symbology. Model 2 provides additional features and can be automatically differentiated from Model 1.

Model 2 is the recommended model and should normally be used.

This barcode is printed using field data specified in a subsequent ^FD string.

Encodable character sets include numeric data, alphanumeric data, 8-bit byte data, and Kanji characters.

IMPORTANT: If additional information about this barcode is required, refer to www.aimglobal.org.

Format: ^BQa,b,c,d,e

Parameters	Details
a = field orientation	**Values:** normal (^FW has no effect on rotation)
b = model	**Values:** 1 (original) and 2 (enhanced – recommended) **Default:** 2
c = magnification factor	**Values:** 1 to 100 **Default:** 1 on 150 dpi printers 2 on 200 dpi printers 3 on 300 dpi printers 6 on 600 dpi printers
d = error correction	**Values:** H = ultra-high reliability level Q = high-reliability level M = standard level L = high-density level **Default:** Q = if emptyM = invalid values
e = mask value	**Values:** 0 - 7 **Default:** 7

Example

This is an example of a QR Code barcode:

ZPL II CODE	QR CODE BAR CODE
^XA ^FO100,100 ^BQN,2,10 ^FDMM,AAC-42^FS ^XZ	

On the pages that follow are specific commands for formatting the ^BQ command with the ^FD statements that contain the information to be coded.

QR Switches (formatted into the ^FD field data)

There are 4 switch fields that are allowed, some with associated parameters and some without. Two of these fields are always present, one is optional, and one's presence depends on the value of another. The switches are always placed in a fixed order. The four switches, in order, are:

Mixed mode <D>iijjxx, Optional (note that this switch ends with a comma ",")

Error correction level <H, Q, M, L> Mandatory

Data input <A, M>, Mandatory (note that this switch ends with a comma ",")

Character Mode <N, A, Bdddd, K> Conditional (present if data input is M)

Mixed mode (Optional)

= D - allows mixing of different types of character modes in one code.

ii = code No. – a 2-digit number in the range of 01 to 16

Value = subtracted from the Nth number of the divided code (must be two digits).

jj = No. of divisions – a 2-digit number in the range 02 to 16

Number of divisions (must be two digits).

xx = parity data – a 2-digit hexadecimal character in the range 00 to FF

Parity data value is obtained by calculating the input data (the original input data before divided byte-by-byte through the EX-OR operation).

, = the mixed mode switch, when present, is terminated with a comma

Error Correction Level (Required)

= H, Q, M, or L

H = ultra-high reliability level

Q = high-reliability level

M = standard level (default)

L = high-density level

Data Input (Required)

= A or M followed by a comma

A = Automatic Input (default). Character Mode is not specified.

Data character string JIS8 unit, Shift JIS. When the input mode is Automatic Input, the binary codes of 0x80 to 0x9F and 0xE0 to 0xFF cannot be set.

M = Manual Input. Character Mode must be specified.

Two types of data input modes exist: Automatic (A) and Manual (M). If A is specified, the

character mode does not need to be specified. If M is specified, the character mode must be

specified.

Character Mode (Required when data input = M)

= N, A, Bxxxx, or K

N = numeric: digits 0 – 9

A = alphanumeric: digits 0 – 9, upper case letters A – Z, space, and $%*+-./:) (45 characters)

Bxxxx = 8-bit byte mode. The 'xxxx' is the number of characters and must be exactly 4 decimal digits.

This handles the 8-bit Latin/Kana character set in accordance with JIS X 0201 (character values 0x00 to 0xFF).

K = Kanji — handles only Kanji characters in accordance with the Shift JIS system based on JIS X 0208. This means that all parameters after the character mode K should be 16-bit characters. If there are any 8-bit characters (such as ASCII code), an error occurs.

The data to be encoded follows immediately after the last switch.

Considerations for ^FD When Using the QR Code:

QR Switches (formatted into the ^FD field data)

mixed mode <D>

D = allows the mixing of different types of character modes in one code.

code No. <01 16>

Value = subtracted from the Nth number of the divided code (must be two digits).

No. of divisions <02 16>

Number of divisions (must be two digits).

parity data <1 byte>

Parity data value is obtained by calculating at the input data (the original input data before being divided byte-by-byte through the EX-OR operation).

error correction level <H, Q, M, L>

H = ultra-high reliability level

Q = high-reliability level

M = standard level (default)

L = high-density level

character Mode <N, A, B, K>

N = numeric

A = alphanumeric

Bxxxx = 8-bit byte mode. This handles the 8-bit Latin/Kana character set in accordance with JIS X 0201 (character values 0x00 to 0xFF).

xxxx = number of data characters is represented by two bytes of BCD code.

K = Kanji — handles only Kanji characters in accordance with the Shift JIS system based on JIS X 0208. This means that all parameters after the character mode **K** should be 16-bit characters. If there are any 8-bit characters (such as ASCII code), an error occurs.

data character string <Data>

Follows character mode or it is the last switch in the ^FD statement.

data input <A, M>

A = Automatic Input (default). Data character string JIS8 unit, Shift JIS. When the input mode is Automatic Input, the binary codes of 0x80 to 0x9F and 0xE0 to 0xFF cannot be set.

M = Manual Input

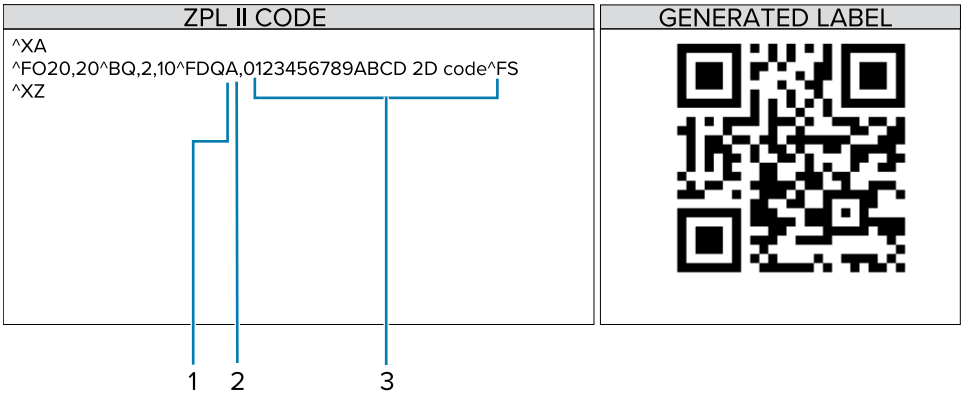
Two types of data input modes exist: Automatic (A) and Manual (M). If A is specified, the character mode does not need to be specified. If M is specified, the character mode must be specified.

^FD Field Data (Normal Mode)

Automatic Data Input (A) with Switches

```
^FD
<error correction level>A,
<data character string>
^FS
```

QR Code, normal mode with automatic data input.

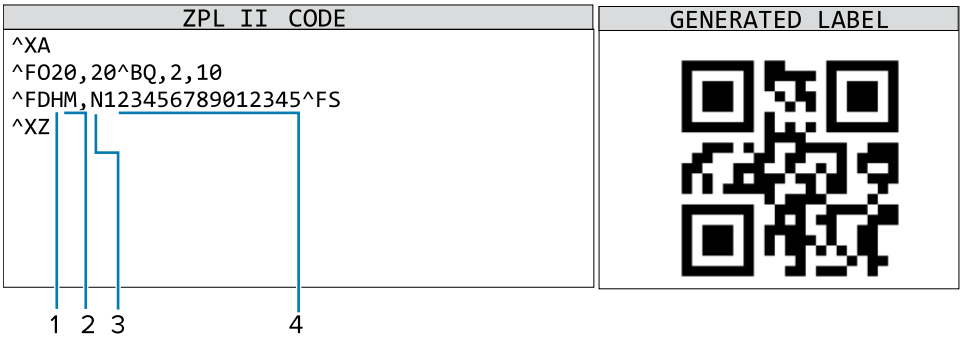


1	Q = error correction level
2	A, = automatic setting
3	data string character

Manual Data Input (M) with Switches

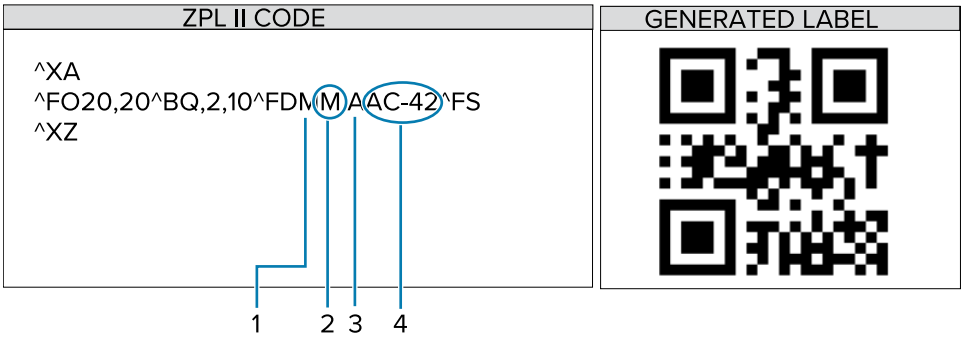
```
^FD
<error correction level>M,
<character mode><data character string>
^FS
```

QR Code, normal mode with manual data input:



1	H = error correction level (ultra-high reliability level
2	M, = input mode (manual input)
3	N = character mode (numeric data)
4	data character string

QR Code, normal mode with standard reliability and manual data input:



1	M = error correction level (standard-high reliability level
2	M, = manual input
3	A = alphanumeric data
4	AC-42 = data character string

^FD Field Data (Mixed Mode – requires more switches)

Automatic Data Input (A) with Switches

```
^FD
<D><code No.> <No. of divisions> <parity data>,
<error correction level> A,
<data character string>,
<data character string>,
< : >,
<data character string n**>
^FS
```

Manual Data Input (M) with Switches

```
^FD
<code No.> <No. of divisions> <parity data>,
<error correction level> M,
<character mode 1> <data character string 1>,
<character mode 2> <data character string 2>,
< : > < : >,
<character mode n> <data character string n**>
^FS
```

n** up to 200 in mixed mode

QR Code, mixed mode with manual data input:

ZPL II CODE	GENERATED LABEL
^XA ^FO,20,20^BQ,2,10 ^FDD03048F,LM,N0123456789,A12AABB,B0006qrcode^FS ^XZ	

<mixed mode identifier>	D	(mixed)
<code No.>	M	(code number)
<No. of divisions>	D	(divisions)
<parity data>	M	(0x0C)
	‘	
<error correction level>	L	(high-density level)
<input mode>	M	(manual input)
	‘	
<character mode>	N	(numeric data)
<data character string>		0123456789
	‘	
<character mode>	A	(alphanumeric data)
<data character string>		12AABB
	‘	
<character mode>	B	(8-bit byte data)
	0006	(number of bytes)
<data character string>		qrcode

Example: This is an example of a bQR Code, mixed mode with automatic data input:

```
^XA
^FO20,20^BQ,2,10
^FDD03040C,LA,012345678912AABBqrcode^FS
^XZ
```

<mixed mode identifier>	D	D (mixed)
<code No.>	M	03 (code number)
<No. of divisions>	D	04 (divisions)
<parity data>	M	0C (0x0C)
<error correction level>	L	L (high-density level)
<input mode>	A	A (automatic input)
<data character string>		012345678912AABBqrcode

.14↑

For proper functionality, when encoding Kanji characters in `^CI28-30` (Unicode) be sure the `JIS.DAT` table is loaded on the printer and specified.

Example:

This is a Unicode example:

ZPL II CODE	QR CODE BAR CODE
^XA ^FO100,100 ^BQN,2,10 ^FDMM,AAC-42^FS ^XZ	

^BR

The ^BR command is a barcode type for space-constrained identification from EAN International and the Uniform Code Council, Inc.

GS1 Databar (formerly Reduced Space Symbology)

Format: ^BRa,b,c,d,e,f

Parameters	Details
a = orientation	**Values:** N = Normal R = Rotated I = Inverted B = Bottom-up **Default:** R
b = symbology type in the GS1 DataBar family	**Values:** 1 = GS1 DataBar Omnidirectional 2 = GS1 DataBar Truncated 3 = GS1 DataBar Stacked 4 = GS1 DataBar Stacked Omnidirectional 5 = GS1 DataBar Limited 6 = GS1 DataBar Expanded 7 = UPC-A 8 = UPC-E 9 = EAN-13 10 = EAN-8 11 = UCC/EAN-128 and CC-A/B12 = UCC/EAN-128 and CC-C **Default:** 1
c = magnification factor	**Values:** 1 to 10 **Default:** 24 dot = 6, 12 dot is 3, 8 dot and lower is 212 dot = 6, > 8 dot is 3, 8 dot and less is 2
d = separator height	**Values:** 1 or 2 **Default:** 1
e = Barcode height	The bar code height only affects the linear portion of the barcode. Only UCC/EAN and CC-A/B/C. **Values:** 1 to 32000 dots **Default:** 25
f = the segment width (GS1 DataBar Expanded only)	**Values:** 2 to 22, even numbers only, in segments per line **Default:** 22

Examples

This is an example of Symbology Type 7 - UPC-A:

ZPL II CODE	GENERA TED LABEL
^XA ^FO10,10^BRN,7,5,2,100 ^FD12345678901|this is composite info^FS ^XZ	

This is an example of Symbology Type 1 - GS1 DataBar Omnidirectional:

ZPL II CODE	GENERA TED LABEL
^XA ^FO10,10^BRN,1,5,2,100 ^FD12345678901|this is composite info^FS ^XZ	

^BS

The ^BS command is the two-digit and five-digit add-on used primarily by publishers to create barcodes for ISBNs (International Standard Book Numbers). These extensions are handled as separate bar codes.

UPC/EAN Extensions

The ^BS command is designed to be used with the UPC-A barcode (^BU) and the UPC-E barcode (^B9).

- ^BS supports a fixed print ratio.
- Field data (^FD) is limited to exactly two or five characters. ZPL II automatically truncates or pads on the left with zeros to achieve the required number of characters.

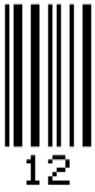
IMPORTANT: If additional information about this barcode is required, go to www.aimglobal.org.

Format: ^BSo,h,f,g

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** current ^FW value
h = Barcode height (in dots)	**Values:** 1 to 32000 **Default:** value set by ^BY
f = print interpretation line	**Values:** N = no Y = yes **Default:** Y
g = print interpretation line above code	**Values:** N = no Y = yes **Default:** Y

Example

This is an example of a UPC/EAN Two-digit barcode:

ZPL II CODE	UPC/EAN 2-DIGIT BAR CODE
^XA ^FO100,100^BY3 ^BSN,100,Y,N ^FD12^FS ^XZ	

UPC/EAN 2-DIGIT BAR CODE CHARACTERS
0 1 2 3 4 5 6 7 8 9

Example

This is an example of a UPC/EAN Five-digit barcode:

ZPL II CODE	UPC/EAN 5-DIGIT BAR CODE
^XA ^FO100,100^BY3 ^BSN,100,Y,N ^FD12345^FS ^XZ	

UPC/EAN 5-DIGIT BAR CODE CHARACTERS
0 1 2 3 4 5 6 7 8 9

Care should be taken in positioning the UPC/EAN extension with respect to the UPC-A or UPC-E code to ensure the resulting composite code is within the UPC specification.

Example

For UPC codes, with a module width of 2 (default), the field origin offsets for the extension are:

This is an example of a UPC-A:

	Supplement Origin X - Offset	Adjustment Y - Offset
Normal	209 Dots	21 Dots
Rotated	0	209 Dots

Example

This is an example of a UPC-E:

	Supplement Origin X - Offset	Adjustment Y - Offset
Normal	122 Dots	21 Dots
Rotated	0	122 Dots

Additionally, the barcode height for the extension should be 27 dots (0.135 inches) shorter than that of the primary code. A primary UPC code height of 183 dots (0.900 inches) requires an extension height of 155 dots (0.765 inches).

Example

This example illustrates how to create a normal UPC-A barcode for the value 7000002198 with an extension equal to 04414:

ZPL II CODE	UPC-A BAR CODE WITH EXTENSION
^XA ^FO100,100^BY3 ^BUN,137 ^FD07000002198^FS ^FO400,121 ^BSN,117 ^FD04414^FS ^XZ	

^BT

The ^BT bar code is the standard for the TCIF can tag telecommunications equipment.

TLC39 Barcode

The TCIF CLEI code, which is the MicroPDF417 barcode, is always four columns. The firmware must determine what mode to use based on the number of characters to be encoded.

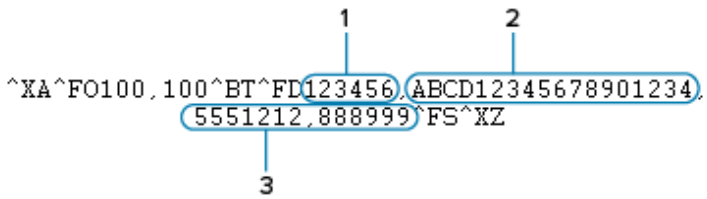
Format: ^BTo,w1,r1,h1,w2,h2

Parameters	Details
o = orientation	**Values:** N = normal R = rotated I = inverted B = bottom up
w1 = width of the Code 39 bar code	**Values: (in dots):** 1 to 10 **Default: (600 dpi printers):** 4 **Default: (200- and 300 dpi printer):** 2
r1 = wide to narrow bar width ratio the Code 39 bar code	**Values:** 2.0 to 3.0(increments of 0.1) **Default:** 2.0
h1 = height of the Code 39 bar code	**Values: (in dots):** 1 to 9999 **Default: (600 dpi printer):** 120 **Default: (300 dpi printer):** 60 **Default: (200 dpi printer):** 40
h2 = row height of the MicroPDF417 bar code	**Values: (in dots):** 1 to 255 **Default: (600 dpi printer):** 8 **Default: (200- and 300 dpi printers):** 4
w2 = narrow bar width of the MicroPDF417 bar code	**Values: (in dots):** 1 to 10 **Default: (600 dpi printer):** 4 **Default: (200- and 300 dpi printers):** 2

How to Print TLC39 Barcode

Example: This is an example on how to print TLC39 barcode. The callouts identify the key components and are followed by a detailed description below:

Use the command defaults to get results that are in compliance with TCIF industry standards; regardless of printhead density.



1	**ECI Number.** If the seventh character is not a comma, only Code 39 prints. This means if more than 6 digits are present, Code 39 prints for the first six digits (and no Micro-PDF symbol is printed). • Must be 6 digits. • Firmware generates invalid character error if the firmware sees anything but 6 digits. • This number is not padded.
2	**Serial number.** The serial number can contain up to 25 characters and is variable length. The serial number is stored in the Micro-PDF symbol. If a comma follows the serial number, then additional data is used below. • If present, must be alphanumeric (letters and numbers, no punctuation). This value is used if a comma follows the ECI number.

3	**Additional data.** If present, it is used for things such as a country code.
	Data cannot exceed 150 bytes. This includes serial number commas.
	• Additional data is stored in the Micro-PDF symbol and appended after the serial number. A comma must exist between each maximum of 25 characters in the additional fields.
	• Additional data fields can contain up to 25 alphanumeric characters per field.
	The result is:
	ZPL II CODE
	`^XA^FO100,`
	`100^BT^FD123456,`
	`ABCd12345678901234,`
	`5551212,`
	`88899`
	`^FS^XZ`
	GENERA TED LABEL

^BU

The ^BU command produces a fixed length, numeric symbology. It is primarily used in the retail industry for labeling packages. The UPC-A barcode has 11 data characters. The 6 dot/mm, 12 dot/mm, and 24 dot/mm printheads produce the UPC-A barcode (UPC/EAN symbologies) at 100 percent size. However, an 8 dot/mm printhead produces the UPC/EAN symbologies at a magnification factor of 77 percent.

UPC-A Barcode

- ^BU supports a fixed print ratio.
- Field data (^FD) is limited to exactly 11 characters. ZPL II automatically truncates or pads on the left with zeros to achieve the required number of characters.

IMPORTANT: If additional information about this barcode is required, go to www.aimglobal.org.

Format: ^BUo,h,f,g,e

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** current ^FW value
h = Barcode height (in dots)	**Values:** 1 to 9999 **Default:** value set by ^BY
f = print interpretation line	**Values:** N = no Y = yes **Default:** Y
g = print interpretation line above code	**Values:** N = no Y = yes **Default:** N
e = print check digit	**Values:** N = no Y = yes **Default:** Y

The font style of the interpretation line depends on the modulus (width of narrow bar) selected in ^BY. Zero is not allowed.

- **6 dot/mm printer:** a modulus of 2 dots or greater prints with an OCR-B interpretation line; a modulus of 1 dot prints font A.
- **8 dot/mm printer:** a modulus of 3 dots or greater prints with an OCR-B interpretation line; a modulus of 1 or 2 dots prints font A.

- **12 dot/mm printer:** a modulus of 5 dots or greater prints with an OCR-B interpretation line; a modulus of 1, 2, 3, or 4 dots prints font A.
- **24 dot/mm printer:** a modulus of 9 dots or greater prints with an OCR-B interpretation line; a modulus of 1 to 8 dots prints font A.

Example

This is an example of a UPC-A barcode with extension:

ZPL II CODE	UPC-A BAR CODE WITH EXTENSION
^XA ^FO100,100^BY3 ^BUN,137 ^FD07000002198^FS ^FO400,121 ^BSN,117 ^FD04414^FS ^XZ	

Comments: The UPC-A bar code uses the Mod 10 check digit scheme for error checking. For further information on Mod 10, see Mod 10 Check Digit on page 1568.

^BX

The ^BX command creates a two-dimensional matrix symbology made up of square modules arranged within a perimeter finder pattern.

Data Matrix Barcode

Format: ^BXo,h,s,c,r,f,g,a

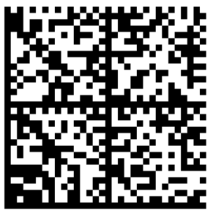
Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** current ^FW value
h = dimensional height of individual symbol elements	**Values:** 1 to the width of the label The individual elements are square — this parameter specifies both module and row height. If this parameter is zero (or not given), the h parameter (bar height) in ^BY is used as the approximate symbol height.
s = quality level	**Values: 0, 50, 80, 100, 140, 200** **Default:** 0 **Quality** refers to the amount of data that is added to the symbol for error correction. The AIM specification refers to it as the ECC value. ECC 50, ECC 80, ECC 100, and ECC 140 use convolution encoding; ECC 200 uses Reed-Solomon encoding. For new applications, ECC 200 is recommended. ECC 000-140 should be used only in closed applications where a single party controls both the production and reading of the symbols and is responsible for overall system performance.
c = columns to encode	**Values:** 9 to 49 Odd values only for quality 0 to 140 (10 to 144); even values only for quality 200. Odd values only for quality 0 to 140 (10 to 144); even values only for quality 200. The number of rows and columns in the symbol is automatically determined. You might want to force the number of rows and columns to a larger value to achieve uniform symbol size. In the current implementation, quality 0 to 140 symbols are square, so the larger of the rows or columns supplied is used to force a symbol to that size. If you attempt to force the data into too small of a symbol, no symbol is printed. If a value greater than 49 is entered, the rows or columns value is set to zero, and the size is determined normally. If an even value is entered, it generates INVALID-P (invalid parameter). If a value is less than 9 but not 0, or if the data is too large for the forced size, no symbol prints; if ^CV is active, INVALID-L prints.
r = rows to encode	**Values:** 9 to 49

Parameters	Details
f = format ID (0 to 6) — not used with quality set at 200	**Values:** 1 = field data is numeric + space (0..9,") – No \&" 2 = field data is uppercase alphanumeric + space (A..Z,") – No \&" 3 = field data is uppercase alphanumeric + space, period, comma, dash, and slash (0..9,A..Z,".-/") 4 = field data is upper-case alphanumeric + space (0..9,A..Z,") – no \&" 5 = field data is full 128 ASCII 7-bit set 6 = field data is full 256 ISO 8-bit set **Default:** 6
g = escape sequence control character	**Values:** any character **Default:** ~ (tilde) This parameter is used only if quality 200 is specified. It is the escape character for embedding special control sequences within the field data. A value must always be specified when using the escape sequence control character. If no value is entered, the command is ignored. The g parameter will continue to be underscore (_) for anyone with firmware version: V60.13.0.12, V60.13.0.12Z, V60.13.0.12B, V60.13.0.12ZB, or later.
a = aspect ratio .16↑ The a parameter is only supported in V60.16.5Z and V53.16.5Z or later.	**Values:** 1 = square 2 = rectangular **Default:** 1

ECC LEVEL	ID = 1	ID = 2	ID = 3	ID = 4	ID = 5	ID = 6
0	596	452	394	413	310	271
50	457	333	291	305	228	200
80	402	293	256	268	201	176
100	300	218	190	200	150	131
140	144	105	91	96	72	63

Maximum Field Sizes

Example: This is an example of a square Data Matrix barcode:

ZPL II CODE	DATA MATRIX BAR CODE
^XA ^FO100,100 ^BXN,10,200 ^FDZEBRA TECHNOLOGIES CORPORATION 333 CORPORATE WOODS PARKWAY VERNON HILLS, IL 60061-3109^FS ^XZ	

Example: This is an example of a rectangle Data Matrix bar code:

ZPL II CODE	DATA MATRIX BAR CODE
^XA ^FO100,100 ^BXN,10,200,,,,,2 ^FDZEBRA TECHNOLOGIES CORPORATION 333 CORPORATE WOODS PARKWAY ^FS ^XZ	

Effects of ^BY on ^BX

w = **module width** (no effect)

r = **ratio** (no effect)

h = **height of symbol**

If the dimensions of individual symbol elements are not specified in the ^BY command, the height of the symbol value is divided by the required rows/columns, rounded, limited to a minimum value of one, and used as the dimensions of individual symbol elements.

Field Data (^FD) for ^BX

Quality 000 to 140

- The **\&** and ll can be used to insert carriage returns, line feeds, and backslash, similar to the PDF417. Other characters in the control character range can be inserted only by using ^FH. Field data is limited to 596 characters for quality **0** to **140**. Excess field data causes no symbol to print; if ^CV is active, INVALID-L prints. The field data must correspond to a user-specified format ID or no symbol prints; if ^CV is active, INVALID-C prints.
- The maximum field sizes for quality **0** to **140** symbols are shown in the table in the g parameter.

Quality 200

- If more than 3072 bytes are supplied as field data, it is truncated to 3072 bytes. This limits the maximum size of a numeric Data Matrix symbol to less than the 3116 numeric characters that the specification would allow. The maximum alphanumeric capacity is 2335, and the maximum 8-bit byte capacity is 1556.
- If ^FH is used, field hexadecimal processing takes place before the escape sequence processing described below.
- The underscore is the default escape sequence control character for quality 200 field data. A different escape sequence control character can be selected by using parameter g in the ^BX command.

The information that follows applies to firmware versions: V60.13.0.12, V60.13.0.12Z, V60.13.0.12B, V60.13.0.12ZB, or later. The input string escape sequences can be embedded in quality 200 field data using the ASCII 95 underscore character (_) or the character entered in parameter g:

- _X is the shift character for control characters (e.g., _@=NUL,_G=BEL,_0 is PAD)
- _1 to _3 for FNC characters 1 to 3 (explicit FNC4, upper shift, is not allowed)
- FNC2 (Structured Append) must be followed by nine digits, composed of three-digit numbers with values between 1 and 254, that represent the symbol sequence and file identifier (for example, symbol 3 of 7 with file ID 1001 is represented by _2214001001)
- 5NNN is code page NNN where NNN is a three-digit code page value (for example, Code Page 9 is represented by _5009)
- _dNNN creates ASCII decimal value NNN for a code word (must be three digits)
- _ in data is encoded by __ (two underscores)

The information that follows applies to all other versions of firmware. The input string escape sequences can be embedded in quality 200 field data using the ASCII 7E tilde character (~) or the character entered in the parameter g:

- ~X is the shift character for control characters (e.g., ~@=NUL,~G=BEL,~0 is PAD)
- ~1 to ~3 for FNC characters 1 to 3 (explicit FNC4, upper shift, is not allowed)
- FNC2 (Structured Append) must be followed by nine digits, composed of three-digit numbers with values between 1 and 254, that represent the symbol sequence and file identifier (for example, symbol 3 of 7 with file ID 1001 is represented by ~2214001001)
- 5NNN is code page NNN where NNN is a three-digit code page value (for example, Code Page 9 is represented by ~5009)
- ~dNNN creates ASCII decimal value NNN for a code word (must be three digits)
- ~ in data is encoded by a ~ (tilde)

^BY

The ^BY command is used to change the default values for the module width (in dots), the wide bar to narrow bar width ratio and the bar code height (in dots). It can be used as often as necessary within a label format.

Bar Code Field Default

Format: ^BYw,r,h

Parameters	Details
w = module width (in dots)	**Values:** 1 to 10 **Initial Value at Power Up:** 2
r = wide bar to narrow bar width ratio	**Values:** 2.0 to 3.0, in 0.1 increments This parameter has no effect on fixed-ratio bar codes. **Default:** 3.0
h = bar code height (in dots)	**Initial Value at Power Up:** 10

For parameter r, the actual ratio generated is a function of the number of dots in parameter w, module width. See the table below. Module width and height (w and h) can be changed at anytime with the ^BY command, regardless of the symbology selected.

Table 6 Module Width Ratios in Dots

Ratio Selected (r)	Module Width in Dots (w)									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
2.0	2:1	2:1	2:1	2:1	2:1	2:1	2:1	2:1	2:1	2:1
2.1	2:1	2:1	2:1	2:1	2:1	2:1	2:1	2:1	2:1	2.1:1
2.2	2:1	2:1	2:1	2:1	2.2:1	2.16:1	2.1:1	2.12:1	2.11:1	2.2:1
2.3	2:1	2:1	2.3:1	2.25:1	2.2:1	2.16:1	2.28:1	2.25:1	2.2:1	2.3:1
2.4	2:1	2:1	2.3:1	2.25:1	2.4:1	2.3:1	2.28:1	2.37:1	2.3:1	2.41:1
2.5	2:1	2.5:1	2.3:1	2.5:1	2.4:1	2.5:1	2.4:1	2.5:1	2.4:1	2.5:1
2.6	2:1	2.5:1	2.3:1	2.5:1	2.6:1	2.5:1	2.57:1	2.5:1	2.5:1	2.6:1
2.7	2:1	2.5:1	2.6:1	2.5:1	2.6:1	2.6:1	2.57:1	2.65:1	2.6:1	2.7:1
2.8	2:1	2.5:1	2.6:1	2.75:1	2.8:1	2.6:1	2.7:1	2.75:1	2.7:1	2.8:1
2.9	2:1	2.5:1	2.6:1	2.75:1	2.8:1	2.8:1	2.85:1	2.87:1	2.8:1	2.9:1
3.0	3:1	3:1	3:1	3:1	3:1	3:1	3:1	3:1	3:1	3:1

Example: Set module width (w) to 9 and the ratio (r) to 2.4. The width of the narrow bar is 9 dots wide and the wide bar is 9 by 2.4, or 21.6 dots. However, since the printer rounds out to the nearest dot, the wide bar is actually printed at 22 dots. This produces a bar code with a ratio of 2.44 (22 divided by 9). This ratio is as close to 2.4 as possible, since only full dots are printed.

Comments: Once a ^BY command is entered into a label format, it stays in effect until another ^BY command is encountered.

^BZ

The POSTAL barcode is used to automate the handling of mail. POSTAL codes use a series of tall and short bars to represent the digits.

- ^BZ supports a print ratio of 2.0:1 to 3.0:1.
- Field data (^FD) is limited to the width (or length, if rotated) of the label and by the barcode specification.

IMPORTANT: If additional information about the POSTAL and PLANET barcode is required, go to www.aimglobal.org, or contact the United States Postal Service at http://pe.usps.gov. If additional information about the INTELLIGENT MAIL barcode is required, see: http://ribbs.usps.gov/OneCodeSolution.

Format: ^BZo,h,f,g,t

Parameters	Details
o = orientation	**Values:** N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from the bottom up, 270 degrees **Default:** current ^FW value
h = Barcode height (in dots)	**Values:** 1 to 32000 **Default:** value set by ^BY
f = print interpretation line	**Values:** N = no Y = yes **Default:** N
g = print interpretation line above code	**Values:** N = no Y = yes **Default:** N
t = Postal code type	**Values:** 0 = Postnet barcode 1 = Plant barcode 2 = Reserved 3 = USPS Intelligent Mail barcode **Default:** 0

Examples

This is an example of a POSTNET barcode:

ZPL II CODE	POSTNET BAR CODE
^XA ^FO100,100^BY3 ^BZN,40,Y,N ^FD12345^FS ^XZ	 12345

POSTNET BAR CODE CHARACTERS
0 1 2 3 4 5 6 7 8 9

This is an example of a USPS Intelligent Mail barcode:

ZPL II CODE	USPS INTELLIGENT MAIL BAR CODE
^XA ^FO100,040^BZ,40,,,3 ^FD00123123456123456789^FS ^XZ	

^CC ~CC

The ^CC command is used to change the format command prefix. The default prefix is the caret (^).

Change Caret

Format: ^CCx or ~CCx

Parameters	Details
x = caret character change	**Values:** any ASCII character **Default:** a parameter is required. If a parameter is not entered, the next character received is the new prefix character.

Example: This is an example of how to change the format prefix to / from a ::

```
^XA
^CC/
/XZ
```

The forward slash (/) is set at the new prefix. Note the /XZ ending tag uses the new designated prefix character (/).

Example: This is an example of how to change the format prefix from ~ to a /:

```
~CC/
/XA/JUS/XZ
```

^CD ~CD

The ^CD and ~CD commands are used to change the delimiter character. This character is used to separate parameter values associated with several ZPL II commands. The default delimiter is a comma (,).

Change Delimiter

Format:^CDa or ~CDa

Parameters	Details
a = delimiter character change	**Values:** any ASCII character **Default:** a parameter is required. If a parameter is not entered, the next character received is the new prefix character.

Example: This shows how to change the character delimiter to a semi-colon (;):

```
^XA
^FO10,10
^GB10,10,3
^XZ
^XA
^CD;
^FO10;10
^GB10;10;3
^XZ
```

To save, the JUS command is required. Here is an example using JUS:

```
~CD;
^XA^JUS^XZ
```

^CF

The ^CF command sets the default font used in your printer. You can use the ^CF command to simplify your programs.

Change the Alphanumeric Default Font

Format: ^CFf,h,w

Parameters	Details
f = specified default font	**Values:** A through Z and 0 to 9 **Initial Value at Power Up:** A
h = individual character height (in dots)	**Values:** 0 to 32000 **Initial Value at Power Up:** 9
w = individual character width (in dots)	**Values:** 0 to 32000 **Initial Value at Power Up:** 5 or last permanent saved value

Parameter f specifies the default font for every alphanumeric field. Parameter h is the default height for every alphanumeric field, and parameter w is the default width value for every alphanumeric field.

The default alphanumeric font is A. If you do not change the alphanumeric default font and do not use any alphanumeric field command (^AF) or enter an invalid font value; any data you specify prints in font A.

Defining only the height or width forces the magnification to be proportional to the parameter defined. If neither value is defined, the last ^CF values given or the default ^CF values for height and width are used.

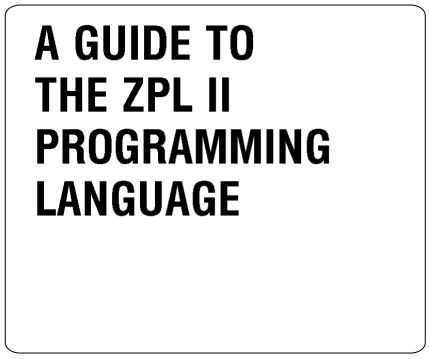
Example

This is an example of ^CF code and the result of the code:

ZPL II CODE

```
^XA
^CF0,89
^FO20,50
^FDA GUIDE TO^FS
^FO20,150
^FDTHE ZPL II^FS
^FO20,250
^FDPROGRAMMING^FS
^FO20,350
^FDLANGUAGE^FS
^XZ
```

GENERATED LABEL



Comments: Any font in the printer, including downloaded fonts, EPROM stored fonts, and fonts A through Z and 0 to 9, can also be selected with ^CW.

^CI

The ^CI command enables you to call up the international character set you want to use for printing. You can mix character sets on a label.

Change International Font/Encoding

Zebra printers can print fonts using international character sets: U.S.A.1, U.S.A.2, UK, Holland, Denmark/Norway, Sweden/Finland, Germany, France 1, France 2, Italy, Spain, and several other sets, including the Unicode character set.

A character within a font can be remapped to a different numerical position.

In x.14 version of firmware and later, this command allows character remapping when the parameter a = 0-13.

Format: ^CIa,s1,d1,s2,d2,...

Parameters	Details
a = desired character set **NOTE:** Note: These parameters are only valid when the parameter a = 1 - 13	Accepted values are 0 - 12 are Zebra Code Page 850 with specific character replacements. For details, see International Character Sets on page 159 and/or Zebra Code Page 1252— Latin Character Set on page 1547. **Values:** • 0 = Single Byte Encoding - U.S.A. 1 Character Set • 1 = Single Byte Encoding - U.S.A. 2 Character Set • 2 = Single Byte Encoding - U.K. Character Set • 3 = Single Byte Encoding - Holland Character Set • 4 = Single Byte Encoding - Denmark/Norway Character Set • 5 = Single Byte Encoding - Sweden/Finland Character Set • 6 = Single Byte Encoding - Germany Character Set • 7 = Single Byte Encoding - France 1 Character Set • 8 = Single Byte Encoding - France 2 Character Set • 9 = Single Byte Encoding - Italy Character Set • 10 = Single Byte Encoding - Spain Character Set • 11 = Single Byte Encoding - Miscellaneous Character Set • 12 = Single Byte Encoding - Japan (ASCII with Yen symbol) Character Set • 13 = Zebra Code Page 850 (see Zebra Code Page 850 — Latin Character Set on page 1543) • 14 = Double Byte Asian Encodings[1] • 15 = Shift-JIS[2] • 16 = EUC-JP and EUC-CN[1]

Parameters	Details
a = desired character set(continued) .14↑ Values 28 to 30 are supported only in firmware version V60.14.x, V50.14.x, or later. .16↑ Values 31 to 36 are supported only in firmware version x.16.x or later.	• 17 = Deprecated - UCS-2 Big Endian[3] • 18 to 23 = Reserved • 24 = Single Byte Asian Encoding[1] • 25 = Reserved • 26 = Multibyte Asian Encodings with ASCII Transparency[1, 4] • 27 = Zebra Code Page 1252 (see Zebra Code Page 1252— Latin Character Set on page 1547) • 28 = Unicode (UTF-8 encoding) - Unicode Character Set • 29 = Unicode (UTF-16 Big-Endian encoding) - Unicode Character Set • 30 = Unicode (UTF-16 Little-Endian encoding) - Unicode Character Set • 31 = Zebra Code Page 1250 (see Zebra Code Page 1250 — Central and Eastern European Latin Character Set on page 1545) is supported for scalable fonts, such as Font 0, or a downloaded TrueType font. Bitmapped fonts (including fonts A-H) do **not** fully support Zebra Code Page 1250. This value is supported only on Zebra G-Series™ printers. • 33 = Code Page 1251 • 34 = Code page 1253 • 35 = Code Page 1254 • 36 = Code Page 1255 Initial Value at Power Up:0
s1 = source 1 (character output image)	**Values:** decimals 0 to 255
d1 = destination 1 (character input)	**Values:** decimals 0 to 255
s2 = source 2 (character output image)	**Values:** decimals 0 to 255
d2 = destination 2 (character input)	**Values:** decimals 0 to 255
... = continuation of pattern	Up to 256 source and destination pairs can be entered in this command.

1. The encoding is controlled by the conversion table (*.DAT). The correct table must be present for the conversion to function. The table generated by ZTools™ is the TrueType font's internal encoding (Unicode).

2. Shift-JIS encoding converts Shift-JIS to JIS and then looks up the JIS conversion in JIS.DAT. This table must be present for Shift-JIS to function.

3. The ^CI17 command has been deprecated, along with the ^F8 and ^F16 commands that are required for the ^CI17 command to function. The recommended replacement is the ^CI28-30 commands.

4. Supports ASCII transparency for Asian encodings. 7F and less are treated as single-byte characters. 80 to FE is treated as the first byte of a 2-byte character 8000 to FEFF in the encoding table for Unicode.

.14↑

80 to FF could mean a quad byte in GB18030. The ^CI26 command can also be used to support the GB 18030 and Big5 HKSCS encodings. The GB 18030 uses the GB18030.DAT encoding table and BIG5 HKSCS uses the BIG5HK.DAT encoding table.

.14↑

The ^CI17 command has been deprecated, along with the ^F8 and ^F16 commands that are required for the ^CI17 command to function. The recommended replacement is the ^CI28-30 commands.

.14↑

We recommend that a ^CI command (or Unicode BOM) is included at the beginning of each ZPL script. This is important when ZPL scripts with different encodings are being sent to a single printer. To assist in the interleaving of encoding schemes, the printer maintains two encoding states (^CI0 - 28 and ^CI29 - 30). It automatically acknowledges when it should switch encoding states, allowing it to distinguish between encodings, and maintains a ^CI for each, but endianness is shared.

Example: This example remaps the Euro symbol (21) decimal to the dollar sign value (36) decimal. When the dollar sign character is sent to the printer, the Euro symbol prints:

ZPL II CODE	GENERATED LABEL
^XA ^CI0,21,36 ^FO100,200^A0N50,50^FD$0123^FS ^XZ	€0123

The font selected determines the shape and resolution of the printed symbol.

International Character Sets

Hex	23	30	40	5B	5C	5D	5E	60	7B	7C	7D	7E
CI0	#	0	@	[	¢	]	^	`	{	¦	}	~
CI1	#	0	@	⅓	¢	⅔	^	`	¼	½	¾	~
CI2	£	0	@	[	¢	]	^	`	{	¦	}	~
CI3	ƒ	0	§	[	IJ	]	^	`	{	ij	}	~
CI4	#	0	@	Æ	Ø	Å	^	`	æ	ø	å	~
CI5	Ü	0	É	Ä	Ö	Å	Ü	é	ä	ö	å	ü
CI6	#	0	§	Ä	Ö	Ü	^	`	ä	ö	ü	ß
CI7	£	0	à	[	ç	]	^	`	é	¦	ù	è
CI8	#	0	à	â	ç	ê	î	ô	é	ù	è	û
CI9	£	0	§	[	ç	é	^	ù	à	ò	è	ì
CI10	#	0	§	¡	Ñ	¿	^	`	{	ñ	ç	~
CI11	£	0	É	Ä	Ö	Ü	^	ä	ë	ï	ö	ü
CI12	#	0	@	[	¥	]	^	`	{	¦	}	~
CI13	#	0	@	[	\	]	^	`	{	¦	}	~

NOTE: Note: `^CI 13 = US keyboard`

Comments: The space character cannot be remapped for any font.

^CM

The ^CM command allows you to reassign a letter designation to the printer's memory devices. If a format already exists, you can reassign the memory device to the corresponding letter without forcing, altering, or recreating the format itself.

Change Memory Letter Designation

Using this command affects every subsequent command that refers to specific memory locations.

Format: ^CMa,b,c,d

Parameters	Details
a = memory alias for B:	**Values:** B:, E:,R:, A:, and NONE **Default:** B:
b = memory alias for E:	**Values:** B:, E:, R:, A:, and NONE **Default:** E:
c = memory alias for R:	**Values:** B:, E:, R:, A:, and NONE **Default:** R:
d = memory alias for A:	**Values:** B:, E:, R:, A:, and NONE **Default:** A:
e = multiple alias	**Values:** M, or no value **Default:** no value • This parameter is supported on Xi4 and ZM400/ZM600 printers using firmware V53.17.7Z or later. • This parameter is supported on G-Series printers using firmware versions v56.17.7Z and v61.17.7Z or later. • This parameter is supported on printers using firmware V60.17.7Z or later.

Comments: Unless the e (multiple alias) parameter is used, when two or more parameters specify the same letter designator, all letter designators are set to their default values.

It is recommended that after entering the ^CM command, ^JUS is entered to save changes to EEPROM. Any duplicate parameters entered will reset the letter designations back to the default.

If any of the parameters are out of specification, the command is ignored.

Example: This example designates letter E: to point to the B: memory device, and the letter B: to point to the E:memory device.

```
^XA
^CME,B,R,A
^JUS
^XZ
```

Example: This example designates that content sent to, or read from the B: or E: memory locations will be sent to or read from the E: memory location.

```
^XA

^CME,E,R,A,M
^JUS
^XZ
```

Example: This example designates that content sent to, or read from the A: or E: memory locations will be sent to or read from the E: memory location.

```
^XA

^CMB,E,R,E,M
^JUS
^XZ
```

Example: This example designates that content sent to, or read from the A:, B: or E: memory locations will be sent to or read from the E: memory location.

```
^XA

^CME,E,R,E,M
^JUS
^XZ
```

NOTE: The last three examples are the only valid uses of the multiple alias parameter.

^CN

The ^CN causes the printer to cycle the media cutter.

Cut Now

IMPORTANT: This command works only when the printer is in Print Mode Kiosk (^MMk). If the printer is not in Print Mode Kiosk, then using this command has no effect. See ^MM on page 305.

Supported Devices:

- KR403

Format: ^CNa

Parameters	Details
a = Cut Mode Override	**Values:** 0 = Use the “kiosk cut amount” setting from ^KV 1 = Ignore “kiosk cut amount” from ^KV and do a full cut **Default:** none The command is ignored if parameters are missing or invalid.

^CO

The ^CO command is used to change the size of the character cache. By definition, a **character cache** (referred to as cache) is a portion of the DRAM reserved for storing scalable characters. All printers have a default 40K cache that is always turned on. The maximum single character size that can be stored, without changing the size of the cache, is 450 dots by 450 dots.

Cache On

There are two types of fonts used in Zebra printers: bitmapped and scalable. Letters, numbers, and symbols in a bitmapped font have a fixed size (for example: 10 points, 12 points, 14 points). By comparison, scalable fonts are not fixed in size.

Because their size is fixed, bitmapped fonts can be moved quickly to the label. In contrast, scalable fonts are much slower because each character is built on an as-needed basis before it is moved to the label. By storing scaled characters in a cache, they can be recalled at a much faster speed.

The number of characters that can be stored in the cache depends on two factors: the size of the cache (memory) and the size of the character (in points) being saved. The larger the point size, the more space in the cache it uses. The default cache stores every scalable character that is requested for use on a label. If the same character, with the same rotation and size is used again, it is quickly retrieved from cache.

It is possible that after a while the print cache could become full. Once the cache is full, space for new characters is obtained by eliminating an existing character from the print cache. Existing characters are eliminated by determining how often they have been used. This is done automatically. For example, a 28-point **Q** that was used only once would be a good candidate for elimination from the cache.

Maximum size of a single print cache character is 1500 dots by 1500 dots. This would require a cache of 274K. When the cache is too small for the desired style, smaller characters might appear but larger characters do not. If possible, increase the size of the cache.

Format: ^COa,b,c

Parameters	Details
a = cache on	**Values:** N = no Y = yes **Default:** Y
b = amount of additional memory to be added to cache (in K)	**Values:** 1 to 9999 **Default:** 40
c = cache type	**Values:** 0 = cache buffer (normal fonts) 1 = internal buffer (recommended for Asian fonts) **Default:** 0

Example: To resize the print cache to 62K, assuming a 22K existing cache:

```
^COY,40
```

Example: To resize the print cache to 100K, assuming a 22K existing cache:

```
^COY,78
```

Print Cache Performance

For printing large characters, memory added to the cache by the ^C0 command is not physically added to the 22K cache already in the printer. In the second example above, the resulting 100K cache is actually two separate blocks of memory, 22K and 78K.

Because large characters need contiguous blocks of memory, a character requiring a cache of 90K would not be completely stored because neither portion of the 100K cache is big enough. Therefore, if large characters are needed, the ^C0 command should reflect the actual size of the cache you need.

Increasing the size of the cache improves the performance in printing scalable fonts. However, the performance decreases if the size of the cache becomes large and contains too many characters. The performance gained is lost because of the time involved searching the cache for each character.

Comments: The cache can be resized as often as needed. Any characters in the cache when it is resized are lost. Memory used for the cache reduces the space available for label bitmaps, graphic, and fonts.

Some Asian fonts require an internal working buffer that is much larger than the normal cache. Since most fonts do not require this larger buffer, it is now a selectable configuration option. Printing with the Asian fonts greatly reduces the printer memory available for labels, graphics, fonts, formats, and label bitmaps.

NOTE: If you have firmware x.12 or greater this command is not required because the printer firmware automatically expands the size of the character cache as needed.

^CP

The ^CP command causes the printer to move a printed label out of the presenter area in one of several ways.

Remove Label

Supported Devices:

- KR403

Format: ^CPa

Parameters	Details
a = kiosk present mode	**Values:** 0 = Eject presented page 1 = Retracts presented page 2 = Takes the action defined by c parameter of ^KV command. **Default:** none The command is ignored if parameters are missing or invalid.

^CT ~CT

The ^CT and ~CT commands are used to change the control command prefix. The default prefix is the tilde (~).

Change Tilde

Format: ^CTa or ~CTa

Parameters	Details
a = change control command character	**Values:** any ASCII character **Default:** a parameter is required. If a parameter is not entered, the next character received is the new control command character.

Example: This is an example of how to change the control command prefix from a ^ to a +:

```
^XA
^CT+
^XZ
+HS
```

^CV

The ^CV command acts as a switch to turn the code validation function on and off. When this command is turned on, all bar code data is checked for these error conditions:

Code Validation

- character not in character set
- check-digit incorrect
- data field too long (too many characters)
- data field too short (too few characters)
- parameter string contains incorrect data or missing parameter

When invalid data is detected, an error message and code is printed in reverse image in place of the bar code. The message reads `INVALID - X` where X is one of these error codes:

C = character not in character set

E = check-digit incorrect

L = data field too long

S = data field too short

P = parameter string contains incorrect data

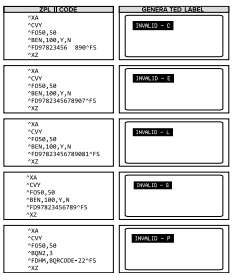
(occurs only on select bar codes)

Once turned on, the ^CV command remains active from format to format until turned off by another ^CV command or the printer is turned off. The command is not permanently saved.

Format: ^CVa

Parameters	Details
a = code validation	**Values:** N = no Y = yes **Default:** N

Example: The examples below show the error labels ^CVY generates when incorrect field data is entered. Compare the letter following **INVALID –** to the listing on the previous page.



Comments: If more than one error exists, the first error detected is the one displayed.

The ^CV command tests the integrity of the data encoded into the bar code. It is not used for (or to be confused with) testing the scan-integrity of an image or bar code.

^CW

All built-in fonts are referenced using a one-character identifier. The ^CW command assigns a single alphanumeric character to a font stored in DRAM, memory card, EPROM, or Flash.

Font Identifier

If the assigned character is the same as that of a built-in font, the downloaded font is used in place of the built-in font. The new font is printed on the label wherever the format calls for the built-in font. If used in place of a built-in font, the change is in effect only until power is turned off.

If the assigned character is different, the downloaded font is used as an additional font. The assignment remains in effect until a new command is issued or the printer is turned off.

Format: ^CWa,d:o.x

Parameters	Details
a = letter of existing font to be substituted, or new font to be added	**Values:** A through Z and 0 to 9 **Default:** a one-character entry is required
d = device to store font in (optional)	**Values:** R:, E:, B:, and A: **Default:** R:
o = name of the downloaded font to be substituted for the built-in, or as an additional font	**Values:** any name up to 8 characters **Default:** if a name is not specified, UNKNOWN is used
x = extension .14↑ .TTE is only supported in firmware version V60.14.x, V50.14.x, or later.	**Values:** .FNT = Font .TTF = TrueType Font .TTE = TrueType Extension

Example: These examples show how to use:

- MYFONT.FNT stored in DRAM whenever a format calls for Font A:

```
^XA
^CWA,R:MYFONT.FNT
^XZ
```

- MYFONT.FNT stored in DRAM additionally as Font Q:

```
^XA
^CWQ,R:MYFONT.FNT
^XZ
```

- NEWFONT.FNT stored in DRAM whenever a format calls for font F:

```
^XA
^CWF,R:NEWFONT.FNT
```

```
^XZ
```

Figure 14 Label Listing Before Assignment

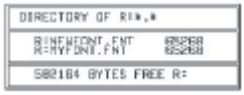


Figure 15 Label Listing After Assignment

~DB

The ~DB command sets the printer to receive a downloaded bitmap font and defines native cell size, baseline, space size, and copyright.

Download Bitmap Font

This command consists of two portions, a ZPL II command defining the font and a structured data segment that defines each character of the font.

Format: ~DBd:o.x,a,h,w,base,space,#char,©,data

Parameters	Details
d = drive to store font	**Values:** R:, E:, B:, and A: **Default:** R:
o = name of font	**Values:** 1 to 8 alphanumeric characters **Default:** if a name is not specified, UNKNOWN is used
x = extension	**Format:** .FNT
a = orientation of native font	**Fixed Value:** normal
h = maximum height of cell (in dots)	**Values:** 1 to 32000 **Default:** a value must be specified
w = maximum width of cell (in dots)	**Values:** 1 to 32000 **Default:** a value must be specified
base = dots from top of cell to character baseline	**Values:** 1 to 32000 **Default:** a value must be specified
space = width of space or non-existent characters	**Values:** 1 to 32000 **Default:** a value must be specified
#char = number of characters in font	**Values:** 1 to 256 (must match the characters being downloaded) **Default:** a value must be specified
© = copyright holder	**Values:** 1 to 63 alphanumeric characters **Default:** a value must be specified

Parameters	Details
`data` = structured ASCII data that defines each character in the font	The # symbol signifies character code parameters, which are separated with periods. The character code is from 1 to 4 characters to allow for large international character sets to be downloaded to the printer. The data structure is: `#xxxx.h.w.x.y.i.data` `#xxxx` = character code `h` = bitmap height (in dot rows) `w` = bitmap width (in dot rows) `x` = x-offset (in dots) `y` = y-offset (in dots) `i` = typesetting motion displacement (width, including inter character gap of a particular character in the font) `data` = hexadecimal bitmap description

Example: This is an example of how to use the ~DB command. It shows the first two characters of a font being downloaded to DRAM.

```
~DBR:TIMES.FNT,N,5,24,3,10,2,ZEBRA 1992,
#0025.5.16.2.5.18.
00FF
00FF
FF00
FF00
FFFF
#0037.4.24.3.6.26.
00FF00
0F00F0
0F00F0
00FF00
```

~DE

NEED SHORT DESCRIPTION

Download Encoding

The standard encoding for TrueType Windows® fonts is always Unicode. The ZPL II field data must be converted from some other encoding to Unicode that the Zebra printer understands. The required translation tables are provided with font packs. Some tables can be downloaded from zebra.com.

Format: `~DEd:o.x,s,data`

Parameters	Details
`d` = location of table	**Values:** `R:`, `E:`, `B:`, and `A:` **Default:** `R:`
`o` = name of table	**Values:** any valid name, up to 8 characters **Default:** if a name is not specified, UNKNOWN is used
`x` = extension	**Format:** `.DAT`
`s` = table size	**Values:** the number of memory bytes required to hold the Zebra downloadable format of the font **Default:** if an incorrect value or no value is entered, the command is ignored
`data` = data string	**Values:** a string of ASCII hexadecimal values **Default:** if no data is entered, the command is ignored

Example: This is an example of how to download the required translation table:

```
~DER:JIS.DAT,27848,300021213001... (27848 two-digit hexadecimal values)
```

Comments: For more information on ZTools or ZebraNet Bridge, see the program documentation included with the software.

For assistance with editing or adding mappings to `.DAT` tables, ZebraNet Bridge includes a `.DAT` table editor in the font wizard.

Encoding scheme for the data sent to the printer is the second four character and the encoding scheme for the font is the first four characters throughout the `.DAT` file. The data must be ordered by the second four characters (the encoding table).

Example: This is an example of a `.DAT` table. The table below the example identifies the elements:

~DEE:EXAMPLE.DAT,16,
00310041 — 1
00320042 — 2
00330043 — 3
00340044 — 4

Data must have 0041, 0042, 0043, and 0044 in order. Multiple pairs can be on the same line.

1	Input stream with `0041` will be mapped to 0031. The printer prints "1".

2	Input stream with 0042 will be mapped to 0032. The printer prints "2".
3	Input stream with 0043 will be mapped to 0033. The printer prints "3".
4	Input stream with 0044 will be mapped to 0034. The printer prints "4".

^DF

The ^DF command saves ZPL II format commands as text strings to be later merged using ^XF with variable data. The format to be stored might contain field number (^FN) commands to be referenced when recalled.

Download Format

While the use of stored formats reduces transmission time, no formatting time is saved—this command saves ZPL II as text strings formatted at print time.

Enter the ^DF stored format command immediately after the ^XA command, then enter the format commands to be saved.

Format: ^DFd:o.x

Parameters	Details
d = device to store the image	**Values:** R:, E:, B:, and A: **Default:** R:
o = image name	**Values:** 1 to 16 alphanumeric characters with a file type of 1 to 3 alphanumeric characters separated by a "." **Default:** if a name is not specified, UNKNOWN is used.
x = extension	**Format:** .ZPL

For a complete example of the ^DF and ^XF command, see Exercise 6: ^DF and ^XF — Download Format and Recall Format on page 57.

Example

This example is generated using the ^XF command to recall this format:

ZPL II CODE

```
^XA
^DFR:STOREFMT.ZPL^FS
^FO25,25
^AD,36,20^FN1^FS
^FO165,25
^AD,36,20^FN2^FS
^FO25,75
^AB,22,14^FDBUILT   BY^FS
^FO25,125
^AE,28,15^FN1
^XZ
^XA
^XFR:STOREFMT.ZPL^FS
^FN1^FDZEBRA^FS
^XZ
```

GENERA TED LABEL

~DG

The ~DG command downloads an ASCII Hex representation of a graphic image. If .GRF is not the specified file extension, .GRF is automatically appended.

Download Graphics

For more saving and loading options when downloading files, see ~DY on page 181.

Format: ~DGd:o.x,t,w,data

Parameters	Details
d = device to store the image	**Values:** R:, E:, B:, and A: **Default:** R:
o = image name	**Values:** 1 to 8 alphanumeric characters **Default:** if a name is not specified, UNKNOWN is used
x = extension	**Format:** .GRF
t = total number of bytes in the graphic	See the formula in the examples below.
w = number of bytes per row	See the formula in the examples below.
data = ASCII hexadecimal string defining image	The data string defines the image and is an ASCII hexadecimal representation of the image. Each character represents a horizontal nibble of four dots.

This is the key for the examples that follow:

x = width of the graphic in millimeters

y = height of the graphic in millimeters

z = dots/mm = print density of the printer being programmed

8 = bits/byte

These are some examples related to the ~DG command:

Example: To determine the t parameter use this formula:

(xz/8) x yz = totalbytes

Example: To determine the correct t parameter for a graphic 8 mm wide, 16 mm high, and a print density of 8 dots/mm, use this formula:

8 x 128 = 1024

t = 1024

Raise any portion of a byte to the next whole byte.

Example: To determine the w parameter (the width in terms of bytes per row), use this formula:

xz/8 = (totalbytes)/(row)

w = 8

Example: To determine the correct w parameter for a graphic 8 mm wide and a print density of 8 dots/mm, use this formula:

(8 x 8)/8 = 8 bytes

w = 8

Raise any portion of a byte to the next whole byte.

Parameter `w` is the first value in the `t` calculation.

The data parameter is a string of hexadecimal numbers sent as a representation of the graphic image. Each hexadecimal character represents a horizontal nibble of four dots. For example, if the first four dots of the graphic image are white and the next four black, the dot-by-dot binary code is 00001111. The hexadecimal representation of this binary value is 0F. The entire graphic image is coded in this way, and the complete graphic image is sent as one continuous string of hexadecimal values.

Example: This is an example of using the ~DG command to load a checkerboard pattern into DRAM. The name used to store the graphic is SAMPLE.GRF:

ZPL II CODE	GENERATED LABEL
~DGR:SAMPLE.GRF,00080,010, FFFFFFFFFFFFFFFFFFFF 8000FFFF0000FFFF0001 8000FFFF0000FFFF0001 8000FFFF0000FFFF0001 FFFF0000FFFF0000FFFF FFFF0000FFFF0000FFFF FFFF0000FFFF0000FFFF FFFFFFFFFFFFFFFFFFFF ^XA ^FO20,20^XGR:SAMPLE.GRF,1,1^FS ^XZ	

Comments: Do not use spaces or periods when naming your graphics. Always use different names for different graphics.

If two graphics with the same name are sent to the printer, the first graphic is erased and replaced by the second graphic.

~DN

After decoding and printing the number of bytes in parameter t of the ~DG command, the printer returns to normal Print Mode. Graphics Mode can be aborted and normal printer operation resumed by using the ~DN command.

Abort Download Graphic

Format: ~DN

Comments: If you need to stop a graphic from downloading, you should abort the transmission from the host device. To clear the ~DG command, however, you must send a ~DN command.

~DS

The ~DS command is used to set the printer to receive a downloadable scalable font and defines the size of the font in bytes.

Download Intellifont (Scalable Font)

The ~DS command, and its associated parameters, is the result of converting a vendor-supplied font for use on a Zebra printer. To convert this font use the ZTools utility.

Format: ~DSd:o.x,s,data

Parameters	Details
d = device to store image	**Values:** R:, E:, B:, and A: **Default:** R:
o = image name	**Values:** 1 to 8 alphanumeric characters **Default:** if a name is not specified, UNKNOWN is used
x = extension	**Fixed Value:** .FNT
s = size of font in bytes	**Fixed Value:** this number is generated by ZTools and should not be changed
data = ASCII hexadecimal string that defines font	**Fixed Value:** this number is generated by ZTools and should not be changed

Example: This example shows the first three lines of a scalable font that was converted using the ZTools program and is ready to be downloaded to the printer. If necessary, the destination and object name can be changed.

```
~DSB:CGTIMES.FNT,37080,
00FF00FF00FF00FF
FF0AECB28FFF00FF
```

Comments: Downloaded scalable fonts are not checked for integrity. If they are corrupt, they cause unpredictable results at the printer.

.14↑

If you are using a TrueType font use these commands: ~DT, ~DU, and ~DY. To determine when to use the noted commands, see ^DT on page 179, ^DU on page 180, and ^DY on page 181.

~DT

NEED SHORT DESCRIPTION

Download Bounded TrueType Font

Use ZTools to convert a TrueType font to a Zebra-downloadable format. that has less than 256 characters in it. To convert a font that has more than 256 characters, see ~DU on page 180. ZTools creates a downloadable file that includes a ~DT command. For information on converting and downloading Intellifont information, see ~DS on page 178.

Format: `~DTd:o.x,s,data`

Parameters	Details
`d` = font location	**Values:** `R:`, `E:`, `B:`, and `A:` **Default:** `R:`
`o` = font name	**Values:** any valid TrueType name, up to 8 characters **Default:** if a name is not specified, UNKNOWN is used
`x` = extension	**Fixed Value:** `.DAT`
`s` = font size	**Values:** the number of memory bytes required to hold the Zebra-downloadable format of the font **Default:** if an incorrect value or no value is entered, the command is ignored
`data` = data string	**Values:** a string of ASCII hexadecimal values (two hexadecimal digits/byte). The total number of two-digit values must match parameter `s`. **Default:** if no data is entered, the command is ignored

Example: This is an example of how to download a true type font:

```
~DTR:FONT,52010,00AF01B0C65E...
```

(52010 two-digit hexadecimal values)

~DU

Some international fonts, such as Asian fonts, have more than 256 printable characters. These fonts are supported as **large TrueType fonts** and are downloaded to the printer with the ~DU command.

Download Unbounded TrueType Font

Use ZTools to convert the large TrueType fonts to a Zebra-downloadable format.

The Field Block (^FB) command cannot support the large TrueType fonts.

Format: `~DUd:o.x,s,data`

Parameters	Details
`d` = font location	**Values:** `R:`, `E:`, `B:`, and `A:` **Default:** `R:`
`o` = font name	**Values:** 1 to 8 alphanumeric characters **Default:** if a name is not specified, UNKNOWN is used
`x` = extension	**Format:** `.FNT`
s = font size	**Values:** the number of memory bytes required to hold the Zebra-downloadable format of the font **Default:** if no data is entered, the command is ignored
`data` = data string	**Values:** a string of ASCII hexadecimal values (two hexadecimal digits/byte). The total number of two-digit values must match parameter `s`. **Default:** if no data is entered, the command is ignored

Example: This is an example of how to download an unbounded true type font:

```
~DUR:KANJI,86753,60CA017B0CE7...
```

(86753 two-digit hexadecimal values)

For similar commands, see ~DS on page 178, ~DT on page 179, and ~DY on page 181.

~DY

The ~DY command downloads to the printer graphic objects or fonts in any supported format. This command can be used in place of ~DG for more saving and loading options. ~DY is the preferred command to download TrueType fonts on printers with firmware later than X.13. It is faster than ~DU. The ~DY command also supports downloading wireless certificate files.

Download Objects

NOTE: Note: When using certificate files, your printer supports:

- Using Privacy Enhanced Mail (PEM) formatted certificate files.
- Using the client certificate and private key as two files, each downloaded separately.
- Using exportable PAC files for EAP-FAST.
- Zebra recommends using Linear style memory devices for storing larger objects.

Format: ~DVd:f,b,x,t,w,data

Parameters	Details
d = file location .15↑ .NRD and .PAC files reside on E: in firmware versions V60.15.x, V50.15.x, or later.	**Values:** R:, E:, B:, and A: **Default:** R:
f = file name	**Values:** 1 to 8 alphanumeric characters **Default:** if a name is not specified, UNKNOWN is used
b = format downloaded in data field .14↑ .TTE and .TTF are only supported in firmware versions V60.14.x, V50.14.x, or later.	**Values:** A = uncompressed (ZB64, ASCII) B = uncompressed (.TTE, .TTF, binary) C = AR-compressed (used only by Zebra's BAR-ONE® v5) P = portable network graphic (.PNG) - ZB64 encoded **Default:** a value must be specified

Parameters	Details
x = extension of stored file .14↑ .TTE and .OTF are only supported in firmware versions V60.14.x, V50.14.x, or later. .15↑ .NRD and .PAC are only supported in firmware versions V60.15.x, V50.15.x, or later.	**Values:** B = bitmap E = TrueType Extension (.TTE) G = raw bitmap (.GRF) P = store as compressed (.PNG) T = TrueType (.TTF) or OpenType (.OTF) X = Paintbrush (.PCX) NRD = Non Readable File (.NRD) PAC = Protected Access Credential (.PAC) C = User defined menu file (WML) F = User defined webpage file (HTM) H = Printer feedback file (GET) **Default:** a value other than the accepted values defaults to .GRF
t = total number of bytes in file **Figure 16** .14↑ .TTE is only supported in firmware versions V60.14.x, V50.14.x, or later.	**Values:** .BMP This parameter refers to the actual size of the file, not the amount of disk space. .GRF images: the size after decompression into memory This parameter refers to the actual size of the file, not the amount of disk space. .PCX This parameter refers to the actual size of the file, not the amount of disk space. .PNG images: This parameter refers to the actual size of the file, not the amount of disk space. .TTF This parameter refers to the actual size of the file, not the amount of disk space. .TTE This parameter refers to the actual size of the file, not the amount of disk space.

Parameters	Details
w = total number of bytes per row .14↑ .TTE is only supported in firmware version V60.14.x, V50.14.x, or later. .15↑ .NRD and .PAC files are supported in firmware version V60.15.x, V50.15.x, or later.	**Values:** .GRF images: number of bytes per row .PNG images: value ignored .TTF images: value ignored .TTE images: value ignored .NRD images: value ignored .PAC images: value ignored
data = data	ASCII hexadecimal encoding, ZB64, or binary data, depending on b. A, P = ASCII hexadecimal or ZB64 B, C = binary When binary data is sent, all control prefixes and flow control characters are ignored until the total number of bytes needed for the graphic format is received.

NOTE: When transmitting fonts or graphics, the ~DY command and the binary content can be sent as two separate data streams. In cases where the ~DY command and data content are sent separately, the connection to the printer must be maintained until both the command and data content have been sent. If the command and data content are sent separately, the data light on the printer will remain lit until it receives all the data called for in the ~DY command. The download will be considered complete when the number of bytes called out in the ~DY command have been received.

For best results, graphic files must be monochrome (black and white) or dithered.

Example: This is an example of how to download a binary TrueType Font file of Size bytes using the name fontfile.ttf and storing it to permanent flash memory on the printer:

```
~DYE:FONTFILE.TTF,B,T,SIZE,,
```

These examples show:

- that when the ^IM command is used with the ^FO command, the ^IM command (see ^IM on page 248) moves the logo.png file from a storage area to the 0,0 position on the label. This is the ZPL code:

```
^XA
^FO0,0^IMR:LOGO.PNG^FS
^XZ
```

- that when the ^IL command (see ^IL on page 247) is used at the beginning of a label format, it loads a stored image (logo.png) of a format and merges it with additional data. It is automatically positioned at the 0,0 position of the label and does not require the ^FO command. This is the ZPL code:

```
^XA
^ILR:LOGO.PNG
```

^X

Comments: For more information on ZB64 encoding and compression, see ZB64 Encoding and Compression on page 1579.

These are some important things to know about this command in firmware version V60.14.x, V50.14.x, or later:

- ZebraNet Bridge can be used to download fonts and graphics with this command.
- OpenType tables are only supported when downloading the font with this command
- OpenType fonts (`.0TF`) are supported if they are downloaded as a TrueType font. In the printer `.0TF` fonts have the `.TTF` extension.

~EG

Erase Download Graphics

See ^ID on page 245.

^FB

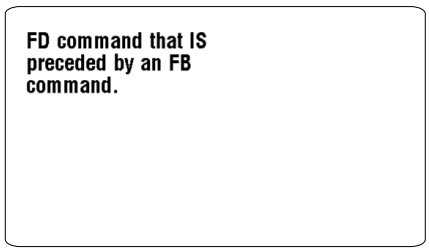
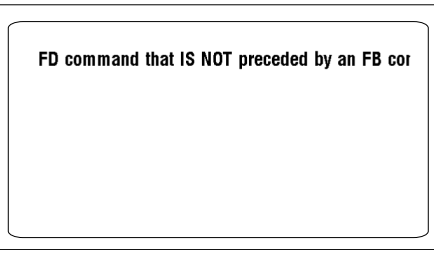
The ^FB command allows you to print text into a defined **block type** format. This command formats an ^FD or ^SN string into a block of text using the origin, font, and rotation specified for the text string. The ^FB command also contains an automatic word-wrap function.

Field Block

Format: ^FBa,b,c,d,e

Parameters	Details
a = width of text block line (in dots)	**Values:** 0 to the width of the label **Default:** 0 If the value is less than the font width or not specified, the text does not print.
b = maximum number of lines in the text block	**Values:** 1 to 9999 **Default:** 1 Text exceeding the maximum number of lines overwrites the last line. Changing the font size automatically increases or decreases the size of the block.
c = add or delete space between lines (in dots)	**Values:** -9999 to 9999 **Default:** 0 Numbers are considered to be positive unless preceded by a minus sign. Positive values add space; negative values delete space.
d = text justification	**Values:** L = left C = center R = right J = justified **Default:** L If J is used, the last line is left-justified.
e = hanging indent (in dots) of the second and remaining lines	**Values:** 0 to 9999**Default:** 0

Example: These are examples of how the ^FB command affects field data.

ZPL II CODE	GENERA TED LABEL
^XA ^CF0,30,30^FO25,50 ^FB250,4,, ^FDFD command that IS\& preceded by an FB \&command. ^FS ^XZ	
^XA ^CF0,30,30^FO25,50 ^FDFD command that IS NOT preceded by an FB command.^FS ^XZ	

Comments:

This scheme can be used to facilitate special functions:

\& = carriage return/line feed

\(*) = soft hyphen (word break with a dash)

\\ = backslash (\)

Item 1: ^CI13 must be selected to print a backslash (\).

Item 2: If a soft hyphen escape sequence is placed near the end of a line, the hyphen is printed. If it is not placed near the end of the line, it is ignored.

(*) = any alphanumeric character

- If a word is too long to print on one line by itself (and no soft hyphen is specified), a hyphen is automatically placed in the word at the right edge of the block. The remainder of the word is on the next line. The position of the hyphen depends on word length, not a syllable boundary. Use a soft hyphen within a word to control where the hyphenation occurs.
- Maximum data-string length is 3K, including control characters, carriage returns, and line feeds.
- Normal carriage returns, line feeds, and **word spaces** at line breaks are discarded.
- When using ^FT (Field Typeset), ^FT uses the baseline origin of the last possible line of text. Increasing the font size causes the text block to increase in size from bottom to top. This could cause a label to print past its top margin.
- When using ^F0 (Field Origin), increasing the font size causes the text block to increase in size from top to bottom.
- ^FS terminates an ^FB command. Each block requires its own ^FB command.

.14↑

While the ^FB command has a text justification parameter that defines the justification of the text within the block, it also interacts with the justification of ^F0 and ^FT that define the justification of the origin.

The ^FB command does not support soft hyphens as a potential line breakpoint. However, soft hyphen characters are always printed as if they were a hyphen.

The ^FB command does not support complex text. For complex text support, use ^TB.

^FC

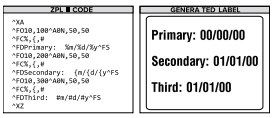
The ^FC command is used to set the clock indicators (delimiters) and the clock mode for use with the Real-Time Clock hardware. This command must be included within each label field command string each time the Real-Time Clock values are required within the field.

Field Clock

Format: ^FCa,b,c

Parameters	Details
a = primary clock indicator character	**Values:** any ASCII character **Default:** %
b = secondary clock indicator character	**Values:** any ASCII character **Default:** none—this value cannot be the same as a or c
c = third clock indicator character	**Values:** any ASCII character **Default:** none—this value cannot be the same as a or b

Entering these ZPL commands sets the primary clock indicator to %, the secondary clock indicator to {, and the third clock indicator to #. The results are printed on a label with Primary, Secondary, and Third as field data.



Comments: The ^FC command is ignored if the Real Time Clock hardware is not present. As of V60.13.0.10, (^SN) functions with (^FC) capabilities.

For more details on the Real Time Clock, see Real Time Clock on page 1590.

^FD

The ^FD command defines the data string for a field. The field data can be any printable character except those used as command prefixes (^ and ~).

Field Data

In RFID printers, it can also be used to specify passwords to write to tags.

Format: ^FDa

Parameters	Details
a = • data to be printed (all printers), or • a password to be written to a RFID tag (rfid printers)	**Values:** any data string up to 3072 bytes **Default:** none—a string of characters must be entered

Comments: The ^ and ~ characters can be printed by changing the prefix characters—see ^CD ~CD on page 153 and ^CT ~CT on page 166. The new prefix characters cannot be printed.

Characters with codes above 127, or the ^ and ~ characters, can be printed using the ^FH and ^FD commands.

- ^CI13 must be selected to print a backslash (\).

For information on using soft hyphens, see ^FB on page 186.

^FE

The ^FE command allows field data concatenation and substring extraction by referencing ^FN fields.

The ^FE command must precede each ^FD command where it is used and only applies to that ^FD field. If a ^FE does not immediately precede a ^FD, then there is no field concatenation character active for that ^FD.

Format: ^FEa

Parameters	Details
a	**Values:** Any character except the current format and control prefix (^ and ~ by default). **Default:** #

This delimiter, when defined, will be used to specify data from ^FN fields, or parts thereof, that will be inserted into a ^FD field. For the following description, it is assumed that the ^FE character is the default character, #. There are two ways to insert data: including an entire ^FN field and including part of a ^FN field.

To insert an entire ^FN field, the syntax placed in the ^FD field would be #n#, where n is the number of the ^FN. As an example:

```
^FN2^FDField FN 2 Data^FS
^FN3^FDField FN 3 Data^FS
^FE#^FD#2# and then #3#^FS
```

would result in the final ^FD with the data `Field FN2 Data and then Field FN3 Data`.

To insert part of a ^FN field, the syntax placed in the ^FD field would be #n,a,x,y#, where:

Parameters	Detail
n	The number of the ^FN from which the data is taken.
a either f or b	f (for forward) indicates that data is taken from the from the start of the ^FN, and b (for backward) indicates that data is taken from the end.
x	The starting position counts from the first or last character. 1 indicates start with the first character. If a is b, then x starts counting from the end of the data and counts backward, so 1 would be starting from the last character. 0 is invalid, and the field inserting description is ignored, which is also the case for negative numbers
y	The number of characters to take. If y is greater than the number of characters available, all the characters are taken from the starting position to the last character.

For the following examples, the first two lines of ZPL are assumed to be:

```
^FN2^FDField FN 2 Data^FS
^FN3^FDField FN 3 Data^FS
```

Example 1

```
^FE#^FD#2,f,1,5#^FS
```

Results in the data `Field`.

Example 2

```
^FE#^FD#2,f,7,4#^FS
```

results in the data `FN 2`

Example 3

```
^FE#^FD#2,b,1,4#^FS
```

Results in the data `Data`

Example 4

```
^FE#^FD#2# and #3,f,10,6#^FS
```

Results in the data `Field FN 2 Data and 3 Data`

Supported Zebra Printers

- ZD421C
- ZD421D
- ZD621D
- ZD621T
- ZT411
- ZT421
- ZT510
- ZT610
- ZT620

^FH

The ^FH command allows you to enter the hexadecimal value for any character directly into the ^FD statement. The ^FH command must precede each ^FD command that uses hexadecimal in its field.

Field Hexadecimal Indicator

Within the ^FD statement, the hexadecimal indicator must precede each hexadecimal value. The default hexadecimal indicator is _ (underscore). There must be a minimum of two characters designated to follow the underscore. The a parameter can be added when a different hexadecimal indicator is needed.

This command can be used with any of the commands that have field data (that is ^FD, ^FV (Field Variable), and ^SN (Serialized Data)).

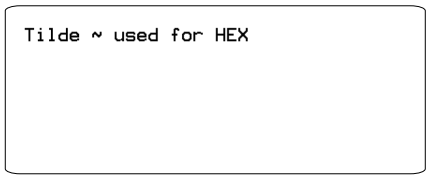
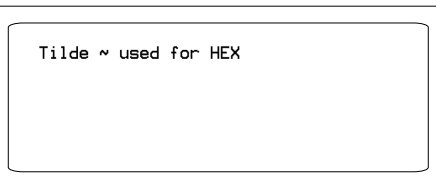
Valid hexadecimal characters are:

0 1 2 3 4 5 6 7 8 9 A B C D E F a b c d e f

Format: ^FHa

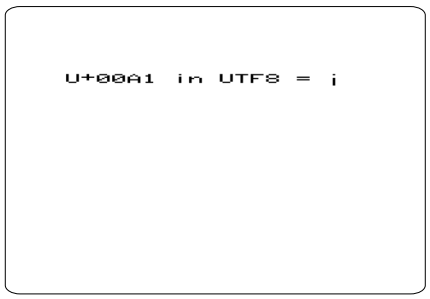
Parameters	Details
a = hexadecimal indicator	**Values:** any character except current format and control prefix (^ and ~ by default) **Default:** _ (underscore)

Example: This is an example of how to enter a hexadecimal value directly into a ^FD statement: This is an example of ASCII data using ^CI0.

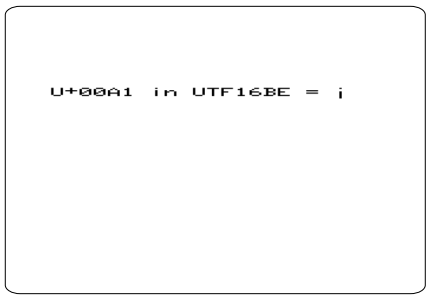
ZPL II CODE	GENERATED LABEL
^XA ^FO100,100 ^AD^FH ^FDTilde _7e used for HEX^FS ^XZ	
^XA ^FO100,100 ^AD^FH\ ^FDTilde \7E used for HEX^FS ^XZ	

Example: These are examples of how ^FH works with UTF-8 and UTF-16BE:

- UTF-8

ZPL II CODE	GENERATED LABEL
^XA ^CI28 ^LL500 ^FO100,100 ^AA,20,20 ^FH^FDU+00A1 in UTF8 = _C2_A1^FS ^XZ	

- UTF-16BE

ZPL II CODE	GENERATED LABEL
^XA ^CI29 ^LL500 ^FO100,100 ^AA,20,20 ^FH^FDU+00A1 in UTF16BE = _00_A1^FS ^XZ	

^FL

The ^FL command provides the ability to link any TrueType font, including private character fonts, to associated fonts.

Font Linking

.14↑

If the base font does not have a glyph for the required character, the printer looks to the linked fonts for the glyph. The font links are user-definable. The font linking remains until the link is broken or the printer is turned off. To permanently save the font linking, use the ^JUS command.

NOTE: For assistance in setting up the font links, use the font wizard in ZebraNet Bridge.

Format: ^FL<ext>,<base>,<link>

Parameters	Details
<ext>	This is the fully-qualified filename of the extension. This file name does not accept wildcards. The supported extensions for this parameter are: .TTF and .TTE. The format for this parameter is the memory device followed by the font name with the extension, as follows: E:SWISS721.TTF
<base>	This is the filename of the base font(s). The base font can be any of the following types: .FNT .TTF or .TTE From these font types, you can only link to a .TTF or TTE. The name of the base font can be expressed as a wild card; doing so will define multiple base fonts. The result will be that all base font files so defined will be linked to the file defined in the <ext> parameter. The filename does not have to match a file that is currently defined on the printer. A specification of *.TTF results in all *.TTF font files loaded on the printer currently or in the future to be linked with the specified <ext> font extension.
<link>	This is an indicator that determines if the extension is to be linked with the base, or unlinked from the base, as follows: **Values:** 0 = <ext> is to be unlinked (disassociated) from the file(s) specified in <base> 1 = <ext> is to be linked (associated) with the file(s) specified by <base> **Default:** must be an accepted value or it is ignored

Comments: A font can have up to five fonts linked to it. The printer's resident font, 0.FNT is always the last font in the list of font links, but is not included in the five-link maximum. It can also be placed anywhere in the font links list.

The default glyph prints when a glyph cannot be found in any of the fonts in the link list. The advanced layout command ^PA determines if the default glyph is a space character or the default glyph of the base font, which is typically a hollow box.

The list of font links can be printed by using the ^LF command or retrieved with the ^HT command.

Example: These examples show the code and output for no font linking and for font linking:

No Font Linking

In the no-font linking example, the Swiss721 font does not have Asian glyphs, which is why Asian glyphs do not print.

ZPL II CODE

```
^XA^LL1200^CW1,E:SWISS721.TTF^CW2,E:ANMDJ.TTF^CI28^FS
^FO100,100^A0,50,50^FDNO FONT LINKING^FS
^FO100,300^A1,50,50^FDTEST WITH SWISS721^FS
^FO100,400^A1,50,50^FDDRAGONFLY 蜻蜓^FS
^FO100,600^A2,50,50^FDTEST WITH ANMDJ^FS
^FO100,700^A2,50,50^FDDRAGONFLY 蜻蜓^FS
^XZ
```

GENERATED LABEL

Font Linking

In the font linking example, this code is sent down to link the ANMDJ.TTF font to SWISS721.TTF font:

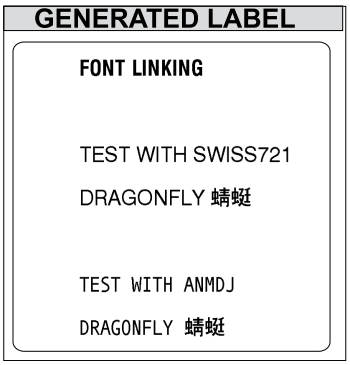
```
^XA

^FLE:ANMDJ.TTF,E:SWISS721.TTF,1^FS
^XZ
```

When the label prints, the Asian characters are printed using the ANMDJ.TTF font, rather than the SWISS721.TTF font.

ZPL II CODE

```
^XA^LL1200^CW1,E:SWISS721.TTF^CW2,E:ANMDJ.TTF^CI28^FS
^FO100,100^A0,50,50^FDFONT LINKING^FS
^FO100,300^A1,50,50^FDTEST WITH SWISS721^FS
^FO100,400^A1,50,50^FDDRAGONFLY 蜻蜓^FS
^FO100,600^A2,50,50^FDTEST WITH ANMDJ^FS
^FO100,700^A2,50,50^FDDRAGONFLY 蜻蜓^FS
^XZ
```

GENERATED LABEL



^FM

The ^FM command allows you to control the placement of bar code symbols.

Multiple Field Origin Locations

It designates field locations for the PDF417 (^B7) and MicroPDF417 (^BF) bar codes when the structured append capabilities are used. This allows printing multiple bar codes from the same set of text information.

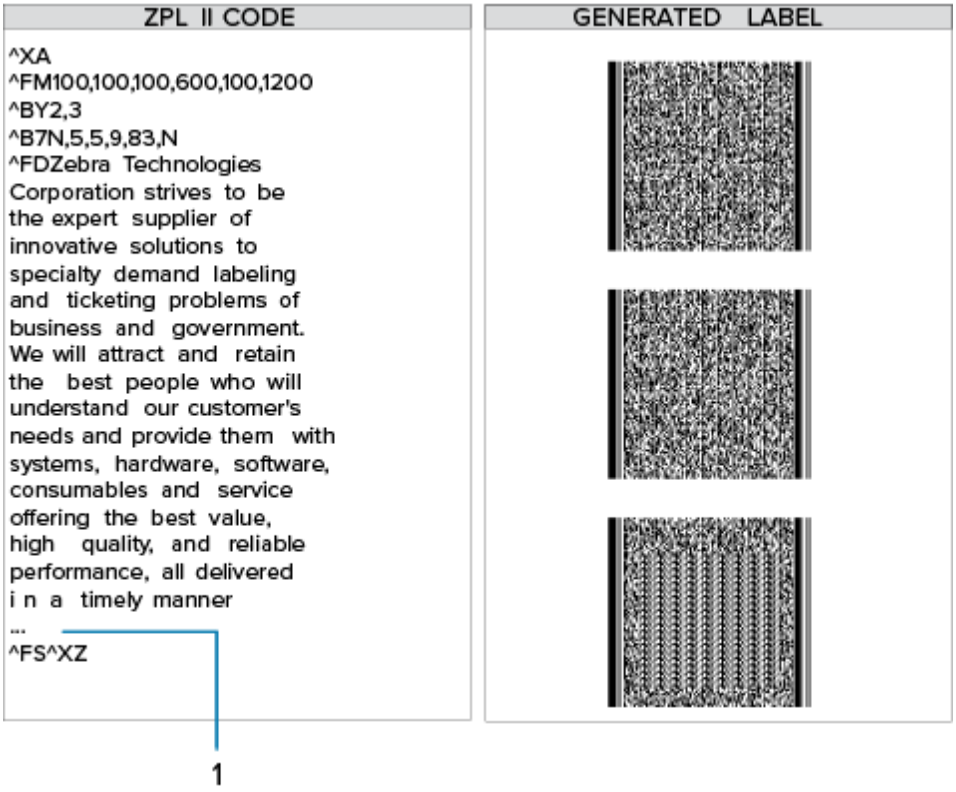
The structured append capability is a way of extending the text printing capacity of both bar codes. If a string extends beyond what the data limitations of the bar code are, it can be printed as a series: 1 of 3, 2 of 3, 3 of 3. Scanners read the information and reconcile it into the original, unsegmented text.

The ^FM command triggers multiple bar code printing on the same label with ^B7 and ^BF only. When used with any other commands, it is ignored.

Format:^FMx1,y1,x2,y2,...

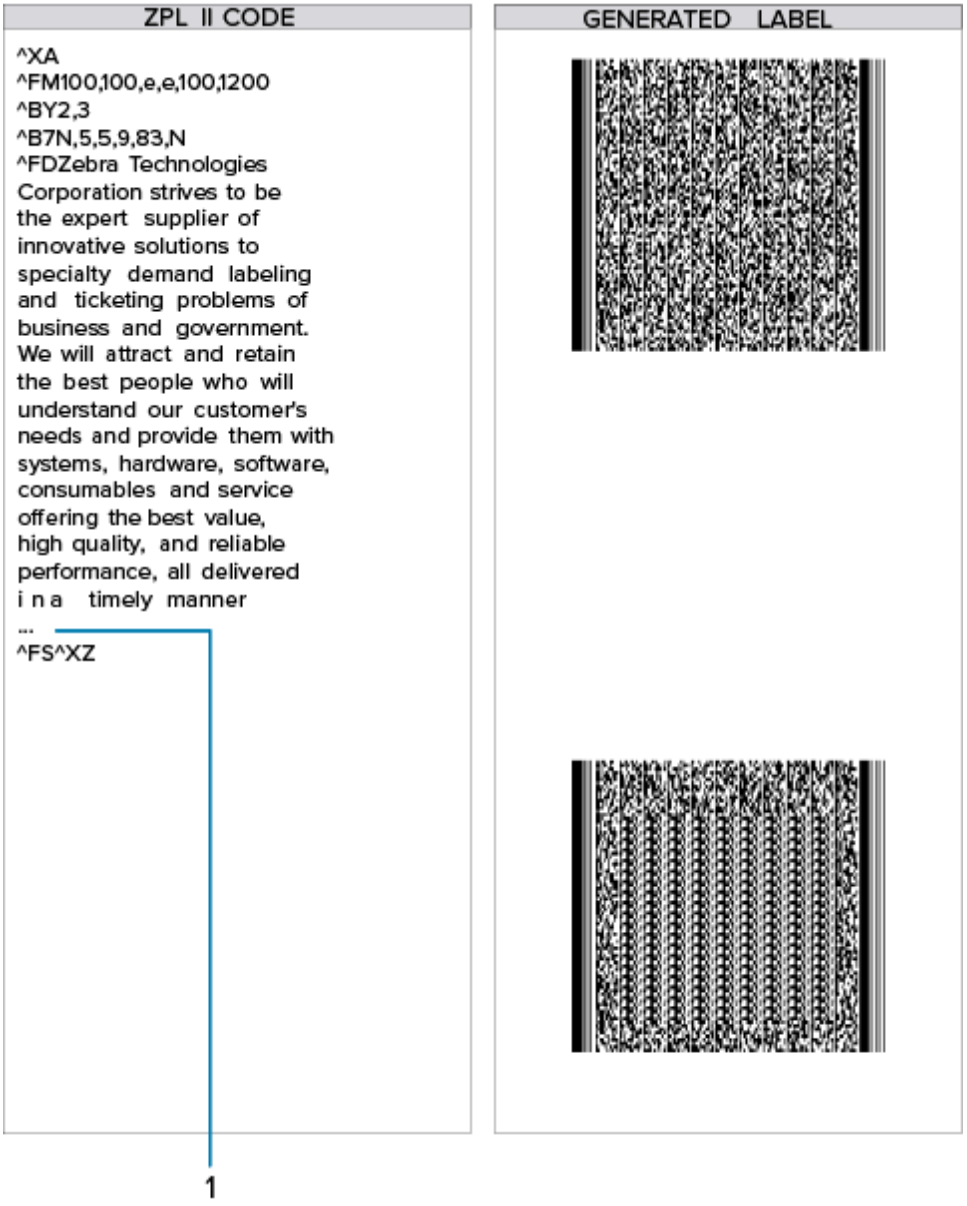
Parameters	Details
x1 = x-axis location of first symbol (in dots)	**Values:** 0 to 32000e = exclude this bar code from printing **Default:** a value must be specified a value must be specified
y1 = y-axis location of first symbol (in dots)	**Values:** 0 to 32000e = exclude this bar code from printing **Default:** a value must be specified
x2 = x-axis location of second symbol (in dots)	**Values:** 0 to 32000e = exclude this bar code from printing **Default:** a value must be specified
y2 = y-axis location of second symbol (in dots)	**Values:** 0 to 32000e = exclude this bar code from printing **Default:** a value must be specified
... = continuation of X,Y pairs	**Maximum number of pairs:** 60

Example: This example shows you how to generate three bar codes with the text “Zebra Technologies Corporation strives to be...” would need to be repeated seven times, which includes 2870 characters of data (including spaces) between ^FD and ^FS:



1	The ellipse is not part of the code. It indicates that the text needs to be repeated seven times, as mentioned in the example description.

Example: This example assumes a maximum of three bar codes, with bar code 2 of 3 omitted:



Comments: Subsequent bar codes print once the data limitations of the previous bar code have been exceeded. For example, bar code 2 of 3 prints once 1 of 3 has reached the maximum amount of data it can hold. Specifying three fields does not ensure that three bar codes print; enough field data to fill three bar code fields has to be provided.

The number of the `x,y` pairs can exceed the number of bar codes generated. However, if too few are designated, no symbols print.

^FN

The ^FN command numbers the data fields. This command is used in both ^DF (Store Format) and ^XF (Recall Format) commands.

Field Number

In a stored format, use the ^FN command where you would normally use the ^FD (Field Data) command. In recalling the stored format, use ^FN in conjunction with the ^FD command.

The optional "a" parameter can be used with the KDU Plus to cause prompts to be displayed on the KDU unit. Also, when the Print on Label link is selected on the Directory page of ZebraLink enabled printers the field prompt displays.

The number of fields and data that can be stored is dependent in the available printer memory.

NOTE: The maximum number of ^FN commands that can be used depends on the amount of data that is placed in the fields on the label. It is recommended to use 400 or fewer fields.

Format: ^FN#"a"

Parameters	Details
# = number to be assigned to the field	**Values:** 0 to 9999 **Default:** 0
"a" = optional parameter*	**Values:** 255 alphanumeric characters maximum (a-z,A-Z,1-9 and space) **Default:** optional parameter
* This parameter is only available on printers with firmware V50.13.2, V53.15.5Z, V60.13.0.1, or later. For a complete example of the ^DF and ^XF command, see Exercise 6: ^DF and ^XF - Download Format and Recall Format.	

Comments:

- The same ^FN value can be stored with several different fields.
- If a label format contains a field with ^FN and ^FD, the data in that field prints for any other field containing the same ^FN value.
- For the "a" parameter to function as a prompt the characters used in the "a" parameter must be surrounded by double quotes (see example).

The ^FN1"Name" would result in "Name" being used as the prompt on the KDU unit.

^FO

The ^F0 command sets a field origin, relative to the label home (^LH) position. ^F0 sets the upper-left corner of the field area by defining points along the x-axis and y-axis independent of the rotation.

Field Origin

Format: ^F0x,y,z

Parameters	Details
x = x-axis location (in dots)	**Values:** 0 to 32000 **Default:** 0
y = y-axis location (in dots)	**Values:** 0 to 32000 **Default:** 0
z = justification .14↑ The z parameter is only supported in firmware versions V60.14.x, V50.14.x, or later.	**Values:** 0 = left justification 1 = right justification 2 = auto justification (script dependent) **Default:** last accepted ^FW value or ^FW default

Comments: If the value entered for the x or y parameter is too high, it could position the field origin completely off the label.

.14↑

This command interacts with the field direction parameter of ^FP and with the rotation parameter of ^A. For output and examples, see Field Interactions on page 1583.

The auto justification option might cause unexpected results if variable fields or bidirectional text are used with ^F0. For the best results with bidirectional text and/or variable fields, use either the left of right justification option.

^FP

The ^FP command allows vertical and reverse formatting of the font field, commonly used for printing Asian fonts.

Field Parameter

Format: ^FPd,g

Parameters	Details
d = direction	**Values:** H = horizontal printing (left to right) V = vertical printing (top to bottom) R = reverse printing (right to left) **Default:** H
g = additional inter-character gap (in dots)	**Values:** 0 to 9999 **Default:** 0 if no value is entered

Example: This is an example of how to implement reverse and vertical print:

ZPL II CODE	GENERATED LABEL
^XA ^FO100,50 ^FPV,10 ^AV ^FDvertical^FS ^XZ	v e r t i c a l
^XA ^FO350,50 ^FPR,10 ^AV ^FDreverse^FS ^XZ	esrever

.14↑

For vertical and reverse printing directions, combining semantic clusters are used to place characters.

This command interacts with the justification parameters of ^F0 and ^FT and with the rotation parameter of ^A. For output and examples, see Field Interactions on page 1583.

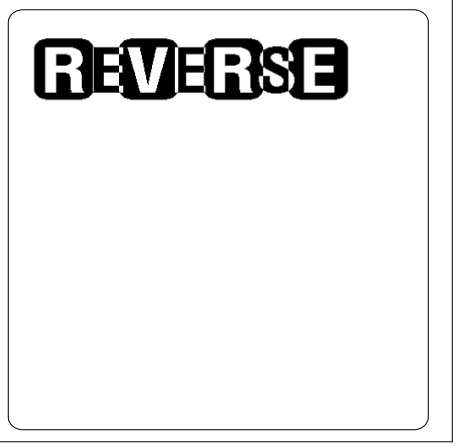
^FR

The ^FR command allows a field to appear as white over black or black over white. When printing a field and the ^FR command has been used, the color of the output is the reverse of its background.

Field Reverse Print

Format: ^FR

Example: In this example, the ^GB command creates areas of black, allowing the printing to appear white:

ZPL II CODE	GENERA TED LABEL
^XA ^PR1 ^FO100,100 ^GB70,70,70,,3^FS ^FO200,100 ^GB70,70,70,,3^FS ^FO300,100 ^GB70,70,70,,3^FS ^FO400,100 ^GB70,70,70,,3^FS ^FO107,110^CF0,70,93 ^FR^FDREVERSE^FS ^XZ	

Comments: The ^FR command applies to only one field and has to be specified each time. When multiple ^FR commands are going to be used, it might be more convenient to use the ^LR command.

^FS

The ^FS command denotes the end of the field definition. Alternatively, ^FS command can also be issued as a single ASCII control code SI (Control-O, hexadecimal 0F).

Field Separator

Format: ^FS

NOTE: It is recommended to place an ^FS after every command that creates a printable line.

^FT

The ^FT command sets the field position, relative to the home position of the label designated by the ^LH command. The typesetting origin of the field is fixed with respect to the contents of the field and does not change with rotation.

Field Typeset

NOTE: The ^FT command is capable of the concatenation of fields.

Format: ^FTx,y,z

Parameters	Details
x = x-axis location (in dots)	**Values:** 0 to 32000 **Default:** position after the last formatted text field
y = y-axis location (in dots)	**Values:** 0 to 32000 **Default:** position after the last formatted text field
z = justification .14↑ The z parameter is only supported in firmware version V60.14.x, V50.14.x, or later.	**Values:** 0 = left justification 1 = right justification 2 = auto justification (script dependent) **Default:** last accepted ^FW value or ^FW default The auto-justification option may cause unexpected results if variable fields or bidirectional text are used with ^FT. For best results with bidirectional text and/or variable fields, use either the left or right justification options.

Table 7 Typeset Justification

Left Justified	Text	For examples, see Field Interactions on page 1583.
	Bar Codes	Origin is the base of barcode, at the left edge
	Graphic Boxes	The origin is bottom-left corner of the box
	Images	Origin is bottom-left corner of the image area
Right Justified	Text	For examples, see Field Interactions on page 1583.
	Bar Codes	Origin is the base of barcode, at the right edge
	Graphic Boxes	Origin is bottom-right corner of the box
	Images	Origin is bottom-right corner of the image area

Example: This is an example of the ^FT command and concatenation:

ZPL II CODE	GENERATED LABEL
^XA ^FT10,200^A0N,30,20^FDACME ™ ^FS ^FT^GS^FDC^FS ^FT^A0N,30,20^FDSummer ^FS ^FT^A0N,60,50^FDClearance ^FS ^FT^A0N,120,100^FDSale ^FS ^XZ	

When a coordinate is missing, the position following the last formatted field is assumed. This **remembering** simplifies field positioning with respect to other fields. Once the first field is positioned, other fields follow automatically.

There are several instances where using the ^FT command without specifying x and y parameters is not recommended:

- when positioning the first field in a label format
- at any time with the ^FN (Field Number) command
- following an ^SN (Serialization Data) command
- variable data
- bidirectional text

.14↑

The right typeset justified is available only for printers with firmware version V60.14.x, V50.14.x, or later.

This command interacts with the field direction parameters of ^FP and with the rotation parameter of ^A. For output and code examples, see Field Interactions on page 1583.

^FV

^FV replaces the ^FD (field data) command in a label format when the field is variable.

Field Variable

Format: ^FVa

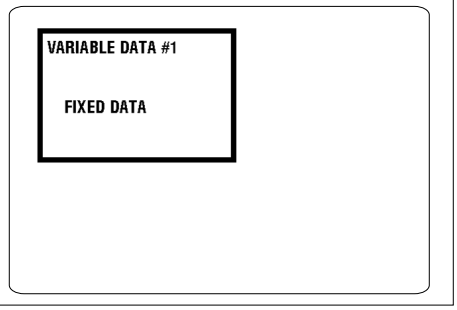
Parameters	Details
a = variable field data to be printed	**Values:** 0 to 3072 byte string **Default:** if no data is entered, the command is ignored

Example: This is an example of how to use the ^MC and ^FV command:

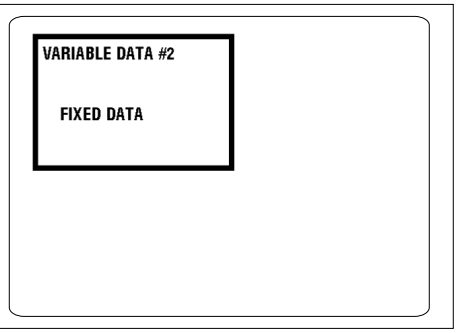
ZPL II CODE

```
^XA
^FO40,40
^GB300,203,8^FS
^FO55,60^CF0,25
^FVVARIABLE  DATA  #1^FS
^FO80,150
^FDFIXED  DATA^FS
^MCN
^XZ
```

GENERATED LABEL




```
^XA
^FO55,60 ^CF0,25
^FVVARIABLE  DATA  #2^FS
^MCY
^XZ
```



Comments: ^FV fields are always cleared after the label is printed. ^FD fields are not cleared.

^FW

The ^FW command sets the default orientation for all command fields that have an orientation (rotation) parameter (and in x.14 sets the default justification for all commands with a justification parameter). Fields can be rotated 0, 90, 180, or 270 degrees clockwise by using this command. In x.14, justification can be left, right, or auto.

Field Orientation

The ^FW command affects only fields that follow it. Once you have issued a ^FW command, the setting is retained until you turn off the printer or send a new ^FW command to the printer.

Format: ^FWr,z

Parameters	Details
r = rotate field	**Values:** N = normal R = rotated 90 degrees I = inverted 180 degrees B = bottom-up 270 degrees, read from the bottom up **Initial Value at Power Up:** N
z = justification .14↑ The z parameter is available only with printers with firmware version V60.14.x, V50.14.x, or later.	**Values:** 0 = left justification 1 = right justification 2 = auto justification (script dependent) **Default:** auto for ^TB and left for all other commands

Example: This example shows how ^FW rotation works in conjunction with ^F0. In this example, note that:

- the fields using A0N print the field in normal rotation
- the fields with no rotation indicated (A0) follow the rotation used in the ^FW command (^FWR).

ZPL II CODE	GENERATED LABEL
^XA ^FWR ^FO150,90^A0N,25,20^FDZebra Technologies^FS ^FO115,75^A0,25,20^FD0123456789^FS ^FO150,115^A0N,25,20^FD333 Corporate Woods Parkway^FS ^FO400,75^A0,25,20^FDXXXXXXXXX^FS ^XZ	

Comments: ^FW affects only the orientation in commands where the rotation parameter has not been specifically set. If a command has a specific rotation parameter, that value is used.

.14↑

^FW affects only the justification in commands where the parameter has not been set. If a command has a specific justification parameter, that value is used.

^FX

The ^FX command is useful when you want to add **non-printing** informational comments or statements within a label format. Any data after the ^FX command up to the next caret (^) or tilde (~) command does not have any effect on the label format. Therefore, you should avoid using the caret (^) or tilde (~) commands within the ^FX statement.

Comment

Format: ^FXc

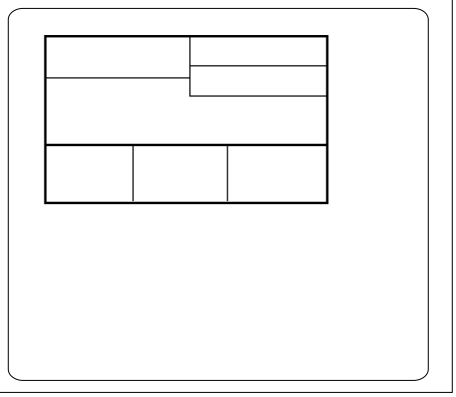
Parameters	Details
c = non-printing comment	Creates a non-printable comment.

Example: This is an example of how to use the ^FX command effectively:

ZPL II CODE

```
^XA
^LH100,100^FS
^FXSHIPPING  LABEL^FS
^FO10,10^GB470,280,4^FS
^FO10,190^GB470,4,4^FS
^FO10,80^GB240,2,2^FS
^FO250,10^GB2,100,2^FS
^FO250,110^GB226,2,2^FS
^FO250,60^GB226,2,2^FS
^FO156,190^GB2,95,2^FS
^FO312,190^GB2,95,2^FS
^XZ
```

GENERA TED LABEL



Comments: Correct usage of the ^FX command includes following it with the ^FS command.

^GB

The ^GB command is used to draw boxes and lines as part of a label format. Boxes and lines are used to highlight important information, divide labels into distinct areas, or improve the appearance of a label. The same format command is used for drawing either boxes or lines.

Graphic Box

Format: ^GBw,h,t,c,r

Parameters	Details
w = box width (in dots)	**Values:** value of t to 32000 **Default:** value used for thickness (t) or 1
h = box height (in dots)	**Values:** value of t to 32000 **Default:** value used for thickness (t) or 1
t = border thickness (in dots)	**Values:** 1 to 32000 **Default:** 1
c = line color	**Values:** B = black W = white **Default:** B
r = degree of corner-rounding	**Values:** 0 (no rounding) to 8 (heaviest rounding) **Default:** 0

For the w and h parameters, keep in mind that printers have a default of 6, 8, 12, or 24 dots/millimeter. This comes out to 153, 203, 300, or 600 dots per inch. To determine the values for w and h, calculate the dimensions in millimeters and multiply by 6, 8, 12, or 24.

If the width and height are not specified, you get a solid box with its width and height as specified by the value t.

The roundness index is used to determine a rounding radius for each box. Formula:

rounding-radius = (rounding-index / 8) * (shorter side / 2)

where the shorter side is the lesser of the width and height (after adjusting for minimum and default values).

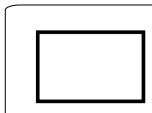
Example: Here are a few examples of graphic boxes:

Width: 1.5 inch; Height: 1 inch; Thickness: 10; Color: default; Rounding: default

ZPL II CODE

```
^XA
^FO50,50
^GB300,200,10^FS
^XZ
```

GENERA TED LABEL



Width: 0 inch; Height: 1 inch; Thickness: 20; Color: default; Rounding: default:

ZPL II CODE	GENERATED LABEL
^XA ^FO50,50 ^GB0,203,20^FS ^XZ	

Width: 1 inch; Height: 0 inch; Thickness: 30; Color: default; Rounding: default

ZPL II CODE	GENERA TED LABEL
^XA ^FO50,50 ^GB203,0,20^FS ^XZ	

Width: 1.5 inch; Height: 1 inch; Thickness: 10; Color: default; Rounding: 5

ZPL II CODE	GENERA TED LABEL
^XA ^FO50,50 ^GB300,200,10,,5^FS ^XZ	

^GC

The ^GC command produces a circle on the printed label. The command parameters specify the diameter (width) of the circle, outline thickness, and color. Thickness extends inward from the outline.

Graphic Circle

Format: ^GCd,t,c

Parameters	Details
d = circle diameter (in dots)	**Values:** 3 to 4095 (larger values are replaced with 4095) **Default:** 3
t = border thickness (in dots)	**Values:** 2 to 4095 **Default:** 1
c = line color	**Values:** B = black W = white **Default:** B

Example

This is an example of how to create a circle on the printed label:

ZPL II CODE	GENERA TED LABEL
^XA ^FO50,50 ^GC250,10,B^FS ^XZ	

^GD

The ^GD command produces a straight diagonal line on a label. This can be used in conjunction with other graphic commands to create a more complex figure.

Graphic Diagonal Line

Format: ^GDw,h,t,c,o

Parameters	Details
w = box width (in dots)	**Values:** 3 to 32000 **Default:** value of t (thickness) or 3
h = box height (in dots)	**Values:** 3 to 32000 **Default:** value of t (thickness) or 3
t = border thickness (in dots)	**Values:** 1 to 32000 **Default:** 1
c = line color	**Values:** B = black W = white **Default:** B
o = orientation (direction of the diagonal)	**Values:** R (or /) = right-leaning diagonal (or \) = left-leaning diagonal**Default:** R

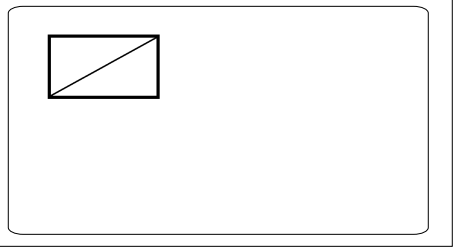
Example

This is an example of how to create a diagonal line connecting one corner with the opposite corner of a box on a printed label:

ZPL II CODE

```
^XA
^FO150,100
^GB350,203,10^FS
^FO155,110
^GD330,183,10,,R^FS
^XZ
```

GENERA TED LABEL



^GE

The ^GE command produces an ellipse in the label format.

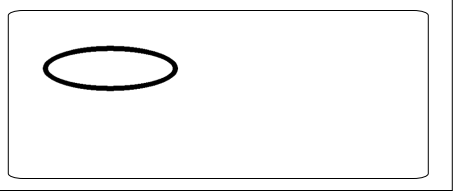
Graphic Ellipse

Format: ^GEw,h,t,c

Parameters	Details
w = ellipse width (in dots)	**Values:** 3 to 4095 (larger values are replaced with 4095) **Default:** the value used for thickness (t) or 1
h = ellipse height (in dots)	**Values:** 3 to 4095 **Default:** the value used for thickness (t) or 1
t = border thickness (in dots)	**Values:** 2 to 4095 **Default:** 1
c = line color	**Values:** B = black W = white **Default:** B

Example

This is an example of how to create an ellipse on a printed label:

ZPL II CODE	GENERA TED LABEL
^XA ^FO100,100 ^GE300,100,10,B^FS ^XZ	

^GF

The ^GF command allows you to download graphic field data directly into the printer's bitmap storage area. This command follows the conventions for any other field, meaning a field orientation is included. The graphic field data can be placed at any location within the bitmap space.

Graphic Field

Format: ^GFa,b,c,d,data

Parameters	Details
a = compression type	**Values:** A = ASCII hexadecimal (follows the format for other download commands) B = binary (data sent after the c parameter is strictly binary) C = compressed binary (data sent after the c parameter is in compressed binary format. The data is compressed on the host side using Zebra's compression algorithm. The data is then decompressed and placed directly into the bitmap.) **Default:** A
b = binary byte count	**Values:** 1 to 99999 This is the total number of bytes to be transmitted for the total image or the total number of bytes that follow parameter d. For ASCII download, the parameter should match parameter c. Out-of-range values are set to the nearest limit. **Default:** command is ignored if a value is not specified
c = graphic field count	**Values:** 1 to 99999 This is the total number of bytes comprising the graphic format (width x height), which is sent as parameter d. Count divided by bytes per row gives the number of lines in the image. This number represents the size of the image, not necessarily the size of the data stream (see d). **Default:** command is ignored if a value is not specified
d = bytes per row	**Values:** 1 to 99999 This is the number of bytes in the downloaded data that comprise one row of the image. **Default:** command is ignored if a value is not specified
data = data	**Values:** **ASCII hexadecimal data:** 00 to FF A string of ASCII hexadecimal numbers, two digits per image byte. CR and LF can be inserted as needed for readability. The number of two-digit number pairs must match the above count. Any numbers sent after count is satisfied are ignored. A comma in the data pads the current line with **00** (white space), minimizing the data sent. ~DN or any caret or tilde character prematurely aborts the download. **Binary data:** Strictly binary data is sent from the host. All control prefixes are ignored until the total number of bytes needed for the graphic format is sent.

Example: This example downloads 8,000 total bytes of data and places the graphic data at location 100,100 of the bitmap. The data sent to the printer is in ASCII form.

```
^FO100,100^GFA,8000,8000,80,ASCII data
```

Example: This example downloads 8,000 total bytes of data and places the graphic data at location 100,100 of the bitmap. The data sent to the printer is in binary form.

```
^FO100,100^GFB,8000,8000,80,Binary data
```

^GS

The ^GS command enables you to generate the registered trademark, copyright symbol, and other symbols.

Graphic Symbol

Format: ^GSo,h,w

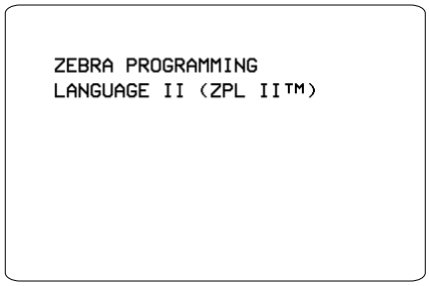
Parameters	Details
o = field orientation	**Values:** N = normalR = rotate 90 degrees clockwiseI = inverted 180 degreesB = bottom-up, 270 degrees**Default:** N or last ^FW value
h = character height proportional to width (in dots)	**Values:** 0 to 32000 **Default:** last ^CF value
w = character width proportional to height (in dots)	**Values:** 0 to 32000 **Default:** last ^CF value

Use the ^GS command followed by ^FD and the appropriate character (A through E) within the field data to generate the desired character:

ZPL II CODE

```
^XA^CFD
^FO50,50
^FDZEBRA  PROGRAMMING^FS
^FO50,75
^FDLANGUAGE  II (ZPL  II )^FS
^FO280,75
^GS^FDC^FS
^XZ
```

GENERA TED LABEL



A = ® (Registered Trade Mark)
B = © (Copyright)
C = ™ (Trade Mark)
D = (Underwriters Laboratories approval)
E = (Canadian Standards Association approval)

~HB

When the ~HB command is sent to the printer, a data string is sent back to the host. The string starts with an <STX> control code sequence and terminates by an <ETX><CR><LF> control code sequence.

Battery Status

NOTE: This command only responds to mobile printers.

Format: ~HB

Parameters

When the printer receives the command, it returns:

```
<STX>hh.hh,bb.bb,bt<ETX><CR><LF>
```

`<STX>`	=	ASCII start-of-text character
`hh.hh`	=	current head voltage reading in integers
`bb.bb`	=	current battery voltage reading in integers
`bt`	=	battery temperature in Celsius
`<ETX>`	=	ASCII end-of-text character
`<CR>`	=	ASCII carriage return
`<LF>`	=	ASCII line feed character

- This command is used for the power-supply battery of the printer and should not be confused with the battery backed-up RAM.
- For a more precise voltage reading, you can use the power.voltage SGD command, which returns a value to the nearest hundredths of a volt (X.XX).

~HD

The ~HD command echoes printer status information that includes the power supply and head temperature using the terminal emulator.

Head Diagnostic

Format: ~HD

Example: This is an example of the ~HD command:

```
Head Temp = 29
Ambient Temp = 00
Head Test = Passed
Darkness Adjust = 23
Print Speed = 2
Slew Speed = 6
Backfeed Speed = 2
Static_pitch_length = 0521
Dynamic_pitch_length = 0540
Max_dynamic_pitch_length = 0540
Min_dynamic_pitch_length = 0537
COMMAND PFX = ~ : FORMAT PFX = ^ : DELIMITER = ,
P30 INTERFACE = None
P31 INTERFACE = None
P32 INTERFACE = Front Panel         Revision 5
P33 INTERFACE = None
P34 INTERFACE = None
P35 INTERFACE = None
Dynamic_top_position = 0008

No ribbon A/D = 0000
```

^HF

The ^HF command sends stored formats to the host.

Host Format

Format: ^HFd,o,x

Parameters	Details
d = device to recall image	**Values:** R:, E:, B:, and A: **Default:** R:
o = image name	**Values:** 1 to 8 alphanumeric characters **Default:** if a name is not specified, UNKNOWN is used
x = extension	**Fixed Value:** .ZPL

Example: This example shows the sequence and results.

Using a terminal emulator, you download this code to the printer:

```
^XA
^DFB:FILE1.ZPL
^FO100,100^A0,100
^FDTEST^FS
^XZ
```

Then you send this code to the printer:

```
^XA
^HFB:FILE1.ZPL
^XZ
```

The terminal emulator returns this code:

```
^XA^DFFILE1,
^FO100,100^A0,100^FDTEST^FS
^XZ
```

^HG

The ^HG command is used to upload graphics to the host. The graphic image can be stored for future use, or it can be downloaded to any Zebra printer.

Host Graphic

Format: ^HGd:o.x

Parameters

- d (device location of object)- Values=R:, E:, B:, and A:;

Default= search priority

- o (object name)- Values=R:, E:, B:, and A;

Default =if a name is not specified, UNKNOWN is used

- x (extension)- Fixed Value=.GRF

Comments: For more information on uploading graphics, see .

^HH

The ^HH command echoes printer configuration back to the host using a terminal emulator.

Configuration Label Return

Format: ^HH

Example: This is an example of what is returned to the host when ^XA^HH^XZ is sent to the printer:

```
+10                     DARKNESS
+000                    TEAR OFF
TEAR OFF                PRINT MODE
NON-CONTINUOUS          MEDIA TYPE
WEB                     SENSOR TYPE
DIRECT-THERMAL          PRINT METHOD
050 6/8 MM              PRINT  WIDTH
0622                    LABEL LENGTH
22.0IN   557MM          MAXIMUM LENGTH
9600                    BAUD
8 BITS                  DATA BITS
NONE                    PARITY
XON/XOFF                HOST HANDSHAKE
NONE                    PROTOCOL
000                     NETWORK ID
NORMAL MODE             COMMUNICATIONS
<~>  7EH                CONTROL PREFIX
<^>  5EH                FORMAT PREFIX
<,>  2CH                DELIMITER CHAR
ZPL II                  ZPL MODE
NO MOTION               MEDIA POWER UP
NO MOTION               HEAD CLOSE
DEFAULT                 BACKFEED
+000                    LABEL TOP
+0000                   LEFT POSITION
026                     WEB S.
068                     MEDIA S.
050                     MARK S.
001                     MARK MED S.
CS                      MODES ENABLED
..                      MODES DISABLED
864 8/MM FULL           RESOLUTION
V32.10.2    <-          FIRMWARE
V2.2.6.98.A             HARDWARE ID
CUSTOMIZED              CONFIGURATION
1024............R:      RAM
8192............B:      MEMORY CARD
0768............E:      ONBOARD FLASH
NONE                    FORMAT CONVERT
NONE                    OPTION
05/14/03                RTC DATE
02:23                   RTC TIME
DYNAMIC                 IP RESOLUTION
ALL                     IP PROTOCOL
010.003.005.090         IP ADDRESS
255.255.255.000         SUBNET MASK
010.003.005.001         DEFAULT GATEWAY
```

~HI

The ~HI command is designed to be sent from the host to the Zebra printer to retrieve information. Upon receipt, the printer responds with information on the model, software version, dots-per-millimeter setting, memory size, and any detected options.

Host Identification

Format: ~HI

When the printer receives this command, it returns:

```
XXXXXX,V1.0.0,dpm,000KB,X
```

XXXXXX = model of Zebra printer

V1.0.0 = version of software

dpm = dots/mm

6, 8, 12, or 24 dots/mm printheads

000KB = memory

512KB = 1/2 MB

1024KB = 1 MB

2048KB = 2 MB

4096KB = 4 MB

8192KB = 8 MB

x = recognizable options

only options specific to printer are shown (cutter, options, et cetera.)

~HM

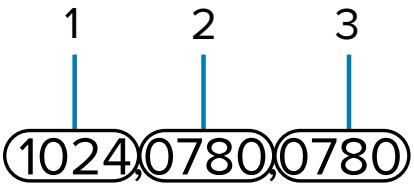
Sending ~HM to the printer immediately returns a memory status message to the host. Use this command whenever you need to know the printer's RAM status.

Host RAM Status

When ~HM is sent to the Zebra printer, a line of data containing information on the total amount, maximum amount, and available amount of memory is sent back to the host.

Format: ~HM

Example: This example shows when the ~HM is sent to the printer, a line of data containing three numbers are sent back to the host. Each set of numbers is identified and explained in the table that follows:



1	The total amount of RAM (in kilobytes) installed in the printer. In this example, the printer has 1024K RAM installed.
2	The maximum amount of RAM (in kilobytes) available to the user. In this example, the printer has a maximum of 780K RAM available.
3	The amount of RAM (in kilobytes) currently available to the user. In this example, there is 780K of RAM in the printer currently available to the user.

Comments: Memory taken up by bitmaps is included in the currently available memory value (due to ^MCN).

Downloading a graphic image, fonts, or saving a bitmap affects only the amount of RAM. The total amount of RAM and maximum amount of RAM does not change after the printer is turned on.

~HQ

The ~HQ command group causes the printer to send information back to the host.

Host Query

Format: ~HQquery-type

Parameter	Details
query-type	For detailed examples of these parameters, see ~HQ Examples on page 228. **Values:** • ES = requests the printer's status - see Table 8 Error Flags (~HQES) on page 226 and Table 9 Warning Flags (~HQES) on page 227 • HA = hardware address of the internal wired print server • JT = requests a summary of the printer's printhead test results • MA = maintenance alert settings • MI = maintenance information • OD = odometer • PH = printhead life history • PP = printer's Plug and Play string • SN = printer's serial number • UI = USB product ID and BDC release version • **Default:** must be an accepted value or the command is ignored

Comments: The response to the ~HQ command starts with STX, a CR LF is inserted between each line, and the response ends with ETX.

Table 8 Error Flags (~HQES)

Error Flags	Flag	Group 2	Group 1 (X = Value can be any hexadecimal number [0-9, A-F])							
		Nibbles 16-9	Nibble 8	Nibble 7	Nibble 6	Nibble 5	Nibble 4	Nibble 3	Nibble 2	Nibble 1
No Error	0	00000000	0	0	0	0	0	0	0	0
Error Present	1	00000000	X	X	X	X	X	X	X	X
Printhead Thermistor Open	1	00000000	X	X	X	X	X	2	X	X
Invalid Firmware Config.	1	00000000	X	X	X	X	X	1	X	X
Printhead Detection Error	1	00000000	X	X	X	X	X	X	8	X
Bad Printhead Element	1	00000000	X	X	X	X	X	X	4	X
Motor Over Temperature	1	00000000	X	X	X	X	X	X	2	X

Table 8 Error Flags (^HQES) (Continued)

Error Flags	Flag	Group 2	Group 1 (X = Value can be any hexadecimal number [0-9, A-F])							
		Nibbles 16-9	Nibble 8	Nibble 7	Nibble 6	Nibble 5	Nibble 4	Nibble 3	Nibble 2	Nibble 1
Printhead Over Temperature	1	00000000	X	X	X	X	X	X	1	X
Cutter Fault	1	00000000	X	X	X	X	X	X	X	8
Head Open	1	00000000	X	X	X	X	X	X	X	4
Ribbon Out	1	00000000	X	X	X	X	X	X	X	2
Media Out	1	00000000	X	X	X	X	X	X	X	1
Clear Paper Path Failed[1]	1	00000000	X	X	X	X	8	X	X	X
Paper Feed Error[1]	1	00000000	X	X	X	X	4	X	X	X
Presenter Not Running[1]	1	00000000	X	X	X	X	2	X	X	X
Paper Jam during Retract[1]	1	00000000	X	X	X	X	1	X	X	X
Black Mark not Found[1]	1	00000000	X	X	X	8	X	X	X	X
Black Mark Calibrate Error[1]	1	00000000	X	X	X	4	X	X	X	X
Retract Function timed out[1]	1	00000000	X	X	X	2	X	X	X	X
Paused	1	00000000	X	X	X	1	X	X	X	X

1. This error flag is supported only on KR403 printers.

Table 9 Warning Flags (^HQES)

Error Flags	Flag	Group 2	Group 1 (X = Value can be any hexadecimal number [0-9, A-F])							
		Nibbles 16-9	Nibble 8	Nibble 7	Nibble 6	Nibble 5	Nibble 4	Nibble 3	Nibble 2	Nibble 1
No Warning	0	00000000	0	0	0	0	0	0	0	0
Warning Present	1	00000000	X	X	X	X	X	X	X	X
Paper-near-end Sensor[1]	1	00000000	X	X	X	X	X	X	X	8
Replace Printhead	1	00000000	X	X	X	X	X	X	X	4
Clean Printhead	1	00000000	X	X	X	X	X	X	X	2
Need to Calibrate Media	1	00000000	X	X	X	X	X	X	X	1
Sensor 1 (Paper before head)[1]	1	00000000	X	X	X	X	X	X	1	X
Sensor 2 (Black mark)[1]	1	00000000	X	X	X	X	X	X	2	X

Table 9 Warning Flags (^HQES) (Continued)

Error Flags	Flag	Group 2	Group 1 (X = Value can be any hexadecimal number [0-9, A-F])							
		Nibbles 16-9	Nibble 8	Nibble 7	Nibble 6	Nibble 5	Nibble 4	Nibble 3	Nibble 2	Nibble 1
Sensor 3 (Paper after head)[1]	1	00000000	X	X	X	X	X	X	4	X
Sensor 4 (loop ready)[1]	1	00000000	X	X	X	X	X	X	8	X
Sensor 5 (presenter)[1]	1	00000000	X	X	X	X	X	1	X	X
Sensor 6 (retract ready)[1]	1	00000000	X	X	X	X	X	2	X	X
Sensor 7 (in retract)[1]	1	00000000	X	X	X	X	X	4	X	X
Sensor 8 (at bin)[1]	1	00000000	X	X	X	X	X	8	X	X

1. This error flag is supported only on KR403 printers.

^HQ Examples

This section provides detail examples of all the available parameters.

Example: This example shows how to request the printer's status.

To request the printer's status, type ~HQES

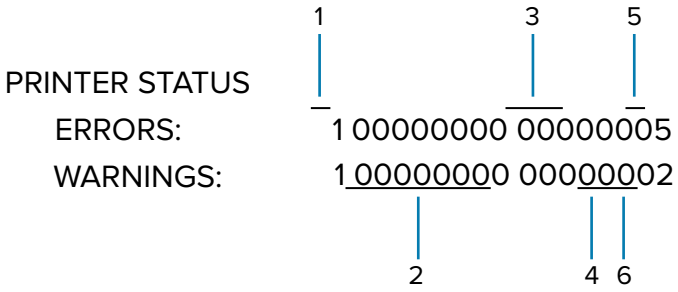
The printer responds with data similar to this:

```
PRINTER STATUS
ERRORS:         1 00000000 00000005
WARNINGS:       1 00000000 00000002
```

In this example, the Printer Status resolves to these conditions:

- The cover/printhead is open (value = 4).
- Media is out or not loaded into the printer (value = 1).
- The printhead needs to be cleaned (value = 2).
- Error nibble 1 is equal to 5 when the error status values are added together (4 + 1).

This illustration identifies the printer status definitions:



1	Flag

2	Nibble 16-9
3	Nibble 8-4
4	Nibble 3
5	Nibble 2
6	Nibble 1

Example: This example shows how the printer responds when the printer receives the ~HQES command:

To see how the printer responds, type ~HQES

The printer responds with data similar to this:

```
PRINTER STATUS
ERRORS:   1 00000000 0000000B
WARNINGS: 0 00000000 00000000
```

In this example, the printer status resolves to the following conditions:

- The cutter has a fault (value = 8).
- Ribbon is out or not loaded into the printer (value = 2).
- Media is out or not loaded into the printer (value = 1).
- Error byte 1 is equal to B when the error status values are added together (`8 + 2 + 1 = hexadecimal B`).

Example: This is an example of how to retrieve the hardware address of the internal wired print server.

To get the hardware address of the internal wired print server, type ~HQHA

The printer responds with data similar to this:

```
MAC ADDRESS
00:07:4d:2c:e0:7a
```

Example: This is an example of how to request a summary of the printer's printhead test results.

The ^JT command is used to initiate printhead testing, set the testing interval, and set the element range to be tested. For more details see,^JT.

To request a summary of the printer's printhead test, type ~HQJT

The printer responds with data similar to this:

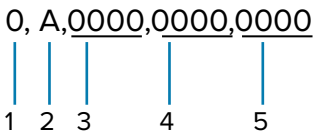
```
PRINT HEAD TEST RESULTS
0,A,0000,0000,0000
```

When the printer has printed enough labels to trigger a printhead test, the initial data changes.

To request a summary of the printer's printhead test, type ~HQJT The printer responds with data similar to this:

```
PRINT HEAD TEST RESULTS:
0,A,0015,0367,0000
```

This illustration identifies the printhead test field definitions:



1	Element failure
2	Manual (M) or automatic (A) range
3	First test element
4	Last test element
5	Failure count

Example: This is an example of how to use the maintenance alert query for the ~HQ command.

To get the current settings, type ~HQMA

The printer responds with data similar to this:

```
~HQMA
MAINTENANCE ALERT SETTINGS
HEAD REPLACEMENT INTERVAL:      1 km
HEAD REPLACEMENT FREQUENCY:     0 M
HEAD CLEANING INTERVAL:         0 M
HEAD CLEANING FREQUENCY:        0 M
PRINT REPLACEMENT ALERT:         NO
PRINT CLEANING ALERT:            NO
UNITS:                            C
```

Example: This is an example of how to use the maintenance information query for the ~HQ command.

Note that the message is controlled by the ^MI command.

To get the current settings, type ~HQMI

The printer responds with data similar to this:

```
MAINTENANCE ALERT MESSAGES
CLEAN: PLEASE CLEAN PRINT HEAD
REPLACE: PLEASE REPLACE PRINT HEAD
```

Example: This is an example of how to use the odometer query for the ~HQ command.

Note that the units of measure are controlled by the ^MA command. Also, if the "Early Warning Maintenance State" is turned "ON" the printer response would also list LAST CLEANED and CURRENT PRINTHEAD LIFE counters.

To get the current settings, type ~HQOD

The printer responds with data similar to this:

```
PRINT METERS
TOTAL NONRESETTABLE:         8560 "
USER RESETTABLE CNTR1:          9 "
```

```
USER RESETTABLE CNTR2:       8560 "
```

The units of measure are set to inches.

To change the units of measure to centimeters, type:

```
^XA^MA,,,,C
^XZ
```

The units of measure are set to centimeters.

a. To check the settings, type ~HQOD

The printer responds with data similar to this:

```
PRINT METERS
TOTAL NONRESETTABLE:        21744 cm
USER RESETTABLE CNTR1:         24 cm
USER RESETTABLE CNTR2:      21744 cm
```

Example: This is an example of how to use the printhead life query for the ~HQ command.

Note that the units of measure are controlled by the ^MA command.

To get the current settings, type ~HQPH

The printer responds with data similar to this:

```
LAST CLEANED: 257 "
HEAD LIFE HISTORY
#   DISTANCE
1:        257 "
2:       1489 "
3:       7070 "
```

line 1	The current life of the print head.
lines 2–10	Tracks the measurement for each time the print head is changed. (The example only shows lines 2 and 3.)

Example: This is an example of how to request the printer's Plug and Play string.

To request the printer's Plug and Play string, type ~HQPP

The printer responds with data similar to this:

```
    PLUG AND PLAY MESSAGES
    MFG: Zebra Technologies
    CMD: ZPL
    MDL: GX420t
```

Example: This is an example of how to retrieve the printer's serial number.

To get the printer's serial number, type ~HQSN

The printer responds with data similar to this:

```
SERIAL NUMBER
41A06440023
```

Example: This is an example of how to retrieve the printer's USB product ID and BCD release version.

To get the printer's USB product ID and BCD release version, type ~HQUI

The printer responds with data similar to this:

```
USB INFORMATION
PID:                        0085
RELEASE VERSION:            15.01
```

~HS

When the host sends ~HS to the printer, the printer sends three data strings back. Each string starts with an <STX> control code and is terminated by an <ETX><CR><LF> control code sequence. To avoid confusion, the host prints each string on a separate line.

Host Status Return

NOTE: When a ~HS command is sent the printer will not send a response to the host if the printer is in one of these conditions:

- MEDIA OUT
- RIBBON OUT
- HEAD OPEN
- REWINDER FULL
- HEAD OVER-TEMPERATURE

String 1

```
<STX>aaa,b,c,dddd,eee,f,g,h,iii,j,k,l<ETX><CR><LF>
```

aaa	communication (interface) settings[1]
b	paper out flag (1 = paper out)
c	pause flag (1 = pause active)
dddd	label length (value in number of dots)
eee	number of formats in receive buffer
f	buffer full flag (1 = receive buffer full)
g	communications diagnostic mode flag (1 = diagnostic mode active)
h	partial format flag (1 = partial format in progress)
iii	unused (always 000)
j	corrupt RAM flag (1 = configuration data lost)
k	temperature range (1 = under temperature)
l	temperature range (1 = over temperature)

1. This string specifies the printer's baud rate, number of data bits, number of stop bits, parity setting, and type of handshaking. This value is a three-digit decimal representation of an eight-bit binary number. To evaluate this parameter, first convert the decimal number to a binary number.

$$aaa=a^8\ a^7\ a^6\ a^5\ a^4\ a^3\ a^2\ a^1\ a^0$$

The nine-digit binary number is read according to this table:

a^7 = Handshake • 0 = Xon/Xoff • 1 = DTR	a^8 a^2 a^1 a^0 = Baud 0 000 = 110 0 001 = 300 0 010 = 600 0 011 = 1200 0 100 = 2400 0 101 = 4800 0 110 = 9600 0 111 = 19200 1 000 = 28800 (available only on certain printer models) 1 001 = 38400 (available only on certain printer models) 1 010 = 57600 (available only on certain printer models) 1 011 = 14400
a^6 = Parity Odd/Even • 0 = Odd • 1 = Even	
a^5 = Disable/Enable • 0 = Disable • 1 = Enable	
a^4 = Stop Bits • 0 = 2 Bits • 1 = 1 Bit	
a^3 = Data Bits • 0 = 7 Bits • 1 = 8 Bits	

String 2

```
<STX>mmm,n,o,p,q,r,s,t,uuuuuuuu,v,www<ETX><CR><LF>
```

mmm	function settings[1]
n	unused
o	**head up** flag (1 = head in up position)
p	**ribbon out** flag (1 = ribbon out)
q	**thermal transfer mode** flag (1 = Thermal Transfer Mode selected)
r	Print Mode

.14↑ Values 4 to 5 are supported only in firmware version V60.14.x, V50.14.x, V53.15.x, or later.	0 = Rewind 1 = Peel-Off 2 = Tear-Off 3 = Cutter 4 = Applicator 5 = Delayed cut 6 = Linerless Peel 7 = Linerless Rewind 8 = Partial Cutter 9 = RFID K = Kiosk S = A = Kiosk CutStream
s	print width mode
t	**label waiting** flag (1 = label waiting in Peel-off Mode)
uuuuuuuu	labels remaining in batch
v	**format while printing** flag (always 1)
www	number of graphic images stored in memory

1. This string specifies the printer's media type, sensor profile status, and communication diagnostics status. As in String 1, this is a three-digit decimal representation of an eight-bit binary number. First, convert the decimal number to a binary number. These values are only supported on the ZE500, Xi4, RXi4, ZM400/ZM600, and RZ400/RZ600 printers.

```
mmmm = m7 m6 m5 m4 m3 m2 m1 m0
```

The eight-digit binary number is read according to this table:

m^7 = Media Type • 0 = Die-Cut • 1 = Continuous	m^4 m^3 m^2 m^1 = Unused • 0 = Off • 1 = On
m^6 = Sensor Profile • 0 = Off	m^0 = Print Mode • 0 = Direct Thermal • 1 = Thermal Transfer
m^5 = Communications Diagnostics • 0 = Off • 1 = On	

String 3

```
<STX>xxxx,y<ETX><CR><LF>
```

y	0 (static RAM not installed) 1 (static RAM installed)

^HT

The ^HT command receives the complete list of font links over a communication port.

Host Linked Fonts List

.14↑

This command is available only for printers with firmware version V60.14.x, V50.14.x, or later.

The `SWISS.721.TTF` is the base font, `ANMDJ.TTF` is the first linked font, and `MSGOTHIC.TTF` is the second linked font:

ZPL II CODE	DATA RETURNED
^XA ^HT ^XZ	LIST OF FONT LINKS E:SWISS721.TTF E:ANMDJ.TTF E:MSGOTHIC.TTF

This is the code that was used to establish the font links:

```
^XA
^FLE:ANMDJ.TTF,E:SWISS721.TTF,1^FS
^FLE:MSGOTHIC.TTF,E:SWISS721.TTF,1^FS
^XZ
```

~HU

This command returns the table of configured ZebraNet Alert settings to the host.

Return ZebraNet Alert Configuration

Format: ~HU

Example: If the ~HU command is sent to the printer with existing Alert messages set to go to e-mail and SNMP traps, the data returned would look something like the information below. See ^SX on page 351.

```
B,C,Y,Y,ADMIN@COMPANY.COM,0
J,F,Y,Y,,0
C,F,Y,Y,,0
D,F,Y,Y,,0
E,F,Y,N,,0
F,F,Y,N,,0
H,C,Y,N,ADMIN@COMPANY.COM,0
N,C,Y,Y,ADMIN@COMPANY.COM,0
O,C,Y,Y,ADMIN@COMPANY.COM,0
P,C,Y,Y,ADMIN@COMPANY.COM,0
```

IMPORTANT: If there are no ^SX (alerts) set, the printer will not respond to the ~HU command.

The first line indicates that condition B (ribbon out) is routed to destination C (e-mail address).

The next two characters, Y and Y, indicate that the condition set and condition clear options have been set to yes.

The following entry is the destination that the Alert e-mail should be sent to; in this example it is admin@company.com.

The last figure seen in the first line is 0, which is the port number.

Each line shows the settings for a different Alert condition as defined in the ^SX command.

^HV

Use this command to return data from specified fields, along with an optional ASCII header, to the host computer. You can use this command with any field that has been assigned a number with the ^FN and ^RF commands.

Host Verification

Format: ^HV#,n,h,t,a

Parameters	Details
# = field number specified with another command	The value assigned to this parameter should be the same as the one used in another command. **Values:** 0 to 9999 **Default:** 0
n = number of bytes to be returned	**Values:** 1 to 256 **Default:** 64
h = header to be returned with the data	Delimiter characters terminate the string. This field is Field Hex (^FH) capable. **Values:** 0 to 3072 bytes **Default:** no header
t = termination	This field is Field Hex (^FH) capable. **Values:** 0 to 3072 characters
a = command applies to	When ^PQ is greater than 1 or if a void label occurs, send one response for a label format or one for every label printed. **Values:** • F = Format • L = Label **Default:** F

The following code:

```
^XA
.
.
.
^FH_^HV0,8,EPC[,]_0D_0A,L^FS
^PQ2
^XZ
```

Would return data similar to this:

```
EPC[12345678]
EPC[55554444]
```

^HW

^HW is used to transmit a directory listing of objects in a specific memory area (storage device) back to the host device. This command returns a formatted ASCII string of object names to the host.

Host Directory List

Each object is listed on a line and has a fixed length. The total length of a line is also fixed. Each line listing an object begins with the asterisk (*) followed by a blank space. There are eight spaces for the object name, followed by a period and three spaces for the extension. The extension is followed by two blank spaces, six spaces for the object size, two blank spaces, and three spaces for option flags (reserved for future use). The format looks like this:

```
<STX><CR><LF>
DIR R: <CR><LF>
*Name.ext(2sp.)(6 obj. sz.)(2sp.)(3 option flags)
*Name.ext(2sp.)(6 obj. sz.)(2sp.)(3 option flags)
<CR><LF>
-xxxxxxx bytes free
<CR><LF>
<ETX>
<STX> = start of text
<CR><LR> = carriage return/line feed
<ETX> = end on text
```

The command might be used in a stand-alone file to be issued to the printer at any time. The printer returns the directory listing as soon as possible, based on other tasks it might be performing when the command is received.

This command, like all ^ (caret) commands, is processed in the order that it is received by the printer.

Format: ^HWd:o.x

Parameters	Details
d = location to retrieve object listing	**Values:** R:, E:, B:, A: and Z: **Default:** R:
o = object name	**Values:** 1 to 8 alphanumeric characters **Default:** asterisk (*). A question mark (?) can also be used.
x = extension	**Values:** any extension conforming to Zebra conventions **Default:** asterisk (*). A question mark (?) can also be used.
f = format .16↑ The f parameter is only supported in firmware version V60.16.0Z and V53.16.0Z or later.	**Values:** c = column format d = default format **Default:** d

Example: Listed is an example of the ^HW command to retrieve from information R:

```
^XA
```

```
^HWR:*.*
^XZ
```

The printer returned this information as the Host Directory Listing:`-DIR R:*.*`

```
*R:ARIALN1.FNT 49140
*R:ARIALN2.FNT 49140
*R:ARIALN3.FNT 49140
*R:ARIALN4.FNT 49140
*R:ARIALN.FNT 49140
*R:ZEBRA.GRF 8420
-794292 bytes free R:RAM
```

^HY

The ^HY command is an extension of the ^HG command. ^HY is used to upload graphic objects from the printer in any supported format.

Upload Graphics

Format: ^HYd:o.x

Parameters	Details
d = location of object	**Values:** R:, E:, B:, and A: **Default:** search priority
o = object name	**Values:** 1 to 8 alphanumeric characters **Default:** an object name must be specified
x = extension	**Values:** G = .GRF (raw bitmap format)P = .PNG (compressed bitmap format) **Default:** format of stored storage

Comments: The image is uploaded in the form of a ~DY command. The data field of the returned ~DY command is always encoded in the ZB64 format.

^HZ

The ^HZ command is used for returning printer description information in XML format. The printer returns information on format parameters, object directories, individual object data, and print status information.

Display Description Information

Format: ^HZb

Parameters	Details
b = display description to return	**Values:** a = display all information f = display printer format setting information l = display object directory listing information o = display individual object data information r = display printer status information **Default:** if the value is missing or invalid, the command is ignored

Format: ^HZO,d:o.x,l

Parameters	Details
d = location of stored object	**Values:** R:, E:, B:, and A: **Default:** R:
o = object name	**Values:** 1 to 8, or 1 to 16 alphanumeric characters based on parameter l. **Default:** if a name is not specified, UNKNOWN is used.
x = extension	Supported extensions for objects (parameter o) include: .FNT — font .GRF — graphic .PNG — compressed graphic .ZPL — stored format .DAT — encoding table .ZOB — downloadable object .STO — Alert data file
l = long filename support	**Values:** Y = Yes If Y, the object data stores the filename as 16 characters. The data is only compatible with firmware version V60.13.0.5, or later. N = No If N, the object data stores the filename as 8 characters. The data is forward and backward compatible with all versions of firmware. **Default:** N

Example: This example shows the object data information for the object SAMPLE.GRF located on R:.

```
^XA
```

```
^HZO,R:SAMPLE.GRF
^XZ
```

^ID

The ^ID command deletes objects, graphics, fonts, and stored formats from storage areas. Objects can be deleted selectively or in groups. This command can be used within a printing format to delete objects before saving new ones, or in a stand-alone format to delete objects.

Object Delete

The image name and extension support the use of the asterisk (*) as a wild card. This allows you to easily delete a selected groups of objects.

Format: ^IDd:o.x

Parameters	Details
d = location of stored object	**Values:** R:, E:, B:, and A: **Default:** R:
o = object name	**Values:** any 1 to 8 character name **Default:** if a name is not specified, UNKNOWN is used
x = extension	**Values:** any extension conforming to Zebra conventions **Default:** .GRF

To delete stored formats from DRAM:

```
^XA
^IDR:*.ZPL^FS
```

```
^XZ
```

To delete formats and images named SAMPLE from DRAM, regardless of the extension:

```
^XA
^IDR:SAMPLE.*^FS
^XZ
```

To delete the image SAMPLE1.GRF prior to storing SAMPLE2.GRF:

```
^XA
^FO25,25^AD,18,10
^FDDelete^FS
^FO25,45^AD,18,10
^FDthen Save^FS
^IDR:SAMPLE1.GRF^FS
^ISR:SAMPLE2.GRF^FS^XZ
```

In this the * is a wild card, indicating that all objects with the .GRF extension are deleted:

```
^XA
^IDR:*.GRF^FS
```

```
^XZ
```

Comments: When an object is deleted from R:, the object can no longer be used and memory is available for storage. This applies only to R: memory. With the other memory types (A:, B:, E:) the deleted object is no longer available. The memory space recovers when an automatic defragmentation or initialization occurs.

The ^ID command also frees up the uncompressed version of the object in DRAM.

If the name is specified as *.Z0B, all downloaded bar code fonts (or other objects) are deleted.

If the named downloadable object cannot be found in the R:, E:, B:, and A: device, the ^ID command is ignored.

^IL

The ^IL command is used at the beginning of a label format to load a stored image of a format and merge it with additional data. The image is always positioned at ^FO0,0.

Image Load

IMPORTANT: See ^IS.

Using this technique to overlay the image of constant information with variable data greatly increases the throughput of the label format.

Format: ^ILd:o.x

Parameters	Details
d = location of the stored object	**Values:** R:, E:, B:, and A: **Default:** R:
o = object name	**Values:** 1 to 8 alphanumeric characters **Default:** if a name is not specified, UNKNOWN is used
x = extension	**Fixed Value:** .GRF, .PNG

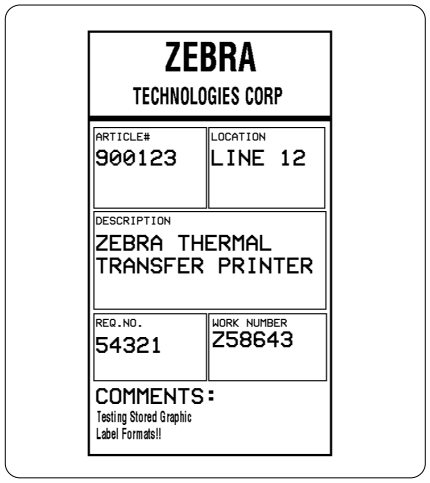
Example

This example recalls the stored image SAMPLE2.GRF from DRAM and overlays it with the additional data. The graphic was stored using the ^IS command. For the stored label format, see the ^IS command.

ZPL II CODE

```
^XA
^ILR:SAMPLE2.GRF^FS
^CFD,36,20
^FO15,210
^FD900123^FS
^FO218,210
^FDLINE  12^FS
^FO15,360^AD
^FDZEBRA  THERMAL^FS
^FO15,400^AD
^FDTRANSFER  PRINTER^FS
^FO15,540
^FD54321^FS
^FO220,530
^FDZ58643^FS
^FO15,670^A0,27,18
^FDTesting  Stored  Graphic^FS
^FO15,700^A0,27,18
^FDLabel  Formats!!^FS
^XZ
```

GENERA TED LABEL



^IM

The ^IM command performs a direct move of an image from storage area into the bitmap. The command is identical to the ^XG command (Recall Graphic), except there are no sizing parameters.

Image Move

Format: ^IMd:o.x

Parameters	Details
d = location of stored object	**Values:** R:, E:, B:, and A: **Default:** search priority
o = object name	**Values:** 1 to 8 alphanumeric characters **Default:** if a name is not specified, UNKNOWN is used
x = extension	**Fixed Value:** .GRF, .PNG

Example: This example moves the image SAMPLE.GRF from DRAM and prints it in several locations in its original size.

```
^XA

^FO100,100^IMR:SAMPLE.GRF^FS
^FO100,200^IMR:SAMPLE.GRF^FS
^FO100,300^IMR:SAMPLE.GRF^FS
^FO100,400^IMR:SAMPLE.GRF^FS
^FO100,500^IMR:SAMPLE.GRF^FS
^XZ
```

Comments: By using the ^FO command, the graphic image can be positioned anywhere on the label.

The difference between ^IM and ^XG: ^IM does not have magnification, and therefore might require less formatting time. However, to take advantage of this, the image must be at a 8-, 16-, or 32-bit boundary.

^IS

The ^IS command is used within a label format to save that format as a graphic image rather than as a ZPL II script. It is typically used toward the end of a script. The saved image can later be recalled with virtually no formatting time and overlaid with variable data to form a complete label.

Image Save

Using this technique to overlay the image of constant information with the variable data greatly increases the throughput of the label format.

IMPORTANT: See ^IL.

Format: ^ISd:o.x,p

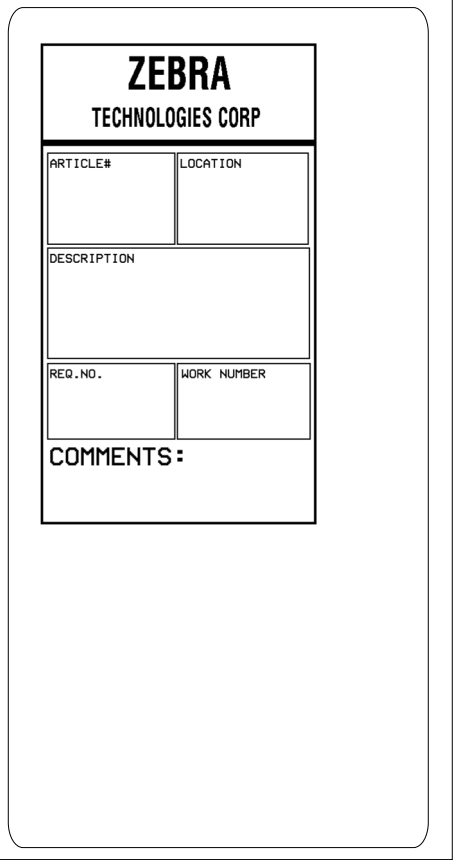
Parameters	Details
d = location of the stored object	**Values:** R:, E:, B:, and A: **Default:** R:
o = object name	**Values:** 1 to 8 alphanumeric characters **Default:** if a name is not specified, UNKNOWN is used
x = extension	**Values:** .GRF or .PNG **Default:** .GRF
p = print image after storing	**Values:** N = no Y = yes **Default:** Y

Example

This is an example of using the ^IS command to save a label format to DRAM. The name used to store the graphic is SAMPLE2.GRF.

ZPL II CODE	GENERA TED LABEL

```
^XA
^LH10,15^FWN^BY3,3,85^CFD,36
^GB430,750,4^FS
^FO10,170^GB200,144,2^FS
^FO10,318^GB410,174,2^FS
^FO212,170^GB206,144,2^FS
^FO10,498^GB200,120,2^FSR
^FO212,498^GB209,120,2^FS
^FO4,150^GB422,10,10^FS
^FO135,20^A0,70,60
^FDZEBRA^FS
^FO80,100^A0,40,30
^FDTECHNOLOGIES   CORP^FS
^FO15,180 ^CFD,18,10^FS
^FDARTICLE#^FS
^FO218,180
^FDLOCATION^FS
^FO15,328
^FDDESCRIPTION^FS
^FO15,508
^FDREQ.NO.^FS
^FO220,508
^FDWORK  NUMBER^FS
^FO15,630^AD,36,20
^FDCOMMENTS:^FS
^ISR:SAMPLE2.GRF,Y
^XZ
```



~JA

The ~JA command cancels all format commands in the buffer. It also cancels any batches that are printing.

Cancel All

The printer stops after the current label is finished printing. All internal buffers are cleared of data and the **DATA** LED turn off.

Submitting this command to the printer scans the buffer and deletes only the data before the ~JA in the input buffer — it does not scan the remainder of the buffer for additional ~JA commands.

Format: ~JA

^JB

The ^JB command is used to initialize various types of Flash memory available in the Zebra printers.

Initialize Flash Memory

Format: ^JBa

Parameters	Details
a = device to initialize	**Values:** A = Option Flash memory B = Flash card (PCMCIA) E = internal Flash memory **Default:** a device must be specified

Example: This is an example of initializing the different types of flash memory:

^JBA – initializes initial Compact Flash memory when installed in the printer.

^JBB – initializes the optional Flash card when installed in the printer.

^JBE – initializes the optional Flash memory when installed in the printer.

NOTE: Initializing memory can take several minutes. Be sure to allow sufficient time for the initialization to complete before power cycling the printer.

~JB

Reset Optional Memory

The ~JB command is used for these conditions:

- The ~JB command must be sent to the printer if the battery supplying power to the battery powered memory card fails and is replaced. A bad battery shows a battery dead condition on the Printer Configuration Label.
- The ~JB command can also be used to intentionally clear (reinitialize) the B: memory card. The card must not be write protected.

Format: ~JB

Comments: If the battery is replaced and this command is not sent to the printer, the memory card cannot function.

~JC

The ~JC command is used to force a label length measurement and adjust the media and ribbon sensor values.

Set Media Sensor Calibration

Format: ~JC

Comments: In Continuous Mode, only the media and ribbon sensors are calibrated.

This command is ignored on the HC100™ printer.

~JD

The ~JD command initiates Diagnostic Mode, which produces an ASCII printout (using current label length and full width of printer) of all characters received by the printer. This printout includes the ASCII characters, the hexadecimal value, and any communication errors.

Enable Communications Diagnostics

Format: ~JD

~JE

The ~JE command cancels Diagnostic Mode and returns the printer to normal label printing.

Disable Diagnostics

Format: ~JE

~JF

Set Battery Condition

There are two low battery voltage levels sensed by the PA/PT400™ printers. When battery voltage goes below the first level, the green LED begins flashing as a warning but printing continues. When this warning occurs, it is recommended to recharge the battery.

As printing continues, a second low voltage level is reached. At this point, both green and orange LEDs flash as a warning, and printing automatically pauses.

When pause on low voltage is active (~JFY) and the battery voltage level falls below the second low voltage level, printing pauses and an error condition is displayed as an indication that the printer should be plugged into the battery charger. By pressing FEED, printing continues on a label-by-label basis, but there is a high risk of losing label format information due to the continued decrease of battery voltage.

When pause on low voltage is not active (~JFN), and the battery voltage level falls below the second low voltage level, printing continues and the orange LED remains off. If the battery voltage continues to decrease, label information could be lost and cause the printer to stop operating. This option should be selected only when the printer is connected to the Car Battery Adapter. From time to time the printer might sense that battery voltage is below the first low voltage level, but due to the continuous recharging of the car battery, further loss of battery voltage is not a concern and printing continues.

If this option is not selected when using the Car Battery Adapter, you might need to press FEED to take the printer out of Pause Mode and print each label.

Format: ~JFp

Parameters	Details
p = pause on low voltage	**Values:** Y (pause on low voltage) or N (do not pause) N is suggested when the printer is powered by the Car Battery Adapter. **Default:** Y

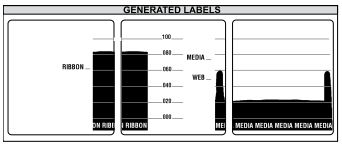
~JG

The ~JG command prints a graph (media sensor profile) of the sensor values.

Graphing Sensor Calibration

Format: ~JG

Sending the ~JG command to a printer configured for thermal transfer produces a series of labels resembling this image:



Comments

The HC100™ printer does not perform a calibration, but does print a sensor profile label.

^JH

The ^JH command configures the early warning messages that appear on the LCD.

Early Warning Settings

- ZE500 series
- **Xi**IIII, **Xi**IIII**Plus**, Xi4, RXi4
- **PAX**3, **PAX**4
- ZM400, ZM600, RZ400, RZ600
- S4M
- G-Series (“f” parameter only)

Format: ^JHa,b,c,d,e,f,g,h,i,j

Parameter	Details
a = early warning media a = supplies warning (Xi4 and RXi4 printers only)	This parameter is for XiIIIPlus, Xi4, RXi4, PAX3, and PAX4 printers only. **Values:** E = enable D = disable **Default:** D
b = labels per roll	This parameter is for XiIIIPlus, PAX3, and PAX4 printers only. **Values:** 100 to 9999 **Default:** 900
c = media replaced	This parameter is for XiIIIPlus, PAX3, and PAX4 printers only. **Values:** Y = yes N = no **Default:** N

Parameter	Details
d = ribbon length	This parameter is for XiIIIPlus, PAX3, PAX4, and ZE500 printers only. **Values:** XiIIIPlus series printers: N = 0M 0 = 100M 1 = 150M 2 = 200M 3 = 250M 4 = 300M 5 = 350M 6 = 400M 7 = 450M PAX series printers: N = 0M 0 = 100M 1 = 150M 2 = 200M 3 = 250M 4 = 300M 5 = 350M 6 = 400M 7 = 450M 10 = 600M 11 = 650M 12 = 700M 13 = 750M 14 = 800M 15 = 850M 16 = 900M ZE500 series printers: N = 0M 0 = 100M 1 = 150M 2 = 200M 3 = 250M 4 = 300M 5 = 350M 6 = 400M 7 = 450M 10 = 600M **Default:** 1 - for 96XiIIIPlus 260 7 - for all other printers

Parameter	Details
e = ribbon replaced	This parameter is for **Xi**IIII**Plus**, **PAX**3, and **PAX**4 printers only. **Values:** Y = yes N = no **Default:** N
f = early warning maintenance	This parameter is for Xi4, RXi4, **PAX**4, ZM400, ZM600, RZ400, RZ600, and S4M printers only. **Values:** E = enabled D = disabled **Default:** D **IMPORTANT:** On G-Series printers, this parameter must be enabled for the ^MA driven system to work.
g = head cleaning interval	Accepted value exceptions: accepted values for **Xi**IIII printer are 100M through 450M; accepted values for 600 dpi **Xi**IIII printers are 100M through 150M; accepted values for **PAX**4 series printers are up to 900M by increments of 50M; accepted values for ZM400/ZM600, RZ400/RZ600, and S4M printers are 0M through 450M. **Values:** 0 = 100M 1 = 150M 2 = 200M 3 = 250M 4 = 300M 5 = 350M 6 = 400M 7 = 450M 8 = 500M 9 = 550M 10= 600M 11 = 650M 12 = 700M 13 = 750M 14 = 800M 15 = 850M 16 = 900M **Default:** 1 - for 96XiIIIPlus 7 - for all other printers

Parameter	Details
h = head clean	**Values:** N = No Y = Yes **Default:**N
i = head life threshold	**Values:** 0 – 0 in or off100-3500000 in**Default:** 1000000
j = head replaced	**Values:** N = no Y = yes **Default:** N

Comments: To permanently save the changes to the ^JH command, send ^XA^JUS^XZ.

^JI

^JI works much like the ~JI command. Both commands are sent to the printer to initialize the Zebra BASIC Interpreter.

Start ZBI (Zebra BASIC Interpreter)

.16↑

Identifies features that are available in printers with firmware version V60.16.2Z, V53.16.2Z, or later.

In interactive mode, ^JI can be sent through one of the communication ports (serial, parallel, or Ethernet) to initialize the printer to receive ZBI commands. This command can be sent from one of the Zebra software utilities, such as ZTools, or from a terminal emulation program.

When the command is received, the printer responds by sending a ZBI header back to the console, along with the program version number. This indicates that the interpreter is active.

Format: ^JId:o.x,b,c,d

Parameters	Details
d = location of program to run after initialization	**Values:** R:, E:, B:, and A: **Default:** location must be specified
o = name of program to run after initialization	**Values:** any valid program name **Default:** name must be specified
x = extension of program to run after initialization	**Fixed Value:** .BAS, .BAE **NOTE:** .BAE is only supported in firmware version V60.16.0Z or later
b = console control	**Values:** Y = console on N = console off **Default:** Y
c = echoing control	**Values:** Y = echo on N = echo off **Default:** Y
d = memory allocation for ZBI *	**Values:** 20K to 1024K **Default:** 50K

* This parameter is only available on printers with firmware V60.12.0.x or earlier.

Comments

When the printer is turned on, it can receive ZPL II commands and label formats. However, for the printer to recognize ZBI commands and programs, it must be initialized using ^JI or ~JI.

Only one ZBI interpreter can be active in the printer at a time. If a second ^JI or ~JI command is received while the interpreter is running, the command is ignored.

The interpreter is deactivated by entering one of two commands:

ZPL at the ZBI prompt

~JQ at an active ZPL port

~JI

~JI works much like the ^JI command. Both commands are sent to the printer to initialize the Zebra BASIC Interpreter.

Start ZBI (Zebra BASIC Interpreter)

Identifies features that are available in printers with firmware version V60.16.2Z, V53.16.2Z, or later.

In interactive mode, ~JI can be sent through one of the communication ports (serial, parallel, or Ethernet) to initialize the printer to receive ZBI commands. This command can be sent from one of the Zebra software utilities, such as ZTools, or from a standard PC program, such as Hyper terminal.

When the command is received, the printer responds by sending a ZBI header back to the console, along with the program version number. This indicates that the interpreter is active.

Format: ~JI

Comments: While receiving commands, the printer echoes the received characters back to the source. This can be toggled on and off with the ZBI ECHO command.

When the printer is turned on, it can receive ZPL II commands and label formats. However, for the printer to recognize ZBI commands and formats, it must be initialized using ^JI or ~JI.

Only one ZBI interpreter can be active in the printer at a time. If a second ~JI or ^JI command is received while the interpreter is running, the command is ignored.

The interpreter is deactivated by entering one of these commands:

ZPL at the ZBI prompt

~JQ at an active ZPL port

^JJ

The ^JJ command allows you to control an online verifier or applicator device.

Set Auxiliary Port

Format: ^JJa,b,c,d,e,f

Parameters	Details
a = operational mode for auxiliary port	**Values:** 0 = off 1 = reprint on error—the printer stops on a label with a verification error. When **PAUSE** is pressed, the label reprints (if ^JZ is set to reprint). If a bar code is near the upper edge of a label, the label feeds out far enough for the bar code to be verified and then backfeeds to allow the next label to be printed and verified. 2 = maximum throughput—the printer stops when a verification error is detected. The printer starts printing the next label while the verifier is still checking the previous label. This mode provides maximum throughput, but does not allow the printer to stop immediately on a label with a verification error. **Default:** 0
b = application mode	**Values:** 0 = off 1 = End Print signal normally high, and low only when the printer is moving the label forward. 2 = End Print signal normally low, and high only when the printer is moving the label forward. 3 = End Print signal normally high, and low for 20 ms when a label has been printed and positioned. 4 = End Print signal normally low, and high for 20 ms when a label has been printed and positioned. **Default:** 0 **NOTE:** The Set/Get/Do command device.applicator.end_print on page 664controls the same setting as the b parameter.
c = application mode start signal print	**Values:** p = Pulse Mode – Start Print signal must be de-asserted before it can be asserted for the next label. l = Level Mode – Start Print signal does not need to be de-asserted to print the next label. As long as the Start Print signal is low and a label is formatted, a label prints. **Default:** 0

Parameters	Details
d = application label error mode	**Values:** e = error mode—the printer asserts the **Service Required** signal (svce_req - pin 10) on the application port, enters into Pause Mode, and displays an error message on the LCD. f = Feed Mode—a blank label prints when the web is not found where expected to sync the printer to the media. **Default:** f
e = reprint mode	**Values:** e = enabled—the last label reprints after the signal is asserted. If a label is canceled, the label to be reprinted is also canceled. This mode consumes more memory because the last printed label is not released until it reprints. d = disabled—printer ignores the Reprint signal. **Default:** d
f = ribbon low mode	**Values:** e = enabled – printer warning issued when ribbon low. d = disabled – printer warning not issued when ribbon low. **Default:** e

~JL

The ~JL command is used to set the label length. Depending on the size of the label, the printer feeds one or more blank labels.

Set Label Length

Format: ~JL

^JM

The ^JM command lowers the density of the print—24 dots/mm becomes 12, 12 dots/mm becomes 6, 8 dots/mm becomes 4, and 6 dots/mm becomes 3. ^JM also affects the field origin (^F0) placement on the label (see example below).

Set Dots per Millimeter

When sent to the printer, the ^JM command doubles the format size of the label. Depending on the printhead, normal dot-per-millimeter capabilities for a Zebra printer are 12 dots/mm (304 dots/inch), 8 dots/mm (203 dots/inch), or 6 dots/mm (153 dots/inch).

This command must be entered before the first ^FS command in a format. The effects of ^JM are persistent.

Format: ^JMn

Parameters	Details
n = set dots per millimeter	**Values:** A = 24 dots/mm, 12 dots/mm, 8 dots/mm, or 6 dots/mm B = 12 dots/mm, 6 dots/mm, 4 dots/mm, or 3 dots/mm **Default:** A

Example

This example shows the effects of alternating the dots per millimeter:

ZPL II CODE	GENERA TED LABEL
^XA ^JMA^FS ^FO100,100 ^B2N,50,Y,N,N ^FD1234567890^FS ^XZ	
^XA ^JMB^FS ^FO100,100 ^B2N,50,Y,N,N ^FD1234567890^FS ^XZ	

Comments: If ^JMB is used, the UPS MaxiCode bar code becomes out of specification.

~JN

The ~JN command turns on the head test option. When activated, ~JN causes the printer to halt when a head test failure is encountered.

Head Test Fatal

Once an error is encountered the printer remains in error mode until the head test is turned off (~J0) or power is cycled.

Format: ~JN

Comments: If the communications buffer is full, the printer is not able to receive data. In this condition, the ~J0 command is not received by the printer.

~JO

The ~JO command configures the printer to run the head test with error reporting enabled. When ~JO is used an error will be displayed and printing will stop if the head test fails. The user can push the PAUSE button on the printer to bypass the error. This command differs from the ~JN (Head Test Fatal) command in that a power cycle is not required in the event of a head test failure.

Head Test Non-Fatal

~JO is the default print head test condition. This setting is changed when the printer receives a ~JN (Head Test Fatal) command.

Format: ~JO

~JP

The ~JP command clears the format currently being processed and places the printer into Pause Mode.

Pause and Cancel Format

The command clears the next format that would print, or the oldest format from the buffer. Each subsequent ~JP command clears the next buffered format until the buffer is empty. The **DATA** indicator turns off when the buffer is empty and no data is being transmitted.

Issuing the ~JP command is identical to using **CANCEL** on the printer, but the printer does not have to be in Pause Mode first.

Format: ~JP

~JQ

The ~JQ command is used when Zebra BASIC Interpreter is active. Sending ~JQ to the printer terminates the ZBI session.

Terminate Zebra BASIC Interpreter

Identifies features that are available in printers with firmware version V60.16.2Z, V53.16.2Z, or later.

Format: ~JQ

Comments

Entering ZPL at the command prompt also terminates a ZBI session.

~JR

The ~JR command resets all of the printer's internal software, performs a power-on self-test (POST), clears the buffer and DRAM, and resets communication parameters and default values. Issuing a ~JR command performs the same function as a manual power-on reset.

Power On Reset

Format: ~JR

^JS

NEED SHORT DESCRIPTION

Sensor Select

Format: ^JSa

NOTE: This command is ignored on Zebra ZM400/ZM600 and RZ400/RZ600 printers. This command is only for use with the S4M and Z Series printers (with the exception of the ZM400/ZM600/RZ400/RZ600).

Parameters	Details
a = sensor selection	**Values:** A = auto select R = reflective sensor T = transmissive sensor **Default:** Z series = A S4M = R

~JS

The ~JS command is used to control the backfeed sequence. This command can be used on printers with or without built-in cutters. This command is ignored on the HC100™ printer.

Change Backfeed Sequence

These are the primary applications:

- to allow programming of the rest point of the cut edge of continuous media.
- provide immediate backfeed after peel-off when the printer is used in a print/apply application configuration.

This command stays in effect only until the printer is turned off, a new ~JS command is sent, or the setting is changed on the control panel. When a ~JS command is encountered, it overrides the current control panel setting for the Backfeed Sequence.

The most common way of eliminating backfeed is to operate in Rewind Mode. Rewind Mode does not backfeed at all. After a label prints, the leading edge of the next label is placed at the print line. This eliminates the need to backfeed and does not introduce a non printable area at the leading edge or bottom of the label. It also does not allow the label to be taken from the printer because it is not fed out from under the printhead.

Running in another mode with backfeed turned off allows the label to be removed and eliminates the time-reduction of the backfeed sequence.

Format: ~JSb

Parameters	Values
b = backfeed order in relation to printing	A —100 percent backfeed after printing and cutting B —0 percent backfeed after printing and cutting, and 100 percent before printing the next label N —normal — 90 percent backfeed after label is printed 0 —off — turn backfeed off completely 10 to 90 —percentage value The value entered must be a multiple of 10. Values not divisible by 10 are rounded to the nearest acceptable value. For example, ~JS55 is accepted as 60 percent backfeed. **Default:** N

Comments: When using a specific value, the difference between the value entered and 100 percent is calculated before the next label is printed. For example, a value of 40 means 40 percent of the backfeed takes place after the label is cut or removed. The remaining 60 percent takes place before the next label is printed.

The value for this command is also reflected in the Backfeed parameter on the printer configuration label.

For ~JSN — the Backfeed parameter is listed as DEFAULT

For ~JSA — or 100% the Backfeed parameter is listed as AFTER

For ~JSB — or 0% the Backfeed parameter is listed as BEFORE

For ~JS10 — 10% of the backfeed takes place after the label is cut or removed. The remaining 90% takes place before the next label is printed.

^JT

The ^JT command allows you to change the printhead test interval from every 100 labels to any desired interval. With the ^JT command, the printer is allowed to run the test after printing a label. When a parameter is defined, the printer runs the test after printing a set amount of labels.

Head Test Interval

The printer's default head test state is off. Parameters for running the printhead test are defined by the user.

Format: ^JT####,a,b,c

Parameters	Details
#### = four-digit number of labels printed between head tests	**Values:** 0000 to 9999 If a value greater than 9999 is entered, it is ignored. **Default:** 0000 (off)
a = manually select range of elements to test	**Values:** N = no Y = yes **Initial Value at Power Up:** N
b = first element to check when parameter a is Y	**Values:** 0 to 9999 **Initial Value at Power Up:** 0
c = last element to check when parameter a is Y	**Values:** 0 to 9999 **Initial Value at Power Up:** 9999

Comments: The ^JT command supports testing a range of print elements. The printer automatically selects the test range by tracking which elements have been used since the previous test.

^JT also turns on Automatic Mode to specify the first and last elements for the head test. This makes it possible to select any specific area of the label or the entire print width.

If the last element selected is greater than the print width selected, the test stops at the selected print width.

Whenever the head test command is received, a head test is performed on the next label unless the count is set to 0 (zero).

^JU

The ^JU command sets the active configuration for the printer.

Configuration Update

Format: ^JUa

Parameters	Details
a = active configuration	**Values:** A =reload factory settings and factory network settings. F = reload factory settings. N = reload factory network settings. These values are lost at power-off if not saved with ^JUS. R = recall the last saved settings. S = save current settings. These values are used at power-on. **Default:** a value must be specified

^JW

^JW sets the ribbon tension for the printer it is sent to.

Set Ribbon Tension

Format: ^JWt

Parameters	Details
t = tension	**Values:** L = low M = medium H = high **Default:** a value must be specified

Comments: ^JW is used only for **PAX** series printers.

~JX

The ~JX command cancels a format currently being sent to the printer. It does not affect any formats currently being printed, or any subsequent formats that might be sent.

Cancel Current Partially Input Format

Format: ~JX

^JZ

The ^JZ command reprints a partially printed label caused by a Ribbon Out, Media Out, or Head Open error condition. The label is reprinted as soon as the error condition is corrected.

Reprint After Error

This command remains active until another ^JZ command is sent to the printer or the printer is turned off.

Format: ^JZa

Parameters	Details
a = reprint after error	**Values:** N = no Y = yes **Initial Value at Power Up:** Y

Comments: ^JZ sets the error mode for the printer. If ^JZ changes, only labels printed after the change are affected.

If the parameter is missing or incorrect, the command is ignored.

~KB

The ~KB command places the printer in battery discharge mode. This allows the battery to be drained without actually printing.

Kill Battery (Battery Discharge Mode)

To maintain performance of the rechargeable battery in the portable printers, the battery must be fully discharged and recharged regularly.

Format: ~KB

Comments: While the printer is in Discharge Mode, the green power LED flashes in groups of three flashes.

Discharge Mode might be terminated by sending a printing format to the printer or by pressing either of the control panel keys.

If the battery charger is plugged into the printer, the battery is automatically recharged once the discharge process is completed.

^KD

The ^KD command selects the format that the Real-Time Clock's date and time information presents as on a configuration label. This is also displayed on the Printer Idle LCD control panel display, and displayed while setting the date and time.

Select Date and Time Format (for Real Time Clock)

Format: ^KDa

Parameters	Details
a = value of date and time format	**Values:** 0 = normal, displays **Version Number** of firmware 1 = MM/DD/YY (24-hour clock) 2 = MM/DD/YY (12-hour clock) 3 = DD/MM/YY (24-hour clock) 4 = DD/MM/YY (12-hour clock) **Default:** 0

Comments: If the Real-Time Clock hardware is not present, Display Mode is set to 0 (Version Number).

If Display Mode is set to 0 (Version Number) and the Real-Time Clock hardware is present, the date and time format on the configuration label is presented in format 1.

If Display Mode is set to 0 (Version Number) and the Real-Time Clock hardware is present, the date and time format on the control panel display is presented in format 1.

For more details on select date and time format for the Real Time Clock, see Real Time Clock on page 1590.

^KL

The ^KL command selects the language displayed on the control panel.

Define Language

Format: ^KLa

Parameters	Details
a = language	**Values:** 1 = English 2 = Spanish 3 = French 4 = German 5 = Italian 6 = Norwegian 7 = Portuguese 8 = Swedish 9 = Danish 10 = Spanish2 11 = Dutch 12 = Finnish 13 = Japanese 14 = Korean* 15 = Simplified Chinese* 16 = Traditional Chinese* 17 = Russian* 18 = Polish* 19 = Czech* 20 = Romanian* **Default:** 1
* These values are only supported on the ZT200 Series, ZE500 Series, Xi4, RXi4, ZM400/ ZM600, and RZ400/RZ600 printers.	

^KN

The printer's network name and description can be set using the ^KN command. ^KN is designed to make your Zebra printer easy for users to identify. The name the administrator designates is listed on the configuration label and on the Web page generated by the printer.

Define Printer Name

Format: ^KNa,b

NOTE: If you issue the command ^KN, (without the a and b parameters) you are setting the printer name and description to a blank string.

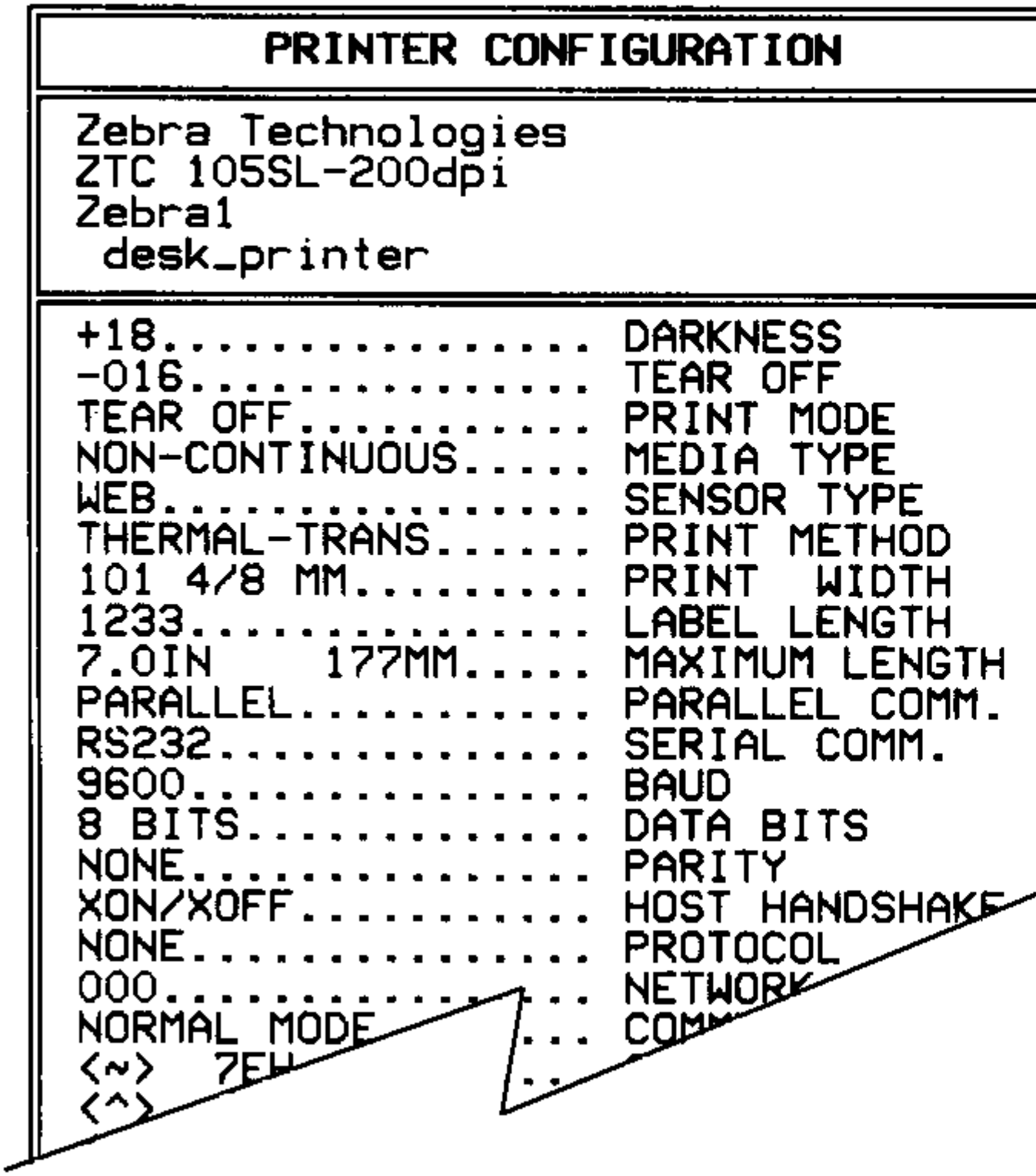
To cause the printer name and printer description settings controlled by the ^KN command to be saved, you must issue the ^JUS command.

Parameters	Details
a = printer name	**Values:** up to 16 alphanumeric characters **Default:** • If no printer name is specified in a printer with a MAC address, the printer name will default to "ZBRxxx," where xxx is the last three octets of the MAC address converted into ASCII text. • For printers without a MAC address, if a value is not entered, the current stored value is erased. If more than 16 characters are entered, only the first 16 are used.
b = printer description	**Values:** up to 35 alphanumeric characters **Default:** if a value is not entered, the current stored value is erased If more than 35 characters are entered, only the first 35 are used. The value of this parameter will be displayed on the printer's web page in parentheses.

Example: This is an example of how to change the printer's network name and description. The sample labels show how a configuration looks before using this command and after using this command:

```
^XA
^KNZebra1,desk_printer
^XZ
```

Before using this command



^KP

The ^KP command is used to define the password that must be entered to access the control panel switches and LCD Setup Mode.

Define Password

Format: ^KPa,b

Parameters	Details
a = mandatory four-digit password	**Values:** any four-digit numeric sequence **Default:** 1234
b = password level	**Values:** 1, 2, 3, 4 **Default:** 3 The b parameter applies only to the S4M printers.

Example: This example shows how to set a new control panel password:

```
^XA
^KP5678
^XZ
```

Example: This example shows how to set a new control panel password (5678) at a specific password level (level 2) (applicable to the S4M printer only):

```
^XA
^KP5678,2
^XZ
```

Comments: If you forget your password, the printer can be returned to a default Setup Mode and the default password 1234 is valid again. Caution should be used, however — this also sets the printer configuration values back to their defaults.

To return the printer to the default factory settings using ZPL, send this:

```
^XA
^JUF
^XZ
```

To return the printer to the default factory settings using the control panel keys, see your printer's User Guide for the procedure.

^KV

The ^KV command sets several parameters that affect the printers operation when ^MM is set to K - Kiosk mode

Kiosk Values

Supported Devices:

- KR403

Format: ^KVa,b,c,d,e

Parameters	Details
a = kiosk cut amount	**Values:** 0 = normal cut 10-60 = partial cut, value = mm of media left uncut **Default:** 0 This parameter is ignored if it is missing or invalid. The current value of the parameter remains unchanged.
b = kiosk cut margin	**Values:** 2 - 9 = mm of distance **Default:** 9 = mm of distance This parameter is ignored if it is missing or invalid. The current value of the parameter remains unchanged.
c = kiosk present type	**Values:** 0 = Eject page when new page is printed 1 = Retract page when new page is printed 2 = Do nothing when new page is printed **Default:** 0 This parameter is ignored if it is missing or invalid. The current value of the parameter remains unchanged.
d = kiosk present timeout	**Values:** 0–300 = If label is not taken, retract label when timeout expires. Timeout is in seconds. Zero (0) indicates that there is no timeout. The label will stay presented until removed manually or a new label is printed. **Default:** 0 This parameter is ignored if it is missing or invalid. The current value of the parameter remains unchanged.

Parameters	Details
e = presenter loop length	**Values:** 0 = paper is fed straight through the presenter 3-1023 = loop length in mm. **Default:** 400 400= gives a loop of approximately 400mm This parameter is ignored if it is missing or invalid. The current value of the parameter remains unchanged. . If this is greater than loop_length_max (see SGD media.present.loop_length_max) then it will be set equal to loop_length_max.

Kiosk Printing Examples

The following examples demonstrate the use of the ^KV, ^CN, ^PN and ^CP commands with 80 mm wide continuous media and the printer set to Kiosk Mode (^MMK).

Example: In this example, the ^KV command is set to the following:

- Cut - Full Cut
- Cut Margin - 9 mm
- Present Type - Eject page when the next page is printed
- Present Timeout - 6 seconds after printing, if the document is not taken, it will be retracted
- Presenter Loop Length - No loop

```
^XA
^MMK
^KV0,9,0,6,0
^FO50,50^A0N,50,50^FDZebra Technologies^FS
^CN1
^PN0
^XZ
```

NOTE: The ^CN1 command (Cut Now) is included to ensure that a full cut is done. The ^PN0 (Present Now) command is included to ensure that the media is ejected when the user pulls on the leading edge of the media. In this example, if the user does not pull on the leading edge of the second document, it will be retracted.

Example: This example contains only one change from the Example 1 - the Presenter Loop Length is now 100mm, and two documents will be printed instead of one.

```
^XA
^MMK
^KV0,9,2,6,100
^FO50,50^A0N,50,50^FDZebra Technologies^FS
^CN1^PN0
^PQ2
^XZ
```

Example: In this example, two documents will be printed, each one will be ejected from the printer.

```
^XA
^MMK
^KV0,9,2,6,100
^FO50,50^A0N,50,50^FDZebra Technologies^FS
^CN1^CP0
^PQ2
^XZ
```

Example: In this example, two documents, with partial cuts, will be printed, and a third document, with a full cut, will be printed.

```
^XA
^MMK
^KV50,9,0,0,0
^FO50,50^A0N,50,50^FDPartial Cut^FS
^CN0^PN0
^PQ2
^XZ
^XA
^MMK
^KV0,9,2,6,0
^FO50,50^A0N,50,50^FDFull Cut^FS
^CN1^CP0
^XZ
```

Example: In this example, four documents will be printed – three with a partial cut and the fourth with a full cut. Additionally, the document length is set to 406 dots and the Media Tracking mode is set to "Continuous Media, Variable Length". The third document contains fields that are positioned outside of the 406 dot length – however, because the printer is set to “Continuous Media, Variable Length" Media Tracking mode, the printer will automatically adjust the document length to compensate.

```
^XA
^MMK
^LL406
^KV20,9,0,0,0
^FO50,50^A0N,50,50^FDPartial Cut^FS
^CN0^PN0
^PQ2
^XZ
^XA
^MMK
^MNV
^KV20,9,0,0,0
^FO50,50^A0N,50,50^FDPartial Cut^FS
^FO50,150^A0N,50,50^FDPrinting Line 1^FS
^FO50,250^A0N,50,50^FDPrinting Line 2^FS
^FO50,350^A0N,50,50^FDPrinting Line 3^FS
^FO50,450^A0N,50,50^FDPrinting Line 4^FS
^FO50,550^A0N,50,50^FDPrinting Line 5^FS
^FO50,650^A0N,50,50^FDPrinting Line 6^FS
^FO50,750^A0N,50,50^FDPrinting Line 7^FS
^FO50,850^A0N,50,50^FDPrinting Line 8^FS
```

```
^FO50,950^A0N,50,50^FDPrinting Line 9^FS
^FO50,1050^A0N,50,50^FDPrinting Line 10^FS
^FO50,1150^A0N,50,50^FDPrinting Line 11^FS
^FO50,1250^A0N,50,50^FDPrinting Line 12^FS
^FO50,1350^A0N,50,50^FDPrinting Line 13^FS
^FO50,1450^A0N,50,50^FDPrinting Line 14^FS
^FO50,1550^A0N,50,50^FDPrinting Line 15^FS
^CN0^PN0
^XZ
^XA
^MMK
^KV0,9,0,0,0
^FO50,50^A0N,50,50^FDFull Cut^FS
^CN0^PN1^CP0
^PQ1
^XZ
```

^LF

The ^LF command prints out a list of the linked fonts.

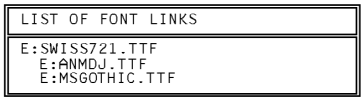
List Font Links

.14↑

This command is available only for printers with firmware version V60.14.x, V50.14.x, or later.

Example

This example shows that `SWISS721.TTF` is the based font. `ANMDJ.TTF` is the first linked font, and `MSGOTHIC.TTF` is the second linked extension:

ZPL II CODE	GENERA TED LABEL
^XA ^LF ^XZ	

This is the code that established the font links:

```
^XA
^FLE:ANMDJ.TTF,E:SWISS721.TTF,1^FS
^FLE:MSGOTHIC.TTF,E:SWISS721.TTF,1^FS
^XZ
```

^LH

The ^LH command sets the label home position.

Label Home

The default home position of a label is the upper-left corner (position 0,0 along the x and y axis). This is the axis reference point for labels. Any area below and to the right of this point is available for printing. The ^LH command changes this reference point. For instance, when working with preprinted labels, use this command to move the reference point below the preprinted area.

This command affects only fields that come after it. It is recommended to use ^LH as one of the first commands in the label format.

Format: ^LHx,y

Parameters	Details
x = x-axis position (in dots)	**Values:** 0 to 32000 **Initial Value at Power Up:** 0 or last permanently saved value
y = y-axis position (in dots)	**Values:** 0 to 32000 **Initial Value at Power Up:** 0 or last permanently saved value

Depending on the printhead used in your printer, use one of these when figuring the values for x and y:

6 dots = 1 mm, 152 dots = 1 inch

8 dots = 1 mm, 203 dots = 1 inch

11.8 dots = 1 mm, 300 dots = 1 inch

24 dots = 1 mm, 608 dots = 1 inch

Comments: To be compatible with existing printers, this command must come before the first ^FS (Field Separator) command. Once you have issued an ^LH command, the setting is retained until you turn off the printer or send a new ^LH command to the printer.

^LL

The ^LL command defines the length of the label. This command is necessary when using continuous media (media not divided into separate labels by gaps, spaces, notches, slots, or holes). This command is not persistent across a power cycle unless a ^JUS is received.

See also zpl.label_length_always on page 1073.

Label Length

To affect the current label and be compatible with existing printers, ^LL must come before the first ^FS (Field Separator) command. Once you have issued ^LL, the setting is retained until you turn off the printer or send a new ^LL command.

Format: ^LLy,x

Parameters	Details
y =	Defines the label length. **Values:** 1 to 32000, not to exceed the maximum label size. While the printer accepts any value for this parameter, the amount of memory installed determines the maximum length of the label. **Default:** typically set through the LCD (if applicable), or to the maximum label length capability of the printer.
x =	Specifies whether the label length applies to all media, including Gap and Mark. **Values:** N or Y, n or y is also accepted. • N means that the ^LL length applies only to continuous media. • Y means that the ^LL length applies to all media, including Gap and Mark. **Default:** N. If no value is present, the current setting is left unchanged.

Comments: These formulas can be used to determine the value of y:

For 6 dot/mm printheads...	Label length in inches x 152.4 (dots/inch) = y
For 8 dot/mm printheads...	Label length in inches x 203.2 (dots/inch) = y
For 12 dot/mm printheads...	Label length in inches x 304.8 (dots/inch) = y
For 24 dot/mm printheads...	Label length in inches x 609.6 (dots/inch) = y

Values for y depend on the memory size. If the entered value for y exceeds the acceptable limits, the bottom of the label is cut off. The label also shifts down from top to bottom.

If multiple ^LL commands are issued in the same label format, the last ^LL command affects the next label unless it is prior to the first ^FS.

This command is ignored on the HC100™ printer.

^LR

The ^LR command reverses the printing of all fields in the label format. It allows a field to appear as white over black or black over white.

Label Reverse Print

Using the ^LR is identical to placing an ^FR command in all current and subsequent fields.

Format: ^LRa

Parameters	Details
a = reverse print all fields	**Values:** N = no Y = yes N or last permanently saved value

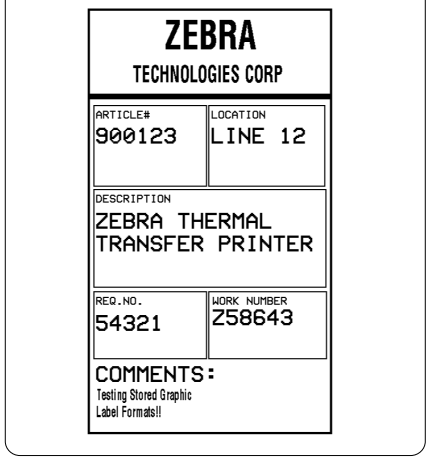
Example

This is an example that shows printing white over black and black over white. The ^GB command is used to create the black background.

ZPL II CODE

```
^XA
^ILR:SAMPLE2.GRF^FS
^CFD,36,20
^FO15,210
^FD900123^FS
^FO218,210
^FDLINE  12^FS
^FO15,360^AD
^FDZEBRA  THERMAL^FS
^FO15,400^AD
^FDTRANSFER  PRINTER^FS
^FO15,540
^FD54321^FS
^FO220,530
^FDZ58643^FS
^FO15,670^A0,27,18
^FDTesting  Stored  Graphic^FS
^FO15,700^A0,27,18
^FDLabel  Formats!!^FS
^XZ
```

GENERA TED LABEL



Comments: The ^LR setting remains active unless turned off by ^LRN or the printer is turned off.

NOTE: ^GB needs to be used together with ^LR.

Only fields following this command are affected.

^LS

The ^LS command allows for compatibility with Z-130 printer formats that are set for less than full label width. It is used to shift all field positions to the left so the same commands used on a Z-130 or Z-220 Printer can be used on other Zebra printers.

Label Shift

To determine the value for the ^LS command, use this formula:

```
Z-130 and Z-220 values for ^LHx + ^FOx
```

(distance from edge of label) = printer value for ^LSa

If the print position is less than 0, set ^LS to 0.

Format: ^LSa

IMPORTANT: The ability to save the ^LS command depends on the version of firmware.

Parameters	Details
a = shift left value (in dots)	**Values:** -9999 to 9999 **Initial Value at Power Up:** 0

Comments: When entering positive values, it is not necessary to use the + sign. The value is assumed to be positive unless preceded by a negative sign (-).

To be compatible with existing Zebra printers, this command must come before the first ^FS (Field Separator) command. Once you have issued an ^LS command, the setting is retained until you turn off the printer or send a new ^LS command to the printer.

^LT

The ^LT command moves the entire label format a maximum of 120 dot rows up or down from its current position, in relation to the top edge of the label. A negative value moves the format towards the top of the label; a positive value moves the format away from the top of the label.

Label Top

This command can be used to fine-tune the position of the finished label without having to change any of the existing parameters.

IMPORTANT: For some printer models, it is possible to request a negative value large enough to cause the media to backup into the printer and become unthreaded from the platen. This condition can result in a printer error or unpredictable results.

Format: ^LTx

Parameters	Details
x = label top (in dot rows)	**Values:** HC100: 0 to 120 XiIIIPlus 600dpi: -240 to 240 All other Zebra printers: -120 to 120 **Default:** a value must be specified or the command is ignored

Comments: The Accepted Value range for x might be smaller depending on the printer platform.

The Label Top value shown on the front panel of the printer is double the value used in the ZPL format.

The ^LT command does not change the media rest position.

^MA

The ^MA command controls how the printer issues printed maintenance alerts. Maintenance alerts are labels that print with a warning that indicates the printhead needs to be cleaned or changed.

Set Maintenance Alerts

.15↑

This command is available only for printers with firmware version V60.15.x, V50.15.x, or later.

- Xi4, RXi4
- ZM400/ZM600, RZ400/RZ600
- S4M with v53.15.5Z or later
- G-Series

IMPORTANT: ^MA settings do not impact or affect the functionality of the Xi4 Supplies Warning system.

Format: `^MAtype,print,printlabel_threshold,frequency,units`

Parameters	Details
`type` = type of alert	**Values:** • `R` = head replacement • `C` = head cleaning **Default:** This parameter must be specified as R or C for `print`, `printlabel_threshold`, and `frequency` to be saved. However, `units` will always be set.
`print` = determines if the alert prints a label	**Values:** • `Y` = print a label • `N` = do not print a label **Default:** `N`
`printlabel threshold`= the distance where the first alert occurs	**Values:** • `R` = head replacement (unit of measurement for head is km with a range of 0 to 150 km) • `C` = clean head with a range of 100 to 2000 meters. • `0` = off (when set to 0, the selected alert is disabled; otherwise, it is enabled. **Default:** `R` = 50 km (1,968,500 inches) and `C` = 0 (off).
`frequency` = distance before reissuing the alert	The unit of measurement is in meters. The range is 0 to 2000. The range for G-Series printers is 0 or 5 to 2000 meters. When set to `0`, the alert label is only printed on power-up or when the printer is reset. **Default:** `0` (print on power-up).

Parameters	Details
units = odometer and printhead maintenance commands	The units parameter reports units of the odometer and printhead maintenance commands, as follows: ~HQOD, ~HQPH, ~WQOD, ~WQPH. **Values:** • C = centimeters (displays as: cm) • I = inches (displays as: ") • M = meters (displays as: M) **Default:** I

Example: This example sets the printed head cleaning message to print after five meters and to repeat every one meter after that until a ~ROC command is issued.

The Early Warning Maintenance setting must be ON. To enable the maintenance alert system on the G-Series™ printer the ^JH command is used; on other Zebra printers, the front panel can also be used.

To set ^MA to print out a label flagging the need to clean the head, type:

```
^XA^MAC,Y,5,1^XZ
```

When the threshold is met, a label will print indicating that the head needs to be cleaned.

For this example, the message on the label looks like this:

For details on resetting the units of measure, see ~HQ examples.

Comments:

Any values outside the specified range are ignored.

The intent of this command is to cause a label to print when the defined threshold is reached.

^MC

In normal operation, the bitmap is cleared after the format has been printed. The ^MC command is used to retain the current bitmap. This applies to current and subsequent labels until cleared with ^MCY.

Map Clear

Format: ^MCa

IMPORTANT: To produce a label template, ^MC must be used with ^FV.

Parameters	Details
a = map clear	**Values:** Y (clear bitmap) or N (do not clear bitmap) **Initial Value at Power Up:** Y

Comments: The ^MC command retains the image of the current label after formatting. It appears in the background of the next label printed.

^MD

The ^MD command adjusts the darkness relative to the current darkness setting.

Media Darkness

Format: ^MDa

Parameters	Details
a = media darkness level	**Values:** -30 to 30, depending on current value **Initial Value at Power Up:** 0 If no value is entered, this command is ignored.

Example: These examples show setting the printer to different darkness levels:

- If the current value (value on configuration label) is 16, entering the command ^MD-9 decreases the value to 7.
- If the current value (value on configuration label) is 1, entering the command ^MD15 increases the value to 16.
- If the current value (value on configuration label) is 25, entering the command ^MD10 increases only the value to 30, which is the maximum value allowed.

Each ^MD command is treated separately in relation to the current value as printed on the configuration label.

NOTE: On Zebra G-Series™ printers the value set with the ^MD command is persistent across power cycles.

IMPORTANT: The darkness setting range for the **&XiIIIPlus;**, Xi4, and RXi4 is 0 to 30 in increments of 0.1. The firmware is setup so that the ^MD and ~SD commands (ZPL darkness commands) accepts that range of settings.

Example: These are examples of the **XiIIIPlus**, Xi4, and RXi4 Darkness Setting:

^MD8.3

~SD8.3

Example: For example, this is what would happen if two ^MD commands were received:

Assume the current value is 15. An ^MD-6 command is received that changes the current value to 9. Another command, ^MD2, is received. The current value changes to 17.

The two ^MD commands are treated individually in relation to the current value of 15.

Comments: The ~SD command value, if applicable, is added to the ^MD command.

^MF

The ^MF command dictates what happens to the media at power-up and at head-close after the error clears.

Media Feed

Format: ^MFp,h

Parameters	Details
p = feed action at power-up	**Values:** F = feed to the first web after sensor C = (see~JC definition) L = (see~JL definition) N = no media feed S = short calibration[1] **Default:** C
h = feed action after closing printhead	**Values:** F = feed to the first web after sensor C = (see ~JC definition) L = (see ~JL definition) N = no media feed S = short calibration[1] **Default:** C

1. These values are supported only on Xi4, RXi4, XiIIIPlus, PAX, ZM400/ZM600, RZ400/RZ600, and S4M printers.

Comments: It is important to remember that if you choose the N setting, the printer assumes that the media and its position relative to the printhead are the same as before power was turned off or the printhead was opened. Use the ^JU command to save changes.

^MI

The ^MI command controls the content of maintenance alert messages, which are reminders printed by the printer to instruct the operator to clean or replace the printhead.

Set Maintenance Information Message

.15↑

This command is available only for printers with firmware version V60.15.x, V50.15.x, or later.

Format: ^MItype,message

Parameters	Details
type = identifies the type of alert	**Values:** • R = head replacement • C = head cleaning **Default:** R
message = message that prints on the label when a maintenance alert occurs	The maximum length of each message is 63 characters. All characters following the comma and preceding the next tilde (~) or carat (^) define the message string. Commas (,) are not allowed in the message. **Default:** • HEAD CLEANING = please clean printhead • HEAD REPLACEMENT = please replace printhead

Example

This example sets the printhead (head) replacement warning message. Printing of this message is controlled by the ^MA command.

1. To customize the text of this label, type something like this:

```
^XA^MIR,PRINT HEAD NEEDS REPLACEMENT - CALL EXT 1000^XZ
```

The label prints whatever you program it to say.

2. For this example, the message on the label looks like this:

^ML

The ^ML command lets you adjust the maximum label length.

Maximum Label Length

NOTE: Note: This command does not apply when in continuous mode.

Format: ^MLa

Parameters	Details
a = maximum label length (in dot rows)	**Values:** the dpi of the printer multiplied by two, up to the maximum length of label **Default:** last permanently saved value

Comments: For calibration to work properly, you must set the maximum label length equal to or greater than your actual label length.

This command is ignored on the HC100™ printer

^MM

The ^MM command determines the action the printer takes after a label or group of labels has printed.

Print Mode

NOTE: Refer to the User Guide for your printer to determine which print modes are supported by your printer.

Format: ^MMa,b

Parameters	Details
a = desired mode	**Values:** T = Tear-off[a] P = Peel-off (not available on **S**-300)[a] R = Rewind (depends on printer model) A = Applicator (depends on printer model) [a] C = Cutter (depends on printer model) D = Delayed cutter [a] F = RFID [a] L = Reserved [a, b] U = Reserved [a, b] K = Kiosk[c] **Default:** The values available for parameter a depend on the printer being used and whether it supports the option. For RFID printers:A = R110PAX4 print enginesF = other RFID printers
b = prepeel select	**Values:** N = no Y = yes **Default:** N The command is ignored if parameters are missing or invalid. The current value of the command remains unchanged. This option is not supported by Link-OS printers.
a. This value is supported on the KR03 or ZD500R printer. b. This value is supported only on the ZM400 and ZM 600 printers. c. This value is supported only on the KR403 printer.	

This list identifies the different modes of operation:

- Tear-off — after printing, the label advances so the web is over the tear bar. The label, with liner attached, can be torn off manually.

- Peel-off — after printing, the label moves forward and activates a Label Available Sensor. Printing stops until the label is manually removed from the printer.
  - Power Peel – liner automatically rewinds using an optional internal rewind spindle.
  - Value Peel – liner feeds down the front of the printer and is manually removed.
  - Prepeel – after each label is manually removed, the printer feeds the next label forward to prepeel a small portion of the label away from the liner material. The printer then backfeeds and prints the label. The prepeel feature assists in the proper peel operation of some media types.
- Rewind — the label and liner are rewound on an (optional) external rewind device. The next label is positioned under the printhead (no backfeed motion).
- Applicator — when used with an application device, the label move far enough forward to be removed by the applicator and applied to an item. This applies only to printers that have applicator ports and that are being used in a print-and-apply system.
- Cutter — after printing, the media feeds forward and is automatically cut into predetermined lengths.
- Delayed cutter — When the printer is in the Delayed Cut PRINT MODE, it will cut the label when it receives the ~JK (Delayed Cut) command. To activate the ~JK command, the printer's PRINT MODE must be set to Delayed Cut and there must be a label waiting to be cut. When the printer is not in the Delayed Cut PRINT MODE, the printer will not cut the label when it receives the ~JK command.

NOTE: Send ~JK in a separate file - it cannot be sent at the end of a set of commands.

The Delayed Cut feature can be activated:

- through PRINT MODE on the printer’s control panel
- with a ^MMD command
- RFID — increases throughput time when printing batches of RFID labels by eliminating backfeed between labels.
- Kiosk — after printing, the media is moved in a presentation position, most applications maintain a loop of media in the printer.

Comments: Be sure to select the appropriate value for the print mode being used to avoid unexpected results.

This command is ignored on the HC100™ printer.

^MN

This command specifies the media type being used and the black mark offset in dots.

Media Tracking

This bulleted list shows the types of media associated with this command:

- Continuous Media – this media has no physical characteristic (such as a web, notch, perforation, black mark) to separate labels. Label length is determined by the ^LL command.
- Continuous Media, variable length – same as Continuous Media, but if portions of the printed label fall outside of the defined label length, the label size will automatically be extended to contain them. This label length extension applies only to the current label. Note that ^MNV still requires the use of the ^LL command to define the initial desired label length.
- Non-continuous Media – this media has some type of physical characteristic (such as web, notch, perforation, black mark) to separate the labels.

Format: ^MNa,b

Parameters	Details
a = media being used	**Values:** N = continuous media Y = non-continuous media web sensing[1, 2] W = non-continuous media web sensing[1, 2] M = non-continuous media mark sensing A = auto-detects the type of media during calibration[1, 3] V = continuous media, variable length[4] **Default:** a value must be entered or the command is ignored
b = black mark offset in dots	This sets the expected location of the media mark relative to the point of separation between documents. If set to 0, the media mark is expected to be found at the point of separation. (i.e., the perforation, cut point, etc.)All values are listed in dots. This parameter is ignored unless the a parameter is set to M. If this parameter is missing, the default value is used. **Values:** -80 to 283 for direct-thermal only printers -240 to 566 for 600 dpi printers -75 to 283 for KR403 printers -120 to 283 for all other printers **Default:** 0

1. Provides the same result.

2. This value is not supported on the KR403 printer.

3. This parameter is supported only on G-series printers.

4. This parameter is supported only on the KR403 printer.

Comments

This command is ignored on the HC100™ printer.

^MP

The ^MP command is used to disable the various mode functions on the control panel. Once disabled, the settings for the particular mode function can no longer be changed and the LED associated with the function does not light.

Mode Protection

Because this command has only one parameter, each mode must be disabled with an individual ^MP command.

Format: ^MPa

Parameters	Details
a = mode to protect	**Values:** D = disable Darkness Mode P = disable Position Mode C = disable Calibration Mode E = enable all modes S = disable all mode saves (modes can be adjusted but values are not saved) W = disable Pause F = disable Feed X = disable Cancel M = disable menu changes **Default:** a value must be entered or the command is ignored

Example: This example shows the ZPL code that disables modes D and C. It also shows the effects on the configuration label before and after the ZPL code is sent:

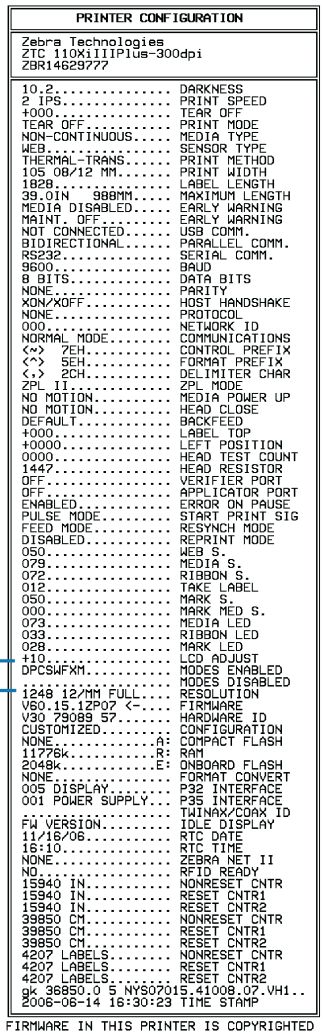
```
^XA
^MPD
^MPC
^XZ
```

Before

```
DPCSWFXM..... MODES ENABLED
............. MODES DISABLED
```

After

```
.P.SWFXM..... MODES ENABLED
D.C.......... MODES DISABLED
```



```
PRINTER CONFIGURATION

Zebra Technologies
ZTC 110Xi3IIIPlus-300dpi
ZBR14629777

10.2.............. DARKNESS
2 IPS............. PRINT SPEED
+000.............. TEAR OFF
TEAR OFF.......... PRINT MODE
NON-CONTINUOUS.... MEDIA TYPE
WEB............... SENSOR TYPE
THERMAL-TRANS..... PRINT METHOD
105 08/12 MM...... PRINT WIDTH
1828.............. LABEL LENGTH
39.0IN   988MM.... MAXIMUM LENGTH
MEDIA DISABLED.... EARLY WARNING
MAINT. OFF........ EARLY WARNING
NOT CONNECTED..... USB COMM.
BIDIRECTIONAL..... PARALLEL COMM.
RS232............. SERIAL COMM.
9600.............. BAUD
8 BITS............ DATA BITS
NONE.............. PARITY
XON/XOFF.......... HOST HANDSHAKE
NONE.............. PROTOCOL
000............... NETWORK ID
NORMAL MODE....... COMMUNICATIONS
<~>  7EH.......... CONTROL PREFIX
<^>  5EH.......... FORMAT PREFIX
<,>  2CH.......... DELIMITER CHAR
ZPL II............ ZPL MODE
NO MOTION......... MEDIA POWER UP
NO MOTION......... HEAD CLOSE
DEFAULT........... BACKFEED
+000.............. LABEL TOP
+0000............. LEFT POSITION
0000.............. HEAD TEST COUNT
1447.............. HEAD RESISTOR
OFF............... VERIFIER PORT
OFF............... APPLICATOR PORT
ENABLED........... ERROR ON PAUSE
PULSE MODE........ START PRINT SIG
FEED MODE......... RESYNCH MODE
DISABLED.......... REPRINT MODE
050............... WEB S.
079............... MEDIA S.
072............... RIBBON S.
012............... TAKE LABEL
050............... MARK S.
000............... MARK MED S.
073............... MEDIA LED
033............... RIBBON LED
028............... MARK LED
+10............... LCD ADJUST
DPCSWFXM.......... MODES ENABLED
.P.SWFXM.......... MODES DISABLED
1248 12/MM FULL... RESOLUTION
V60.15.1ZP07 <-... FIRMWARE
V30 79089 57...... HARDWARE ID
CUSTOMIZED........ CONFIGURATION
NONE.............A: COMPACT FLASH
11776k...........R: RAM
2048k............E: ONBOARD FLASH
NONE.............. FORMAT CONVERT
005 DISPLAY....... P32 INTERFACE
001 POWER SUPPLY... P35 INTERFACE
.................. TWINAX/COAX ID
FW VERSION........ IDLE DISPLAY
11/16/06.......... RTC DATE
16:10............. RTC TIME
NONE.............. ZEBRA NET II
NO................ RFID READY
15940 IN.......... NONRESET CNTR
15940 IN.......... RESET CNTR1
15940 IN.......... RESET CNTR2
39850 CM.......... NONRESET CNTR
39850 CM.......... RESET CNTR1
39850 CM.......... RESET CNTR2
4207 LABELS....... NONRESET CNTR
4207 LABELS....... RESET CNTR1
4207 LABELS....... RESET CNTR2
gk 38850.0 5 NYS07015.41008.07.VH1..
2006-08-14 16:30:23 TIME STAMP

FIRMWARE IN THIS PRINTER IS COPYRIGHTED
```

^MT

The ^MT command selects the type of media being used in the printer.

Media Type

These are the choices for this command:

- Thermal Transfer Media – this media uses a high-carbon black or colored ribbon. The ink on the ribbon is bonded to the media.
- Direct Thermal Media – this media is heat sensitive and requires no ribbon.

Format: ^MTa

Parameters	Details
a = media type used	**Values:** T = thermal transfer mediaD = direct thermal media**Default:** a value must be entered or the command is ignored

Comments

This command is ignored on the HC100™ printer.

^MU

The ^MU command sets the units of measurement the printer uses. ^MU works on a field-by-field basis. Once the mode of units is set, it carries over from field to field until a new mode of units is entered.

Set Units of Measurement

^MU also allows for printing at lower resolutions — 600 dpi printers are capable of printing at 300, 200, and 150 dpi; 300 dpi printers are capable of printing at 150 dpi.

Format: ^MUa,b,c

Parameters	Details
a = units	**Values:** D = dots I = inches M = millimeters **Default:** D
b = format base in dots per inch	**Values:** 150, 200, 300 **Default:** a value must be entered or the command is ignored
c = desired dots-per-inch conversion	**Values:** 300, 600 **Default:** a value must be entered or the command is ignored

Example: This is an example of Setting Units:

Assume 8 dot/millimeter (203 dot/inch) printer.

Field based on dots:

```
^MUd^FO100,100^GB1024,128,128^FS
```

Field based on millimeters:

```
^MUm^FO12.5,12.5^GB128,16,16^FS
```

Field based on inches:

```
^MUi^FO.493,.493^GB5.044,.631,.631^FS
```

Example: This is an example of Converting dpi Values:

^MUd,150,300	Convert a 150 dpi format to a 300 dpi format with a base in dots.
^MUd,150,600	Convert a 150 dpi format to a 600 dpi format with a base in dots.
^MUd,200,600	Convert a 200 dpi format to a 600 dpi format with a base in dots:

To reset the conversion factor to the original format, enter matching values for parameters b and c:

```
^MUd,150,150
^MUd,200,200
^MUd,300,300
```

```
^MUd,600,600
```

Comments: This command should appear at the beginning of the label format to be in proper ZPL II format. To turn the conversion off, enter matching values for parameter b and c.

^MW

The ^MW command allows you to set the head cold warning indicator based on the operating environment.

Modify Head Cold Warning

Format: ^MWa

Parameters	Details
a = enable head cold warning	**Values:** Y = enable head cold warning N = disable head cold warning

IMPORTANT: When a parameter is not given, the instruction is ignored.

^PA

The ^PA command is used to configure advanced text layout features.

Advanced Text Properties

.14↑

This command is available only for printers with firmware version V60.14.x, V50.14.x, or later.

Format: ^PAa,b,c,d

Parameters	Details
a = default glyph	This determines whether the default glyph is a space character or the default glyph of the base font, which is typically a hollow box. **Values:** 0 = off (space as default glyph) 1 = on (default glyph of font is used, often a hollow box, but depends on the font.) **Default:** 0
b = bidirectional text layout	This determines whether the bidirectional text layout is turned on or off. **Values:** 0 = off 1 = on **Default:** 0
c = character shaping	This determines whether character shaping is turned on or off. **Values:** 0 = off 1 = on **Default:** 0
d = OpenType table support	This determines whether the OpenType support is turned on or off. **Values:** 0 = off 1 = on **Default:** 0

^PF

The ^PF command causes the printer to slew labels (move labels at a high speed without printing) a specified number of dot rows from the bottom of the label. This allows faster printing when the bottom portion of a label is blank.

Slew Given Number of Dot Rows

Format: ^PF#

Parameters	Details
# = number of dots rows to slew	**Values:** `0` to `32000` **Default:** a value must be entered or the command is ignored.

^PH ~PH

The ~PH command feeds one label after the format currently being printed is done or when the printer is placed in pause.

Slew to Home Position

The ^PH or ~PH command causes the printer to feed one blank label.

The ^PH command feeds one blank label after the current format prints.

Format: `^PH or ~PH`

~PL

The ~PL command adds an additional amount to how far the paper is ejected during a present cycle. A standard amount of 50 mm is always added to clear the kiosk wall. This amount is added to that 50mm. The total amount of media ejected when a ^PN is executed, then, is 50 mm + ~PL value + ^PN value.

Present Length Addition

Supported Devices:

- KR403

Format: ^PLa

Parameters	Details
a = additional eject length	**Values:** 000-255 = additional mm of media to eject **Default:** 000 The command is ignored if parameters are missing or invalid.

^PM

The ^PM command prints the entire printable area of the label as a mirror image. This command flips the image from left to right.

Printing Mirror Image of Label

Format: ^PMa

Parameters	Details
a = print mirror image of entire label	**Values:** N = no Y = yes **Default:** N

Example: This is an example of printing a mirror image on a label:

ZPL II CODE

```
^XA^PMY
^FO100,100
^CFG
^FDMIRROR^FS
^FO100,160
^FDIMAGE^FS
^XZ
```

GENERATED LABEL

Comments: If the parameter is missing or invalid, the command is ignored. Once entered, the ^PM command remains active until ^PMN is received or the printer is turned off.

~PM

This command places the printer into the Decommissioning Mode.

Decommissioning Mode

Format: ~PMa,b

Parameter	Details
a = printer's serial number	Alphanumeric string. Mandatory parameter.
b = number of times for the flash wipe-out to occur	Optional parameter. Minimum Value = 0 Maximum Value =3 Default Value = 0

NOTE:

- This command is only accepted via USB. It will be ignored if sent over any other communication channel such as Bluetooth or Ethernet. This is done as a security measure to ensure that the person sending the command has physical access to the printer.
- If Decommissioning Mode takes longer than 3 seconds, printers with a screen will display a message while the Decommissioning Mode is in process.
- The serial number specified should match with the serial number of the printer, else the printer does not exit Protected Mode.
- Decommissioning Mode will cause the printer to exit Protected Mode.
- Only use the flash wipe optional parameter if the printer will be resold, recycled, or reused by another group that should not have access to the printer settings data. This may include proprietary fonts, formats, files, or network configuration. A flash wipe does take considerable time, which will vary in length, based on printer model.

RECOMMENDATION: Only use the flash wipe optional parameter if the printer will be resold, recycled, or reused by another group that should not have access to the printer settings data. This may include proprietary fonts, formats, files, or network configuration. A flash wipe takes considerable time, which will vary in length, based on the printer model.

^PN

The ^PN command causes the printer to run a Presenter cycle. The parameter defines the amount of media ejected. The total amount of media ejected when a ^PN is executed, then, is 50 mm + ~PL value + ^PN value. (See ~PL).

Present Now

Supported Devices:

- KR403

Format: ^PNa

Parameters	Details
a = media eject length	**Values:** 0-255 = additional mm of media to eject **Default:** none The command is ignored if parameters are missing or invalid.

^PO

The ^P0 command inverts the label format 180 degrees. The label appears to be printed upside down. If the original label contains commands such as ^LL, ^LS, ^LT and ^PF, the inverted label output is affected differently.

Print Orientation

Format: ^P0a

Parameters	Details
a = invert the label 180 degrees	**Values:** N = normal I = invert **Default:** N

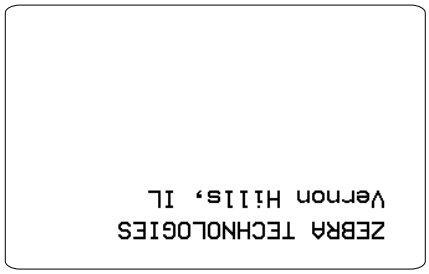
Example

This is an example of printing a label at 180 degrees:

ZPL II CODE

```
^XA^CFD
^POI
^LH330,10
^FO50,50
^FDZEBRA  TECHNOLOGIES^FS
^FO50,75
^FDVernon  Hills,  IL^FS
^XZ
```

GENERA TED LABEL



The ^POI command inverts the x,y coordinates. All image placement is relative to these inverted coordinates. Therefore, a different ^LH (Label Home) can be used to move the print back onto the label.

Comments: If multiple ^P0 commands are issued in the same label format; only the last command sent to the printer is used.

Once the ^P0 command is sent, the setting is retained until another ^P0 command is received, or the printer is turned off.

The N value for the a parameter is not supported on the HC100™ printer.

^PP ~PP

The ~PP command stops printing after the current label is complete (if one is printing) and places the printer in Pause Mode.

Programmable Pause

The ^PP command is not immediate. Therefore, several labels might print before a pause is performed. This command pauses the printer after the current format prints.

The operation is identical to pressing **PAUSE** on the control panel of the printer. The printer remains paused until **PAUSE** is pressed or a ~PS (Print Start) command is sent to the printer.

Format: `^PP or ~PP`

^PQ

The ^PQ command gives control over several printing operations. It controls the number of labels to print, the number of labels printed before printer pauses, and the number of replications of each serial number.

Print Quantity

Format: ^PQq,p,r,o,e

Parameters	Details
q = total quantity of labels to print	**Values:** 1 to 99,999,999 **Default:** 1
p = pause and cut value (labels between pauses)	**Values:** 1 to 99,999,999 **Default:** 0 (no pause)
r = replicates of each serial number	**Values:** 0 to 99,999,999 replicates **Default: :** 0 (no replicates)
o = override pause count	**Values:** N = no Y = yes **Default:** N
e = cut on error label (RFID void is an error label)	**Values:** N = no - if a cutter is installed, a cut will be made after a voided RIFD label ONLY if a cut would be made after the non-voided label and this was the last retry. Y = yes - if a cutter is installed, a cut will be made after ANY voided RFID label. **Default:** Y

If the o parameter is set to Y, the printer cuts but does not pause, and the printer does **not** pause after every group count of labels has been printed. With the o parameter set to N (default), the printer pauses after every group count of labels has been printed.

Example: This example shows the control over print operations:

^PQ50,10,1,Y: This example prints a total of 50 labels with one replicate of each serial number. It prints the total quantity in groups of 10, but does not pause after every group.

^PQ50,10,1,N: This example prints a total of 50 labels with one replicate of each serial number. It prints the total quantity in groups of 10, pausing after every group.

~PR

If the ~PR command is enabled, the last label printed reprints, similar to the applicator asserting the Reprint signal on the applicator port.

Applicator Reprint

This command is similar to device.applicator.reprint on page 668. See also ^JJ.

Format: ~PR

Comments: Pressing **PREVIOUS** on the control panel also causes the last label to reprint.

^PR

The ^PR command determines the media and slew speed (feeding a blank label) during printing.

Print Rate

The printer operates with the selected speeds until the setting is reissued or the printer is turned off.

The print speed is application-specific. Because print quality is affected by media, ribbon, printing speeds, and printer operating modes, it is very important to run tests for your applications.

NOTE: important: Some models go to default print speed when power is turned off.

^PRp,s,b

Parameters	Details
p = print speed	**Values:** 1 = 25.4 mm/sec. (1 inch/sec.)[1] A or 2 = 50.8 mm/sec. (2 inches/sec.) B or 3 = 76.2 mm/sec. (3 inches/sec.) C or 4 = 101.6 mm/sec. (4 inches/sec.) 5 = 127 mm/sec.(5 inches/sec.) D or 6 = 152.4 mm/sec. (6 inches/sec.) 7 = 177.8 mm/sec. (7 inches/sec.) E or 8 = 203.2 mm/sec. (8 inches/sec.) 9 = 220.5 mm/sec. 9 inches/sec.) 10 = 245 mm/sec.(10 inches/sec.) 11 = 269.5 mm/sec.(11 inches/sec.) 12 = 304.8 mm/sec. 12 inches/sec.) 13 = 13 in/sec[2] 14 = 14 in/sec[2] **Default:** A

Parameters	Details
s = slew speed	**Values:** A or 2 = 50.8 mm/sec. (2 inches/sec.) B or 3 = 76.2 mm/sec. (3 inches/sec.) C or 4 = 101.6 mm/sec. (4 inches/sec.) 5 = 127 mm/sec. 5 inches/sec.) D or 6 = 152.4 mm/sec. (6 inches/sec.) 7 = 177.8 mm/sec. (7 inches/sec.) E or 8 = 203.2 mm/sec. (8 inches/sec.) 9 = 220.5 mm/sec. (9 inches/sec.) 10 = 245 mm/sec. (10 inches/sec.) 11 = 269.5 mm/sec. 11 inches/sec.) 12 = 304.8 mm/sec. 12 inches/sec.) 13 = 13 in/sec^2 14 = 14 in/sec^2 **Default:** D
b = backfeed speed	**Values:** A or 2 = 50.8 mm/sec. (2 inches/sec.) B or 3 = 76.2 mm/sec. (3 inches/sec.) C or 4 = 101.6 mm/sec. (4 inches/sec.) 5 = 127 mm/sec.(5 inches/sec.) D or 6 = 152.4 mm/sec. (6 inches/sec.) 7 = 177.8 mm/sec. (7 inches/sec.) E or 8 = 203.2 mm/sec. (8 inches/sec.) 9 = 220.5 mm/sec. 9 inches/sec.) 10 = 245 mm/sec. 10 inches/sec.) 11 = 269.5 mm/sec. 11 inches/sec.) 12 = 304.8 mm/sec. 12 inches/sec.) 13 = 13 in/sec14 = 14 in/sec^2 **Default:** A

1. This value is supported only on the 110Xi4-600dpi, 110XiIIIPlus-600dpi , and RXi printers.

2. This value is supported only on the Xi4 and RXi4 printers.

Comments: The speed setting for p, s, and b is dependent on the limitations of the printer. If a particular printer is limited to a rate of 6 ips (inches per second), a value of 12 can be entered but the printer performs only at a 6 ips rate. See your printer's User Guide for specifics on performance.

This command is ignored on the HC100 printer.

~PS

The ~PS command causes a printer in Pause Mode to resume printing. The operation is identical to pressing **PAUSE** on the control panel of the printer when the printer is already in Pause Mode.

Print Start

~PS

^PW

The ^PW command allows you to set the print width.

Print Width

^PWa

Parameters	Details
a = label width (in dots)	**Values:** 2, to the width of the label If the value exceeds the width of the label, the width is set to the label's maximum size. **Default:** last permanently saved value

Comments

This command is ignored on the HC100™ printer.

~RO

The ~R0 command resets the advanced counters used by the printer to monitor label generation in inches, centimeters, and number of labels.

Reset Advanced Counters

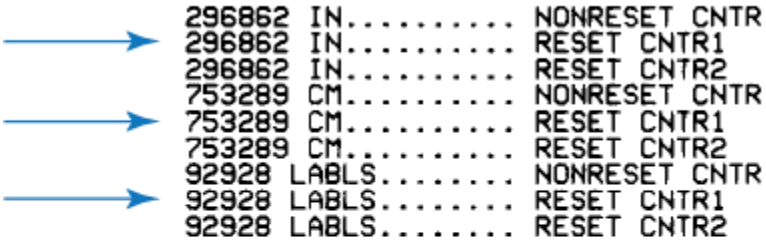
Format: ~R0c

Parameters	Details
c = counter number	**Values:** 1 = reset counter 1 2 = reset counter 2 3 = reset valid RFID label counter 4 = reset voided RFID label counter C = reset head cleaned counter[1] R = reset head replaced counter and head cleaned counter[1] **Default:** a value must be specified or the command is ignored

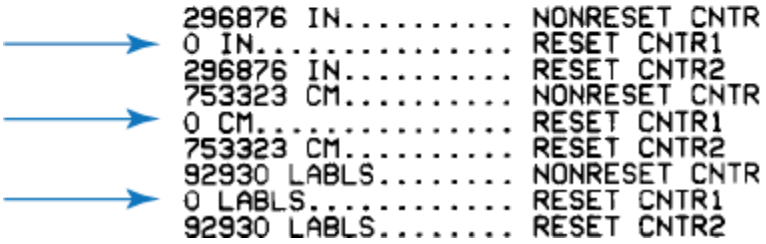
1. These values are supported only on Xi4, RXi4, ZM400/ZM600, RZ400/RZ600, S4M, and G-Series printers.

This example shows how the counter portion of the printer configuration labels looks when counter 1 is reset by sending~R01.

Before



After



Example: This example shows how the counter portion of the printer configuration labels looks when the RFID counters are reset by sending~R03 and ~R04.

Before

```
TM:M6E MICRO..... RFID READER
20.00.00.01..... RFID HW VERSION
01.01.00.EA..... RFID FW VERSION
USA/CANADA..... RFID REGION CODE
USA/CANADA..... RFID COUNTRY CODE
RFID OK..... RFID ERR STATUS
16..... RFID READ PWR
16..... RFID WRITE PWR
F0..... PROG. POSITION
507..... RFID VALID CTR
4..... RFID VOID CTR
```

After

```
TM:M6E MICRO..... RFID READER
20.00.00.01..... RFID HW VERSION
01.01.00.EA..... RFID FW VERSION
USA/CANADA..... RFID REGION CODE
USA/CANADA..... RFID COUNTRY CODE
RFID OK..... RFID ERR STATUS
16..... RFID READ PWR
16..... RFID WRITE PWR
F0..... PROG. POSITION
0..... RFID VALID CTR
0..... RFID VOID CTR
```

^SC

The ^SC command allows you to change the serial communications parameters you are using.

Set Serial Communications

Format: ^SCa,b,c,d,e,f

Parameters	Details
a = baud rate	**Values:** 110[1]; 300; 600; 1200; 2400; 4800; 9600; 14400; 19200; 28800; 38400; or 57600; 115200 **Default:** must be specified or the parameter is ignored
b = word length (in data bits)	**Values:** 7 or 8 **Default:** must be specified
c = parity	**Values:** N (none), E (even), or O (odd) **Default:** must be specified
d = stop bits	**Values:** 1 or 2 **Default:** must be specified
e = protocol mode	**Values:** X = XON/XOFF D = DTR/DSR R = RTS M = DTR/DSR XON/XOFF[2] **Default:** must be specified
f = Zebra protocol	**Values:** A = ACK/NAK N = none Z = Zebra **Default:** must be specified

1. This value is not supported on Xi4, RXi4, ZM400/ZM600, RZ400/RZ600, and S4M printers.

2. This parameter is supported only on G-Series printers. Using the DTR/DSR XON/XOFF mode will cause the printer to respond to either DTR/DSR or XON/XOFF, depending on which method is first received from the host device.

Comments: If any of the parameters are missing, out of specification, not supported by a particular printer, or have a ZPL-override DIP switch set, the command is ignored.

A ^JUS command causes the changes in Communications Mode to persist through power-up and software resets.

~SD

The ~SD command allows you to set the darkness of printing. ~SD is the equivalent of the darkness setting parameter on the control panel display.

Set Darkness

Format: ~SD##

Parameters	Details
## = desired darkness setting (two-digit number)	**Values:** 00 to 30 **Default:** last permanently saved value

IMPORTANT: The darkness setting range for the XiIIIPlus, Xi4, and RXi4 is 0 to 30 in increments of 0.1. The firmware is setup so that the ^MD and ~SD commands (ZPL darkness commands) accept that range of settings.

Example: These are examples of the XiIIIPlus, Xi4, and RXi4 Darkness Setting:

```
^MD8.3
~SD8.3
```

Comments: The ^MD command value, if applicable, is added to the ~SD command.

^SE

The ^SE command is used to select the desired ZPL or ZPL II encoding table.

Select Encoding Table

^SEd:o.x

Parameters	Details
d = location of encoding table	**Values:** R:, E:, B:, and A: **Default:** R:
o = name of encoding table	**Values:** 1 to 8 alphanumeric characters **Default:** a value must be specified.
x = extension	**Fixed Value:** .DAT

The encoding tables are provided with the font card or downloaded in flash with the font. The table appears as XXXXXXX.DAT in a directory label printed by the ZPL commands.

The most active encoding table is indicated by the * on the directory label.

Example:

```
^XA^WD*:*.*^XZ
```

^SF

The ^SF command allows you to serialize a standard ^FD string. The maximum size of the mask and increment string is 3K combined.

Serialization Field (with a Standard ^FD String)

.14↑

In firmware version x.14 and later, strings are serialized from the last character in the backing store with regard to the alignment of the mask and increment strings. For combining semantic clusters that do not get incremented, the mask character % needs to be added to the increment string.

Format: ^SFa,b

Parameters	Details
a = mask string	The mask string sets the serialization scheme. The length of the string mask defines the number of characters (or in firmware version x.14 and later, combining semantic clusters) in the current ^FD string to be serialized. The mask is aligned to the characters (or in firmware version x.14 and later, combining semantic clusters) in the ^FD string starting with the right-most (or in firmware x.14 and later, last) in the backing store position. **Mask String placeholders:**D or d – Decimal numeric 0–9 H or h – Hexadecimal 0–9 plus a-f or A-F O or o – Octal 0–7 A or a – Alphabetic A–Z or a–z N or n – Alphanumeric 0–9 plus A–Z or a–z % – Ignore character or skip
b = increment string	The increment string is the value to be added to the field on each label. The default value is equivalent to a decimal value of one. The string is composed of any characters (or in firmware version x.14 and later, combining semantic clusters) defined in the serial string. Invalid characters (or in firmware version x.14 and later, combining semantic clusters) are assumed to be equal to a value of zero in that characters (or in firmware version x.14 and later, combining semantic clusters) position. The increment value for alphabetic strings starts with 'A' or 'a' as the zero placeholder. This means to increment an alphabetic character (or in firmware version x.14 and later, combining semantic cluster) by one, a value of 'B' or 'b' must be in the increment string.

For characters that do not get incremented, the % character needs to be added to the increment string.

Example: This is an example of serializing a ^FD string. The ZPL II code generates three separate labels as seen in Generated Labels:

ZPL II CODE	GENERA TED LABELS
^XA ^FO100,100 ^CF0,100 ^FD12A^SFnnA,F^FS ^PQ3 ^XZ	12K 12F 12A

This mask has the first characters (or in firmware version x.14 and later, the first combining semantic clusters) as alphanumeric (nn = 12), and the last digit is uppercase alphabetic (A). The decimal value of the increment number is equivalent to 5 (F). The number of labels generated depends on the number specified by the ^PQ command.

In a similar instance, the ^FD string could be replaced with either of the ^FD strings below to generate a series of labels, determined by ^PQ.

Using this ZPL code:

```
^FDBL0000^SFAAdddd,1
```

The print sequence on this series of labels is:

```
BL0000, BL0001,...BL0009, BL0010,...
BL0099, BL0100,...BL9999, BM0000...
```

Using this ZPL code:

```
^FDBL00-0^SFAAdd%d,1%1
```

The print sequence on this series of labels is:

```
BL00-0, BL01-1, BL02-2,...BL09-9,
BL11-0, BL12-1...
```

.14↑

Important notes about masking for firmware version V60.14.x, V50.14.x, or later:

- A single % masks an entire combining semantic cluster rather than a single code point.
- The mask string and increment string should be aligned at the last code point in their respective backing stores.
- Control and bidirectional characters do not require a mask and are ignored for serialization purposes.

The following examples show the importance of capitalization and location within the mask.

Example 1

In this example, the printer cycles with every two printed labels and alternates between H (position 18), and then Z (position 36). With n or N, the serial number increments from 0 - 9 and a–z or A–Z (36 positions overall). With each completed cycle, the second cluster (nn) increments one position (from 00, 01, 02 ...) per cycle:

ZPL II CODE	GENERATED LABELS
^XA ^FO100,50^A0N,50,50^FDzzZ^SFnnN,I^FS ^PQ10 ^XZ	

Example 2

In this example, lowercase i increments with a mask string of nnN. Nothing changes because the first cluster (Z) never triggers the second cluster (zz) to change.

ZPL II CODE	GENERATED LABELS
^XA **^FO100,50^A0N,50,50^FDzzZ^SFnnN,i^FS** **^PQ10** **^XZ**	

^SI

The ^SI command is used to change the values for the media sensors, which are also set during the media calibration process. The media calibration process is described in your specific printer's user's guide.

Set Sensor Intensity

This command is available only for printers with firmware versions V53.15.x or later.

Format: ^SIa,b

Parameters	Details
a = indicates the setting to modify	**Values:** 1 = transmissive sensor brightness setting 2 = transmissive sensor baseline setting **Default:** must be an accepted value or the entire command is ignored
b = the value to use for the sensor being configured	The ranges for this parameter are the same for the accepted values in parameter a. **Values:** 0 to 196 **Default:** must be an accepted value or the entire command is ignored

^SL

The ^SL command is used to specify the Real-Time Clock's mode of operation and language for printing information.

Set Mode and Language (for Real-Time Clock)

Format: `^SLa,b`

Parameters	Details
a = mode	**Values:** S = Start Time Mode. This is the time that is read from the Real-Time Clock when label formatting begins (when ^XA is received). The first label has the same time placed on it as the last label. T = Time Now Mode. This is the time that is read from the Real-Time Clock when the label to be printed is placed in print queue. **Time Now** is similar to a serialized time or date field. `Numeric Value` = With the Enhanced Real Time Clock (V60.13.0.10 or later) a time accuracy tolerance can be specified. Range = 1 to 999 seconds, 0 = one second tolerance `SL30,1` = Accuracy tolerance of 30 seconds and use English. **Default:** S
b = language Value 13 is only supported in firmware versions V60.14.x, V50.14.x, or later.	**Values:** 1 = English 2 = Spanish 3 = French 4 = German 5 = Italian 6 = Norwegian 7 = Portuguese 8 = Swedish 9 = Danish 10 = Spanish 2 11 = Dutch 12 = Finnish1 3 = Japanese 14 = Korean 15 = Simplified Chinese 16 = Traditional Chinese 17 = Russian 18 = Polish **Default:** the language selected with ^KL or the control panel

Comments: These are some comments to be aware of:

- The **^SL** command must be placed before the first **^FO** command.
- As of V60.13.0.10 all supported printers have Enhanced Real Time Clock capabilities the RTC will not print time fields that are more than sixty seconds old, rather it will update the time prior to printing (^SLT or ^SL60). To control time with increments other than sixty seconds the ^SL command can be used with a numeric value (^SL30). ^SLS can keep times longer than sixty seconds.

For more details on set mode and language with the Real-Time Clock, see Real Time Clock on page 1590.

^SN

The ^SN command allows the printer to index data fields by a selected increment or decrement value, making the data fields increase or decrease by a specified value each time a label is printed. This can be performed on 100 to 150 fields in a given format and can be performed on both alphanumeric and barcode fields. A maximum of 12 of the right- most integers are subject to indexing.

Serialization Data

In x.13 and earlier, the first integer found when scanning from right to left starts the indexing portion of the data field.

In x.14 and later, the first integer found when scanning from the end of the backing store towards the beginning starts the indexing portion of the data field.

In x.13 and earlier, if the alphanumeric field to be indexed ends with an alpha character, the data is scanned, character by character, from right to left until a numeric character is encountered. Serialization takes place using the value of the first number found.

In x.14 and later, if the backing store of the alphanumeric field to be indexed ends with an alpha character, the data is scanned, character by character, from the end of the backing store until a numeric character is encountered. Serialization takes place using the value of the first number found.

Format: ^SNv,n,z

Parameters	Details
v = starting value	**Values:** 12-digits maximum for the portion to be indexed **Default:** 1
n = increment or decrement value	**Values:** 12-digit maximum **Default:** 1 To indicate a decrement value, precede the value with a minus (–) sign.
z = add leading zeros (if needed)	**Values:** N = no Y = yes **Default:** N

Example: This example shows incrementing by a specified value:

ZPL II CODE	GENERATED LABELS
^XA ^FO260,110 ^CFG ^SN001,1,Y^FS ^PQ3 ^XZ Note: The ZPL II code above will generate three separate labels, see to the right.	001 002 003

Comments: Incrementing and decrementing takes place for each serial-numbered field when all replicates for each serial number have been printed, as specified in the parameter r of the ^PQ (print quality) command.

If, during the course of printing serialized labels, the printer runs out of either paper or ribbon, the first label printed (after the media or ribbon has been replaced and calibration completed) has the same serial number as the **partial** label printed before the **out** condition occurred. This is done in case the last label before the **out** condition did not fully print. This is controlled by the ^JZ command.

Using Leading Zeros

In the ^SN command, the z parameter determines if leading zeros are printed or suppressed. Depending on which value is used (Y = print leading zeros; N = do not print leading zeros), the printer either prints or suppresses the leading zeros.

The default value for this parameter is N (do not print leading zeros).

Print Leading Zeros

.13↓

In x.13 and earlier, the starting value consists of the right-most consecutive sequence of digits.

.14↑

In x.14 and later, the starting value consists of the first number working backward in the backing store consecutive sequence of digits.

The width (number of digits in the sequence) is determined by scanning from right to left until the first non-digit (space or alpha character) is encountered. To create a specific width, manually place leading zeros as necessary.

Suppressing Leading Zeros

.13↓

In x.13 and earlier, the starting value consists of the right-most consecutive sequence of digits, including any leading spaces.

.14↑

In x.14 or later, the starting value consists of the first number working backward in the backing store consecutive sequence of digits, including any leading spaces.

^SO

The ^S0 command is used to set the secondary and the tertiary offset from the primary Real-Time Clock.

Set Offset (for Real-Time Clock)

NOTE: For each label only one ^S02 command can be used. If more than one offset is required, ^S03 must be used.

Format: ^S0a,b,c,d,e,f,g

Parameters	Details
a = clock set	**Values:** 2 = secondary 3 = third **Default:** value must be specified
b = months offset	**Values:** –32000 to 32000 **Default:** 0
c = days offset	**Values:** –32000 to 32000 **Default:** 0
d = years offset	**Values:** –32000 to 32000 **Default:** 0
e = hours offset	**Values:** –32000 to 32000 **Default:** 0
f = minutes offset	**Values:** –32000 to 32000 **Default:** 0
g = seconds offset	**Values:** –32000 to 32000 **Default:** 0

For more detail on set offset, see Real Time Clock on page 1590.

^SP

The ^SP command allows a label to start printing at a specified point before the entire label has been completely formatted. On extremely complex labels, this command can increase the overall throughput of the print.

Start Print

The command works as follows: Specify the dot row at which the ^SP command is to begin. This creates a label segment. Once the ^SP command is processed all information in that segment prints. During the printing process, all of the commands after the ^SP continue to be received and processed by the printer.

If the segment after the ^SP command (or the remainder of the label) is ready for printing, and media motion does not stop. If the next segment is not ready, the printer stops mid-label and waits for the next segment to be completed. Precise positioning of the ^SP command requires a trial-and-error process, as it depends primarily on print speed and label complexity.

The ^SP command can be effectively used to determine the worst possible print quality. You can determine whether using the ^SP command is appropriate for the particular application by using this procedure.

If you send the label format up to the first ^SP command and then wait for printing to stop before sending the next segment, the printed label is a sample of the worst possible print quality. It drops any field that is out of order.

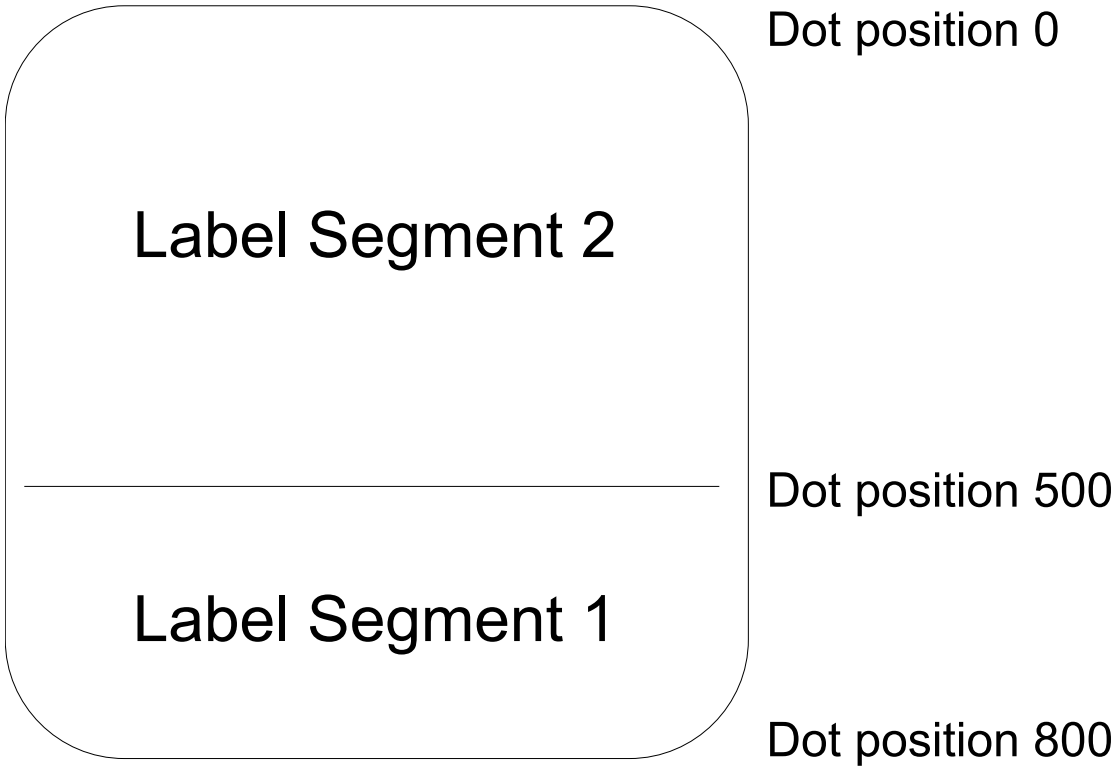
If the procedure above is used, the end of the label format must be:

```
^SP#^FS
```

Format: ^SPa

Parameters	Details
a = dot row to start printing	**Values:** `0` to `32000` **Default:** `0`

Example: In this example, a label 800 dot rows in length uses ^SP`500`. Segment 1 prints while commands in Segment 2 are being received and formatted.


Label Segment 2
Label Segment 1
Dot position 0
Dot position 500
Dot position 800

^SQ

The ^SQ command is used to stop the ZebraNet Alert option.

Halt ZebraNet Alert

Format: ^SQa,b,c

Parameters	Details
a = condition type	**Values:** A = paper out B = ribbon out C = printhead over-temp D = printhead under-temp E = head open F = power supply over-temp G = ribbon-in warning (Direct Thermal Mode) H = rewind full I = cut error J = printer paused K = PQ job completed L = label ready M = head element out N = ZBI (Zebra BASIC Interpreter) runtime error O = ZBI (Zebra BASIC Interpreter) forced error Q = clean printhead R = media low S = ribbon low T = replace head U = battery low V = RFID error (in RFID printers only) W = all errors (in RFID printers only) * = all errors (in non-RFID printers)
b = destination	**Values:** A = serial port B = parallel port C = e-mail address D = TCP/IP E = UDP/IP F = SNMP trap * = wild card to stop alerts for all destinations

Parameters	Details
c = halt messages	**Values:** Y = halt messages N = start messages **Default:** Y

^SR

The ^SR command allows you to set the printhead resistance.

Set Printhead Resistance

Format: ^SR####

Parameters	Details
#### = resistance value (four-digit numeric value)	**Values:** 0488 to 1175 **Default:** last permanently saved value

Comments: To avoid damaging the printhead, this value should be less than or equal to the value shown on the printhead being used. Setting a higher value could damage the printhead.

NOTE: New printer models automatically set head resistance.

^SS

The ^SS command is used to change the values for media, web, ribbon, and label length set during the media calibration process. The media calibration process is described in your specific printer's user's guide.

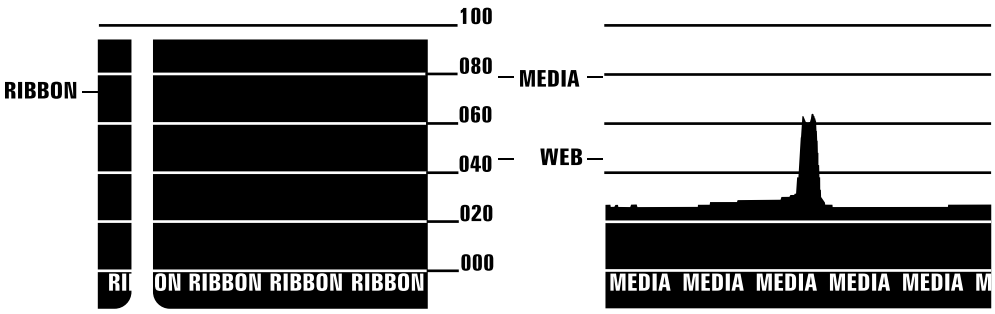
Set Media Sensors

Format: ^SSw,m,r,l,m2,r2,a,b,c

Parameters	Details
w = web (3-digit value)	**Values:** 000 to 100 **Default:** the value is shown on the media sensor profile or configuration label
m = media (3-digit value)	**Values:** 000 to 100 **Default:** the value shown on the media sensor profile or configuration label
r = ribbon (3-digit value)	**Values:** 000 to 100 **Default:** the value is shown on the media sensor profile or configuration label
l = label length (in dots, four-digit value)	**Values:** 0001 to 32000 **Default:** value calculated in the calibration process
m2 = intensity of media LED (3-digit value)	**Values:** 000 to 255 **Default:** value calculated in the calibration process
r2 = intensity of ribbon LED (3-digit value)	**Values:** 000 to 255 **Default:** value calculated in the calibration process
a = mark sensing (3-digit value)	**Values:** 000 to 100 **Default:** value calculated in the calibration process
b = mark media sensing (3-digit value)	**Values:** 000 to 100 **Default:** value calculated in the calibration process
c = mark LED sensing (3-digit value)	**Values:** 000 to 255 **Default:** value calculated in the calibration process

Example

Below is an example of a media sensor profile. Notice the numbers from 000 to 100 and where the words WEB, MEDIA, and RIBBON appear in relation to those numbers. Also, notice the black vertical spike. This represents where the printer sensed the transition from media to web to media.



The media and sensor profiles produced vary in appearance from printer to printer.

Comments: The `m2` and `r2` parameters have no effect in Stripe® **S**-300 and **S**-500 printers. This command is ignored on the HC100™ printer. Maximum values for parameters depend on which printer platform is being used.

^ST

The ^ST command sets the date and time of the Real-Time Clock.

Set Date and Time (for Real-Time Clock)

Format: `^STa,b,c,d,e,f,g`

Parameters	Details
a = month	**Values:** 01 to 12 **Default:** current month
b = day	**Values:** 01 to 31 **Default:** current day
c = year	**Values:** 1998 to 2097 **Default:** current year
d = hour	**Values:** 00 to 23 **Default:** current hour
e = minute	**Values:** 00 to 59 **Default:** current minute
f = second	**Values:** 00 to 59 **Default:** current second
g = format	**Values:** A = a.m. P = p.m. M = 24-hour military **Default:** M

For more details on set date and time, see Real Time Clock on page 1590.

^SX

The ^SX command is used to configure the ZebraNet Alert System.

Set ZebraNet Alert

Format: ^SXa,b,c,d,e,f

NOTE: The values in this table apply to firmware version V48.12.4 or later.

Parameters	Details
a = condition type	**Values:** A = paper out B = ribbon out C = printhead over-temp D = printhead under-temp E = head open F = power supply over-temp G = ribbon-in warning (Direct Thermal Mode) H = rewind full I = cut error J = printer paused K = PQ job completed L = label ready M = head element out N = ZBI (Zebra BASIC Interpreter) runtime error O = ZBI (Zebra BASIC Interpreter) forced error P = power on Q = clean printhead R = media low S = ribbon low T = replace head U = battery low V = RFID error (in RFID printers only) * = all errors **Default:** if the parameter is missing or invalid, the command is ignored

Parameters	Details
b = destination for route alert	**Values:** A = serial port B* = parallel port C = e-mail address D = TCP/IP E = UDP/IP F = SNMP trap **Default:** If this parameter is missing or invalid, the command is ignored. * **Requires** bidirectional communication.
c = enable condition set alert to this destination	**Values:** N = no Y = yes **Values:** Y or previously configured value
d = enable condition clear alert to this destination	**Values:** N = no Y = yes **Values:** N or previously configured value Parameters e and f are sub-options based on destination. If the sub-options are missing or invalid, these parameters are ignored.
e = destination setting	**Values:** Internet e-mail address (e.g. user@company.com) IP address (for example, 10.1.2.123) SNMP trap IP or IPX addresses
f = port number	**Values:** TCP port # (0 to 65535) UPD port # (0 to 65535)

Example: This is an example of the different (b) destinations that you can send for the condition type (a):

Serial: ^SXA,A,Y,Y

Parallel: ^SXA,B,Y,Y

E-Mail: ^SXA,C,Y,Y,admin@company.com

TCP: ^SXA,D,Y,Y,123.45.67.89,1234

UDP: ^SXA,E,Y,Y,123.45.67.89,1234

SNMP Trap: ^SXA,F,Y,Y,255.255.255.255

Comments: In the example above for SNMP Trap, entering 255.255.255.255 broadcasts the notification to every SNMP manager on the network. To route the device to a single SNMP manager, enter a specific address (123.45.67.89).

^SZ

The ^SZ command is used to select the programming language used by the printer. This command gives you the ability to print labels formatted in both ZPL and ZPL II.

Set ZPL Mode

This command remains active until another ^SZ command is sent to the printer or the printer is turned off.

Format: ^SZa

Parameters	Details
a = ZPL version	**Values:** 1 = ZPL 2 = ZPL II **Default:** 2

Comments: If the parameter is missing or invalid, the command is ignored.

~TA

The ~TA command lets you adjust the rest position of the media after a label is printed, which changes the position at which the label is torn or cut.

Tear-off Adjust Position

Format: ~TA###

IMPORTANT: These are some important facts about this command:

- For 600 dpi printers, the step size doubles.
- If the number of characters is **less than** 3, the command is ignored.

Parameters	Details
### = change in media rest position (3-digit value in dot rows must be used.)	**Values:** `-120` to`120` `0` to `120` (on the HC100) **Default:** last permanent value saved

Comments: If the parameter is missing or invalid, the command is ignored.

^TB

The ^TB command prints a text block with defined width and height. The text block has an automatic word-wrap function. If the text exceeds the block height, the text is truncated. This command supports complex text layout features.

Text Blocks

.14↑

This command is available only for printers with firmware version V60.14.x, V50.14.x, or later.

NOTE: ^TB is the preferred command for printing fields or blocks of text, instead of ^FB.

Format: ^TBa,b,c

Parameters	Details
a = block rotation	**Values:** N = normal R = rotate 90 degrees clockwise I = invert 180 degrees B = read from bottom up-270 degrees **Default:** whatever was specified by the last ^A (which has the default of ^FW)
b = block width in dots	**Values:** 1 to the width of the label in dots **Default:** 1 dot
c = block height in dots	**Values:** 1 to the length of the label in dots **Default:** 1 dot

Comments: Facts about the ^TB command:

- Justification of ^TB command comes from ^FO, ^FT, or ^FN command. If no justification is determined then the default is auto justification.
- Data between < and > is processed as an escape sequence; for example, << > will print <.
- The ^TB command has an automatic word-wrap function. Soft hyphens do not print and are not used as a line break position.

^TO

The ^T0 command is used to copy an object or group of objects from one storage device to another. It is similar to the copy function used in PCs.

Transfer Object

Source and destination devices must be supplied and must be different and valid for the action specified. Invalid parameters cause the command to be ignored.

The asterisk (*) can be used as a wild card for object names and extensions. For instance, ZEBRA.* or *.GRF are acceptable forms for use with the ^T0 command.

At least one source parameter (d, o, or x) and one destination parameter (s, o, or x) must be specified. If only ^T0 is entered, the command is ignored.

Format: ^T0s:o.x,d:o.x

Parameters	Details
s = source device of stored object	**Values:** R:, E:, B:, and A: **Default:** if a drive is not specified, all objects are transferred to the drive set in parameter s
o = stored object name	**Values:** any existing object conforming to Zebra conventions **Default:** if a name is not specified, * is used — all objects are selected
x = extension	**Values:** any extension conforming to Zebra conventions **Default:** if an extension is not specified, * is used — all extensions are selected
d = destination device of the stored object	**Values:** R:, E:, B:, and A: **Default:** a destination must be specified
o = name of the object at destination	**Values:** up to 8 alphanumeric characters **Default:** if a name is not specified, the name of the existing object is used
x = extension	**Values:** any extension conforming to Zebra conventions **Default:** if an extension is not specified, the extension of the existing object is used

Comments: Parameters o, x, and s support the use of the wild card (*).

If the destination device does not have enough free space to store the object being copied, the command is canceled.

Zebra files (Z:*.*) cannot be transferred. These files are copyrighted by Zebra Technologies.

Transferring Objects

These are some examples of using the ^T0 command.

Example: To copy the object ZL0G0.GRF from DRAM to an optional Memory Card and rename it ZL0G01.GRF, write the following format:

```
^XA
^T0R:ZL0G0.GRF,B:ZL0G01.GRF
^XZ
```

Example: To copy the object SAMPLE.GRF from an optional Memory Card to DRAM and keep the same name, write this format:

```
^XA
^TOB:SAMPLE.GRF,R:SAMPLE.GRF
^XZ
```

Transferring Multiple Objects

The asterisk (*) can be used to transfer multiple object files (except *.FNT) from DRAM to the Memory Card. For example, assume you have several object files that contain logos. These files are named LOGO1.GRF, LOGO2.GRF, and LOGO3.GRF.

To transfer all these files to the memory card using the name NEW instead of LOGO, place an asterisk after the names NEW and LOGO in the transfer command. This copies all files beginning with LOGO in one command.

```
^XA
^TOR:LOGO*.GRF,B:NEW*.GRF
^XZ
```

During a multiple transfer, if a file is too big to be stored on the memory card, that file is skipped. All remaining files attempt to be transferred. All files that can be stored within the space limitations are transferred, while other files are ignored.

~WC

The ~WC command is used to generate a printer configuration label. The printer configuration label contains information about the printer setup, such as sensor type, network ID, ZPL mode, firmware version, and descriptive data on the R:, E:, B:, and A: devices.

Print Configuration Label

Format: ~WC

Comments: This command works only when the printer is idle.

PRINTER CONFIGURATION

Zebra Technologies
ZTC Z4MPlus-200 dpi
140XiIIIplus
Zebra

+12	DARKNESS
6 IPS	PRINT SPEED
+000	TEAR OFF
TEAR OFF	PRINT MODE
CONTINUOUS	MEDIA TYPE
WEB	SENSOR TYPE
AUTO SELECT	SENSOR SELECT
THERMAL-TRANS	PRINT METHOD
101 4/8 MM	PRINT WIDTH
2000	LABEL LENGTH
39.0IN 988MM	MAXIMUM LENGTH
BIDIRECTIONAL	PARALLEL COMM.
RS232	SERIAL COMM.
9600	BAUD
8 BITS	DATA BITS
NONE	PARITY
XON/XOFF	HOST HANDSHAKE
NONE	PROTOCOL
000	NETWORK ID
NORMAL MODE	COMMUNICATIONS
<~> 7EH	CONTROL PREFIX
<^> 5EH	FORMAT PREFIX
<,> 2CH	DELIMITER CHAR
ZPL II	ZPL MODE
CALIBRATION	MEDIA POWER UP
CALIBRATION	HEAD CLOSE
DEFAULT	BACKFEED
+000	LABEL TOP
+0020	LEFT POSITION
DISABLED	REPRINT MODE
070	WEB S.
070	MEDIA S.

^WD

The ^WD command is used to print a label listing bar codes, objects stored in DRAM, or fonts.

Print Directory Label

For bar codes, the list shows the name of the bar code. For fonts, the list shows the name of the font, the number to use with ^A command, and size. For objects stored in DRAM, the list shows the name of the object, extension, size, and option flags. All lists are enclosed in a double-line box.

Format: ^WDd:o.x

Parameters	Details
d = source device — optional	**Values:** R:, E:, B:, A: and Z: **Default:** R:
o = object name — optional	**Values:** 1 to 8 alphanumeric characters **Default:** * The use of a ? (question mark) is also allowed.
x = extension — optional .14↑ .TTF and .TTE are only supported in firmware version V60.14.x, V50.14.x, or later.	**Values:** any extension conforming to Zebra conventions .FNT =font .BAR = bar code .ZPL = stored ZPL format .GRF = GRF graphic .CO = memory cache .DAT = font encoding .BAS = ZBI encrypted program .BAE = ZBI encrypted program .STO = data storage .PNG = PNG graphic * = all objects. TTF = TrueType Font .TTE = True Type Extension **Default:** * The use of a ? (question mark) is also allowed.

Example: To print a label listing all objects in DRAM, enter:

```
^XA
^WDR:*.*
^XZ
```

Example: To print a label listing all resident bar codes, enter:

```
^XA
^WDZ:*.BAR
^XZ
```

Example: To print a label listing all resident fonts, enter:

```
^XA
^WDZ:*.FNT
^XZ
```

~WQ

The ~WQ command triggers the printer to print a label with odometer, maintenance or alert, and printhead history information.

Write Query

Format: ~WQquery-type

Parameter	Details
query-type	For detailed examples of these parameters, see ~WQ Examples. **Values:** ES = requests the printer's status. For details see, Table 10 Error Flags (~WQES) on page 362 and Table 11 Warning Flags (~WQES) on page 363. HA = hardware address of the internal wired print server JT = requests a summary of the printer's printhead test results MA = maintenance alert settings MI = maintenance information OD = odometer PH = printhead life history PP = printer's Plug and Play string SN = printer's serial number UI = printer's USB product ID and BCD release version **Default:** must be an accepted value or the command is ignored

Table 10 Error Flags (~WQES)

Error Flags	Flag	Group 2	Group 1 (X = Value can be any hexadecimal number [0-9, A-F])							
		Nibbles 16-9	Nibble8	Nibble7	Nibble6	Nibble5	Nibble4	Nibble3	Nibble2	Nibble1
No Error	0	00000000	0	0	0	0	0	0	0	0
Error Present	1	00000000	X	X	X	X	X	X	X	X
Printhead Thermistor Open	1	00000000	X	X	X	X	X	2	X	X
Invalid Firmware Config.	1	00000000	X	X	X	X	X	1	X	X
Printhead Detection Error	1	00000000	X	X	X	X	X	X	8	X
Bad Printhead Element	1	00000000	X	X	X	X	X	X	4	X
Motor Over Temperature	1	00000000	X	X	X	X	X	X	2	X
Printhead Over Temperature	1	00000000	X	X	X	X	X	X	1	X
Cutter Fault	1	00000000	X	X	X	X	X	X	X	8
Head Open	1	00000000	X	X	X	X	X	X	X	4

Table 10 Error Flags (^WQES) (Continued)

Error Flags	Flag	Group 2	Group 1 (X = Value can be any hexadecimal number [0-9, A-F])							
		Nibbles 16-9	Nibble8	Nibble7	Nibble6	Nibble5	Nibble4	Nibble3	Nibble2	Nibble1
Ribbon Out	1	00000000	X	X	X	X	X	X	X	2
Media Out	1	00000000	X	X	X	X	X	X	X	1
Clear Paper Path Failed[1]	1	00000000	X	X	X	X	8	X	X	X
Paper Feed Error[1]	1	00000000	X	X	X	X	4	X	X	X
Presenter Not Running[1]	1	00000000	X	X	X	X	2	X	X	X
Paper Jam during Retract[1]	1	00000000	X	X	X	X	1	X	X	X
Black Mark not Found[1]	1	00000000	X	X	X	8	X	X	X	X
Black Mark Calabrate Error[1]	1	00000000	X	X	X	4	X	X	X	X
Retract Function timed out[1]	1	00000000	X	X	X	2	X	X	X	X
Paused[1]	1	00000000	X	X	X	1	X	X	X	X

1. This error flag is supported only on KR403 printers.

Table 11 Warning Flags (^WQES)

Error Flags	Flag	Group 2	Group 1 (X = Value can be any hexadecimal number [0-9, A-F])							
		Nibbles 16-9	Nibble8	Nibble7	Nibble6	Nibble5	Nibble4	Nibble3	Nibble2	Nibble1
No Warning	0	00000000	0	0	0	0	0	0	0	0
Warning Present	1	00000000	X	X	X	X	X	X	X	X
Paper-near-end Sensor[1]	1	00000000	X	X	X	X	X	X	X	8
Replace Printhead	1	00000000	X	X	X	X	X	X	X	4
Clean Printhead	1	00000000	X	X	X	X	X	X	X	2
Need to Calibrate Media	1	00000000	X	X	X	X	X	X	X	1
Sensor 1 (Paper before head)[1]	1	00000000	X	X	X	X	X	X	1	X
Sensor 2 (Black mark)[1]	1	00000000	X	X	X	X	X	X	2	X
Sensor 3 (Paper after head)[1]	1	00000000	X	X	X	X	X	X	4	X
Sensor 4 (loop ready)[1]	1	00000000	X	X	X	X	X	X	8	X
Sensor 5 (presenter)[1]	1	00000000	X	X	X	X	X	1	X	X

Table 11 Warning Flags (~WQES) (Continued)

Error Flags	Flag	Group 2	Group 1 (X = Value can be any hexadecimal number [0-9, A-F])							
		Nibbles 16-9	Nibble8	Nibble7	Nibble6	Nibble5	Nibble4	Nibble3	Nibble2	Nibble1
Sensor 6 (retract ready)[1]	1	00000000	X	X	X	X	X	2	X	X
Sensor 7 (in retract)[1]	1	00000000	X	X	X	X	X	4	X	X
Sensor 8 (at bin)[1]	1	00000000	X	X	X	X	X	8	X	X

1. This error flag is supported only on KR403 printers.

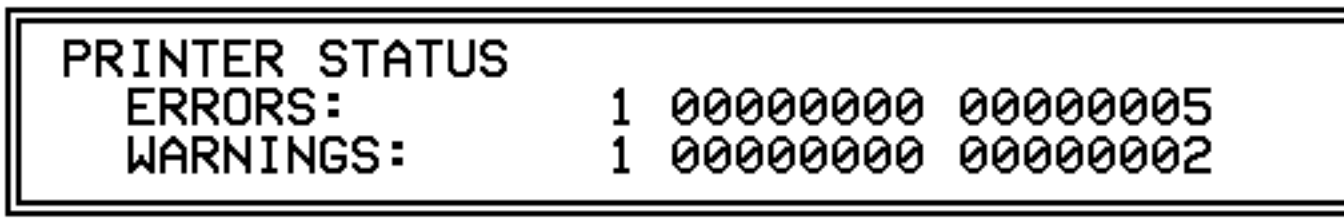
~WQ Examples

This section provides detailed examples of all the available parameters.

Example: This example shows how to request the printer's status.

To request the printer's status, type ~WQES

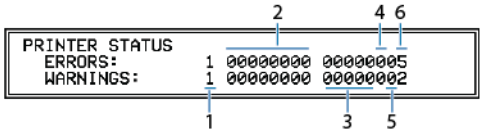
A label similar to this prints out:



In this example, the Printer Status resolves to these conditions:

- The cover/printhead is open (value = 4).
- Media is out or not loaded into the printer (value = 1).
- The printhead needs to be cleaned (value = 2).
- Error nibble 1 is equal to 5 when the error status values are added together (4+1).

This illustration identifies the printer status definitions:

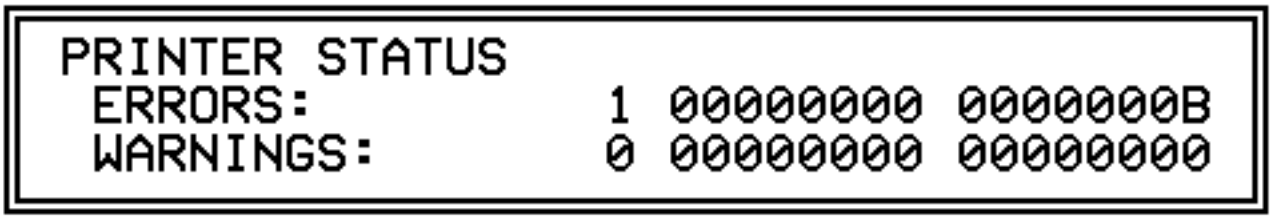


1	Flag
2	Nibble 16-9
3	Nibble 8-4
4	Nibble 3
5	Nibble 2
6	Nibble 1

Example: This example shows how to request the printer's status.

To request the printer's status, type `~WQES`

A label similar to this prints out:



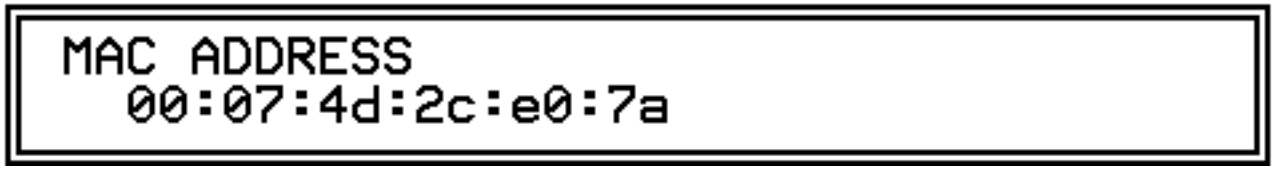
In the example shown above, the Printer Status resolves to the following conditions:

- The cutter has a fault. (value = 8).
- Ribbon is out or not loaded into the printer (value = 2).
- Media is out or not loaded into the printer (value = 1).
- Error byte 1 is equal to B when the error status values are added together (`8 + 2 + 1 = hexadecimal B`).

Example: This is an example of how to print the hardware address of the internal wired print server.

1. To print the hardware address of the internal wired print server, type `~WQHA`

A label similar to this prints out:

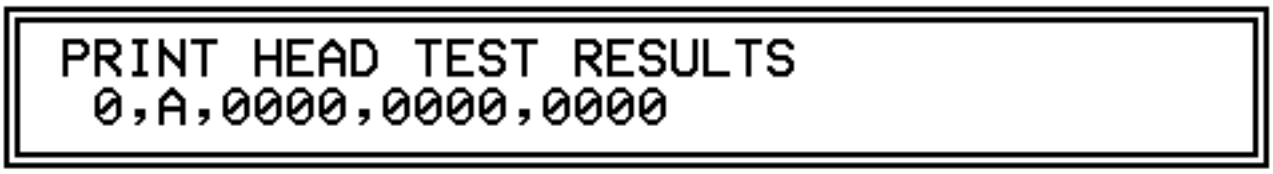


Example: This is an example of how to print a summary of the printer's printhead test results.

The `^JT` command is used to initiate printhead testing, set the testing interval, and set the element range to be tested. For more details see, ^JT.

1. To request a summary of the printer's printhead test, type `~WQJT`

A label similar to this prints out:



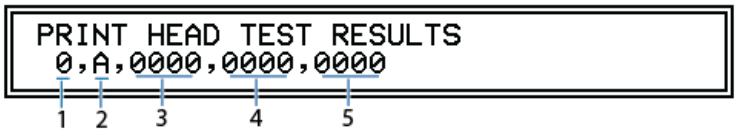
When the printer has printed enough labels to trigger a printhead test, the initial data changes.

1. To request a summary of the printer's printhead test, type `~WQJT`

A label similar to this prints out:

```
PRINT HEAD TEST RESULTS
  0,A,0015,0367,0000
```

This illustration identifies the printhead test field definitions:



1	Element failure
2	Manual (M) or automatic (A) range
3	First test element
4	Last test element
5	Failure count

Example: This is an example of how to print the maintenance alert query for the ~WQ command.

1. To get the current settings, type ~WQMA

A label similar to this prints out:

```
MAINTENANCE ALERT SETTINGS
  HEAD REPLACEMENT INTERVAL:        1 km
  HEAD REPLACEMENT FREQUENCY:       0 M
  HEAD CLEANING INTERVAL:           0 M
  HEAD CLEANING FREQUENCY:          0 M
  PRINT REPLACEMENT ALERT:          NO
  PRINT CLEANING ALERT:             NO
  UNITS:                            C
```

Example: This is an example of how to use the odometer query for the ~WQ command. Note that the units of measure are controlled by the ^MA command. Also, if the "Early Warning Maintenance State" is turned "ON" the printer response would also list LAST CLEANED and CURRENT PRINTHEAD LIFE counters.

1. To get the current settings, type ~WQOD

A label similar to this prints out:

The units of measure are set to inches.

1. To change the units of measure to centimeters, type:

^XA^MA,,,,C

^XZ

The units of measure are set to centimeters.

1. To check the settings, type ~WQOD.

A label similar to this prints out:

Example: This is an example of how to print the maintenance information query for the~WQ command. Note that the message is controlled by the ^MI command.

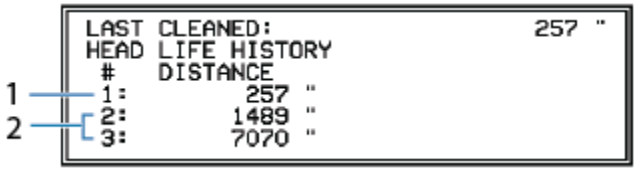
1. To get the current settings, type ~WQMI

A label similar to this prints out:

Example: This is an example of how to print the printhead life query for the ~WQ command. Note that the units of measure are controlled by the ^MA command.

1. To get the current settings, type ~WQPH

A label similar to this prints out:



1	The current life of the print head.
2	Line items 2 through 10 (the example only shows 2 through 3) tracks the measurement for each time the print head is changed.

Example: This is an example of how to print the printer's Plug and Play string.

1. To print the printer's Plug and Play string, type ~WQPP

A label similar to this prints out:

Example: This is an example of how to print the printer's serial number.

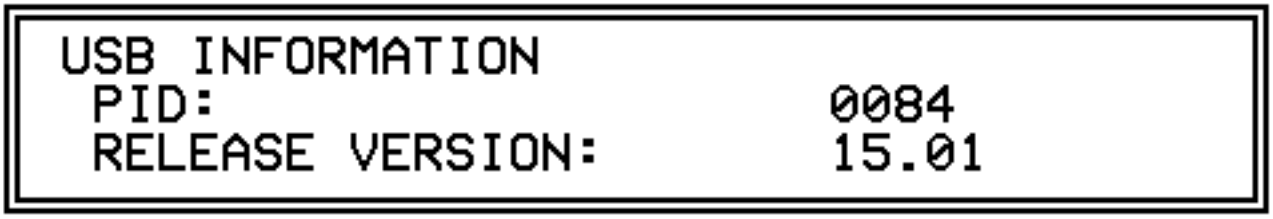
1. To get the printer's serial number, type ~WQSN

A label similar to this prints out:

Example: This is an example of how to print the printer's USB product ID and BCD release version.

1. To print the printer's USB product ID and BCD release version, type ~WQUI

A label similar to this prints out:



^XA

The ^XA command is used at the beginning of ZPL II code. It is the opening bracket and indicates the start of a new label format. This command is substituted with a single ASCII control character STX (control-B, hexadecimal 02).

Start Format

Format: ^XA

Comments: Valid ZPL II format requires that label formats should start with the ^XA command and end with the ^XZ command.

^XB

The ^XB command suppresses forward feed of media to tear-off position depending on the current printer mode. Because no forward feed occurs, a backfeed before printing of the next label is not necessary; this improves throughput. When printing a batch of labels, the last label should not contain this command.

Suppress Backfeed

Format: ^XB

^XB in Tear-off Mode

Normal Operation: backfeed, print, and feed to rest

^XB Operation: print (Rewind Mode)

^XB in Peel-off Mode

Normal Operation: backfeed, print, and feed to rest

^XB Operation: print (Rewind Mode)

NOTE: To prevent jamming in cutter mode, ^XB suppresses backfeed and cutting.

^XF

The ^XF command recalls a stored format to be merged with variable data. There can be multiple ^XF commands in one format, and they can be located anywhere within the code.

Recall Format

When recalling a stored format and merging data using the ^FN (Field Number) function, the calling format must contain the ^FN command to merge the data properly.

While using stored formats reduces transmission time, no formatting time is saved. The ZPL II format being recalled is saved as text strings that need to be formatted at print time.

Format: ^XFd:o.x

Parameters	Details
d = source device of stored image	**Values:** R:, E:, B:, and A: **Default:** search priority (R:, E:, B:, and A:)
o = name of stored image	**Values:** 1 to 8 alphanumeric characters **Default:** if a name is not specified, UNKNOWN is used
x = extension l	**Fixed Value:** .ZPL

For a complete example of the ^DF and ^XF command, seeExercise 6: ^DF and ^XF - Download Format and Recall Format.

^XG

The ^XG command is used to recall one or more graphic images for printing. This command is used in a label format to merge graphics, such as company logos and piece parts, with text data to form a complete label.

Recall Graphic

An image can be recalled and resized as many times as needed in each format. Other images and data might be added to the format.

Format: ^XGd:o.x,mx,my

Parameters	Details
d = source device of stored image	**Values:** R:, E:, B:, and A: **Default:** search priority (R:, E:, B:, and A:)
o = name of stored image	**Values:** 1 to 8 alphanumeric characters **Default:** if a name is not specified, UNKNOWN is used
x = extension l	**Fixed Value:** .GRF
mx = magnification factor on the x-axis	**Values:** 1 to 10 **Default:**1
my = magnification factor on the y-axis	**Values:** 1 to 10 **Default:**1

Example: This is an example of using the ^XG command to recall the image SAMPLE.GRF from DRAM and print it in five different sizes in five different locations on the same label:

```
^XA
^FO100,100^XGR:SAMPLE.GRF,1,1^FS
^FO100,200^XGR:SAMPLE.GRF,2,2^FS
^FO100,300^XGR:SAMPLE.GRF,3,3^FS
^FO100,400^XGR:SAMPLE.GRF,4,4^FS
^FO100,500^XGR:SAMPLE.GRF,5,5^FS
^XZ
```

^XS

The ^XS command controls whether dynamic media calibration is performed to compensate for variations in label length, position, transmissivity, and/or reflectance after a printer is powered-up or the printer has been opened (for example to change or check the media).

Set Dynamic Media Calibration

Format: `^XSlength,threshold`

Parameters	Details
`length` = dynamic length calibration	**Values:** `Y` = enable `N` = disable **Default:** Y
`threshold` = dynamic threshold calibration	**Values:** `Y` = enable `N` = disable **Default:** Y
`gain` = dynamic gain calibration (to be in a future implementation)	**Values:** `Y` = enable `N` = disable **Default:** Y

^XZ

The ^XZ command is the ending (closing) bracket. It indicates the end of a label format. When this command is received, a label prints. This command can also be issued as a single ASCII control character ETX (Control-C, hexadecimal 03).

End Format

Format: ^XZ

Comments: Label formats must start with the ^XA command and end with the ^XZ command to be in valid ZPL II format.

^ZZ

The ^ZZ command places the printer in an idle or shutdown mode.

Printer Sleep

Format: ^ZZt,b

Parameters	Details
t = number of second (idle time) prior to shutdown	**Values:** 0 to 999999 – setting 0 disables automatic shutdown **Default:** last permanently saved value or 0
b = label status at shutdown	**Values:** Y = indicates to shutdown when labels are still queued N = indicates all labels must be printed before shutting down **Default:** N

Comments: The ^ZZ command is only valid on the PA400 and PT400 battery-powered printers.

ZPL Network Commands

This section contains ZPL commands for wired and wireless print servers.

^KC

The ^KC command allows the print server to have its own client identifier (CID).

Set Client Identifier (Option 61)

Format: ^KCa,b,c,d

Parameters	Details
a = enable or disable	**Values:** 0 = disable (default) 1 = enabled, use MAC address 2 = enabled, ASCII value 3 = enabled, HEX value **Default:** 0
b = device	**Values:** 0 = all devices 1 = wireless 2 = external wired 3 = internal wired **Default:** 1
c = prefix (optional)	**Values:** 11 ASCII characters or 22 hexadecimal values. The prefix can be cleared by defaulting the network settings on the printer.
d = identifier	**Values:** 60 ASCII characters or 120 hexadecimal values. Minimum field length is 2 bytes. The suffix can be cleared by defaulting the network settings on the printer.
This applies only to the Xi4, RXi4, ZM400, ZM600, RZ400, or RZ600 printers when it is used with the external ZebraNet 10/100 print server using firmware v1.1.5 or later.	

^NB

Use this command to tell the printer whether to search for a wired print server at bootup. This command is ignored on the Xi4, RXi4, ZM400, ZM600, RZ400, and RZ600 printers.

Search for Wired Print Server during Network Boot

NOTE: Only one print server can be installed in the S4M at one time, so this check does not occur. Table 12 Results of Check for Wired Print Server on page 378 identifies which device becomes the active print server under different conditions.

Table 12 Results of Check for Wired Print Server

If the Check for Wired Print Server is set to:	Installed and Connected to a Live Ethernet Network		Then, the Active Print Server will be:
	Wired	Wireless	
Skip	X	X	Wireless
	X	—	Wired
	—	X	Wireless
Check	X	X	Wired
	X	—	Wired
	—	X	Wireless
A wireless option board must have an active radio that can properly associate to an access point.			

Format: ^NBa

Parameters	Details
a = check for wired print server at boot time	**Values:** C = check S = skip check **Default:** S

^NC

The ^NC command selects the wired or wireless print server as the primary network device.

Select the Primary Network Device

Supported Devices:

- Xi4, RXi4
- ZM400/ZM600, RZ400/RZ600

The Xi4, RXi4, ZM400/ZM600, and RZ400/RZ600 printers support the simultaneous installation of an internal, external, and a wireless print server. Even though all three print servers may be installed, only one is connected to the network and is the active print server. This table outlines priorities and identifies which device becomes the active print server when multiple print servers are installed.

Table 13 Effect of Primary Network Setting on Active Print Server

If the Primary Network is set to:	Installed and Connected to a Live Ethernet Network			Then, the Active Print Server will be:
	Internal	External	Wireless	
Wired	X	X	X	Internal
		X	X	External
			X	Wireless
Wireless	X	X	X	Wireless
	X	X		Internal
		X		External

A wireless option board must have an active radio that can properly associate to an access point.

Format: ^NCa

Parameters	Details
a = primary network device	**Values:** 1 = wired primary 2 = wireless primary **Default:** 1 must be an accepted value or it is ignored

~NC

The ~NC command is used to connect a particular printer to a network by calling up the printer's network ID number.

Network Connect

Format: ~NC###

Parameters	Details
### = network ID number assigned (must be a three-digit entry)	**Values:** 001 to 999 **Default:** 000 (none)

Comments: Use this command at the beginning of any label format to specify which printer on the network is going to be used. Once the printer is established, it continues to be used until it is changed by another ~NC command. This command must be included in the label format to wake up the printer.

The commands ^MW, ~NC, ^NI, ~NR, and ~NT are used only with RS-422/485 printer communications.

^ND

The ^ND command changes the network settings on supported printers.

Change Network Settings

For the external wired print server settings, the ^ND command is the same as the ^NS command. For the wireless print server settings, the ^ND command is the same as the ^WI command.

Format: ^NDa,b,c,d,e,f,g,h,i,j

Parameters	Details
a = the device that is being modified	**Values:** • 1 = external wired • 2 = internal wired • 3 = wireless
b = IP resolution	**Values:** • A = All • B = BOOTP • C = DHCP and BOOTP • D = DHCP • G = Gleaning only (Not recommended when the Wireless Print Server or Wireless Plus Print Server is installed.) • R = RARP • P = Permanent **Default:** A
c = IP address	**Values:** Any properly formatted IP address in the xxx.xxx.xxx.xxx format.
d = subnet mask	**Values:** Any properly formatted subnet mask in the xxx.xxx.xxx.xxx format.
e = default gateway	**Values:** Any properly formatted gateway in the xxx.xxx.xxx.xxx format.
f = WINS server address	**Values:** Any properly formatted WINS server in the xxx.xxx.xxx.xxx format.
g = connection timeout checking	**Values:** Y = yes N = no **Default:** Y
h = timeout value	Time, in seconds, before the connection times out. **Values:** 0 through 9999 **Default:** 300
i = ARP broadcast interval	Time, in minutes, that the broadcast is sent to update the device's ARP cache. **Values:** 0 through 30 **Default:** 0 (no ARP sent)

Parameters	Details
`j` = base raw port number	The port number that the printer should use for its RAW data. **Values:** 1 through 65535 **Default:** `9100`

^NI

The ^NI command is used to assign a network ID number to the printer. This must be done before the printer can be used in a network.

Network ID Number

Format: ^NI###

Parameters	Details
### = network ID number assigned (must be a three-digit entry)	**Values:** 001 to 999 **Default:** 000 (none)

Comments: The last network ID number set is the one recognized by the system.

The commands ~NC, ^NI, ~NR, and ~NT are used only with RS-485 printer communications.

^NN

Use this command to set the Simple Network Management Protocol (SNMP) parameters.

Set SNMP

Format: ^NNa,b,c,d,e,f

Parameters	Details
a = system name	Same as printer name. **Values:** Up to 17 alphanumeric characters
b = system contact	Any contact information as desired (such as a name or phrase) **Values:** Up to 50 alphanumeric characters
c = system location	The printer's model information. **Values:** Up to 50 alphanumeric characters
d = get community name	**Values:** Up to 19 alphanumeric characters **Default:** `public`
e = set community name	**Values:** Up to 19 alphanumeric characters **Default:** `public`
f = trap community name	**Values:** Up to 20 alphanumeric characters **Default:** `public`

^NP

Use this command to specify whether to use the printer's or the print server's LAN/WLAN settings at boot time. The default is to use the printer's settings. This setting only applies to the parallel port connect print servers.

Set Primary/Secondary Device

When the printer is set as the primary device, you can set it up using ZPL commands or the Wireless Setup Wizard utility, and any wired print server inserted into the printer will use those settings. The drawbacks to using the printer as a primary are:

Any wired print server inserted into the printer will lose its original settings if the printer is set to check for the wired print server and the Primary Device is set to PRINTER (see ^NB on page 378).

Format: ^NPa

Parameters	Details
a = device to use as primary	**Values:** P = printerM = MPS/Printserver **Default:** P

~NR

The ~NR command sets all printers in the network to be transparent, regardless of ID or current mode.

Set All Network Printers Transparent

Format: ~NR

Comments: The commands ~NC, ^NI, ~NR, and ~NT are used only with RS-485 printer communications.

^NS

Use this command to change the wired print server network settings.

Change Wired Networking Settings

Format: ^NSa,b,c,d,e,f,g,h,i

Parameters	Details
a = IP resolution	**Values:** A = ALL B = BOOTP C = DHCP AND BOOTP D = DHCP G = GLEANING ONLY R = RARP P = PERMANENT **Default:**A Use of GLEANING ONLY is not recommended when the Wireless Print Server or Wireless Plus Print Server is installed
b = IP address	**Values:** Any properly formatted IP address in the xxx.xxx.xxx.xxx format.
c = subnet mask	**Values:** Any properly formatted subnet mask in the xxx.xxx.xxx.xxx format.
d = default gateway	**Values:** Any properly formatted gateway in the xxx.xxx.xxx.xxx format.
e = WINS server address	**Values:** Any properly formatted WINS server in the xxx.xxx.xxx.xxx format.
f = connection timeout checking	**Values:** Y = Yes N = No **Default:** Y
g = timeout value	Time, in seconds, before the connection times out. **Values:** 0 through 9999 **Default:** 300
h = ARP broadcast interval	Time, in minutes, that the broadcast is sent to update the device's ARP cache. **Values:** 0 through 30 **Default:** 0 (no ARP sent)
i = base raw port number	The port number that the printer should use for its RAW data. **Values:** 1 through 65535 **Default:** 9100

Example:

```
^XA
```

```
^NSa,192.168.0.1,255.255.255.0,192.168.0.2
^XZ
```

Comments:

For the Xi4, RXI4, ZM400/ZM600, and RZ400/RZ600 printers, Zebra recommends that you use the ^ND command instead of the ^NS command.

^NT

Use this command to set the Simple Mail Transfer Protocol (SMTP) parameters. This allows you to set the e-mail settings for alerts.

Set SMTP

Format: ^NTa,b

Parameters	Details
a = SMTP server address	**Values:** Any properly formatted server address in the xxx.xxx.xxx.xxx format
b = print server domain	**Values:** Any properly formatted print server domain name. A domain name is one or more labels separated by a period (“dot”), and a label consists of letters, numbers, and hyphens. An example of a domain name is zebra.com

~NT

The ~NT command sets the currently connected network printer to be transparent.

Set Currently Connected Printer Transparent

Format: ~NT

Comments: With Z Series printers, the ~NT command functions the same as the ~NR command. All Z Series printers on a network receive the transmission.

The commands ~NC, ^NI, ~NR, and ~NT are used only with RS-485 printer communications.

^NW

Use this command to set the timeout value for the printer home page. The printer will prompt for the printer password only the first time that certain screens are accessed until 1) the web authentication timeout value is reached (default value is 5 minutes) or 2) the printer is reset. At that time, the printer will prompt for the password again.

Set Web Authentication Timeout Value

Format: ^NWa

Parameters	Details
a = timeout value	The timeout value in minutes for an IP address to be authenticated to the printer web pages. **Values:** 0 (no secure pages can be accessed without entering the printer password) to 255 minutes **Default:** 5

^WA

Use this command to set the values for the receive and transmit antenna.

Set Antenna Parameters

Format: ^WAa,b

Parameters	Details
a = receive antenna	**Values:** D = diversity L = left R = right **Default:** D
b = transmit antenna	**Values:** D = diversity L = left R = right **Default:** D

^WE

Use this command to command enable Wired Equivalent Privacy (WEP) mode and set WEP values. WEP is a security protocol for wireless local area networks (WLANs).

Set WEP Mode

NOTE:

- The ^WE command is provided only for backward-compatibility with printers using firmware prior to V50.15.x, V53.15.x, or X60.15.x. For these firmware versions and later, use **^WX on page 425** to set the security type and related parameters.
- This command does not apply to printers running Link-OS v6 or later versions.

Be careful to include the exact number of commas required for this command when setting encryption keys (parameters e through h). A missing or extra comma will cause the keys to be stored in the wrong slots and can prevent the printer from joining the wireless network.

Format: ^WEa,b,c,d,e,f,g,h

Parameters	Details
a = encryption mode	**Values:** OFF 40 = 40-bit encryption 128 = 128-bit encryption **Default:** OFF
b = encryption index	Tells the printer which encryption key to use. **Values:** 1 = Key 1 2 = Key 2 3 = Key 3 4 = Key 4 **Default:** 1
c = authentication type	**Values:** O (Open System), S (Shared Key) O = Open System S = Shared Key **Default:** O **NOTE:** If you enable Shared Key authentication with Encryption Mode set to OFF, this value resets to O (Open).
d = encryption key storage	**Values:** H (Hex key storage), S (string key storage) H = Hex key storage S = String key storage **Default:** H

Parameters	Details
e, f, g, h = encryption keys 1 through 4	**Values:** The actual value for the encryption key The encryption mode affects what can be entered for the encryption keys: • For 40-bit, encryption keys can be set to any 5 hex pairs or any 10 alphanumeric characters. • For 128-bit, encryption keys can be set to any 13 hex pairs or any 26 alphanumeric characters. **NOTE:** When using hex storage, do not add a leading 0x on the WEP key.

Example: This example sets encryption to 40-bit, activates encryption key 1, and sets encryption key 1 to the string 12345.

```
^WE40,,,,12345
```

In this example, the Encryption Index, Authentication Type, and Encryption Key Storage parameters are left blank with commas as placeholders for the fields. The printer uses the default values for these parameters.

Example: This example sets encryption to 128-bit, activates encryption key 2, and sets encryption keys 1 and 2 to hex values.

```
^WE128,2,,H,12345678901234567890123456,98765432109876543210987654
```

The value for encryption key 1 is stored and can be activated in the future by the following command:

```
^WE128,1
```

Example: This example sets encryption to 128-bit, activates encryption key 4, and sets encryption key 4 to a hex value.

```
^WE128,4,,H,,,,98765432109876543210987654
```

Values are not required for encryption keys 1 through 3 when setting encryption key 4. In this

example, commas are used as placeholders for the fields for encryption keys 1 through 3.

Any previously stored values for these encryption keys do not change.

NOTE: important: Make sure that you include the exact number of commas required to get to the slot for encryption key 4 (parameter h).

^WL - Set Leap

Use this command to enable Cisco® Lightweight Extensible Authentication Protocol (LEAP) mode and set parameters. LEAP is user authentication method that is available with some wireless radio cards.

Set LEAP Parameters

NOTE: The ^WL command is provided only for backward-compatibility with printers using firmware prior to V50.15.x or X60.15.x. For these firmware versions and later, use **^WX on page 425** to set the security type and related parameters.

Format: ^WLa,b,c

Parameters	Details
a = mode	**Values:** OFF, ON **Default:** OFF
b = user name	**Values:** Any 1 to 32 alphanumeric including special characters **Default:** user
c = password	**Values:** Any 1 to 32 alphanumeric including special characters **Default:** password

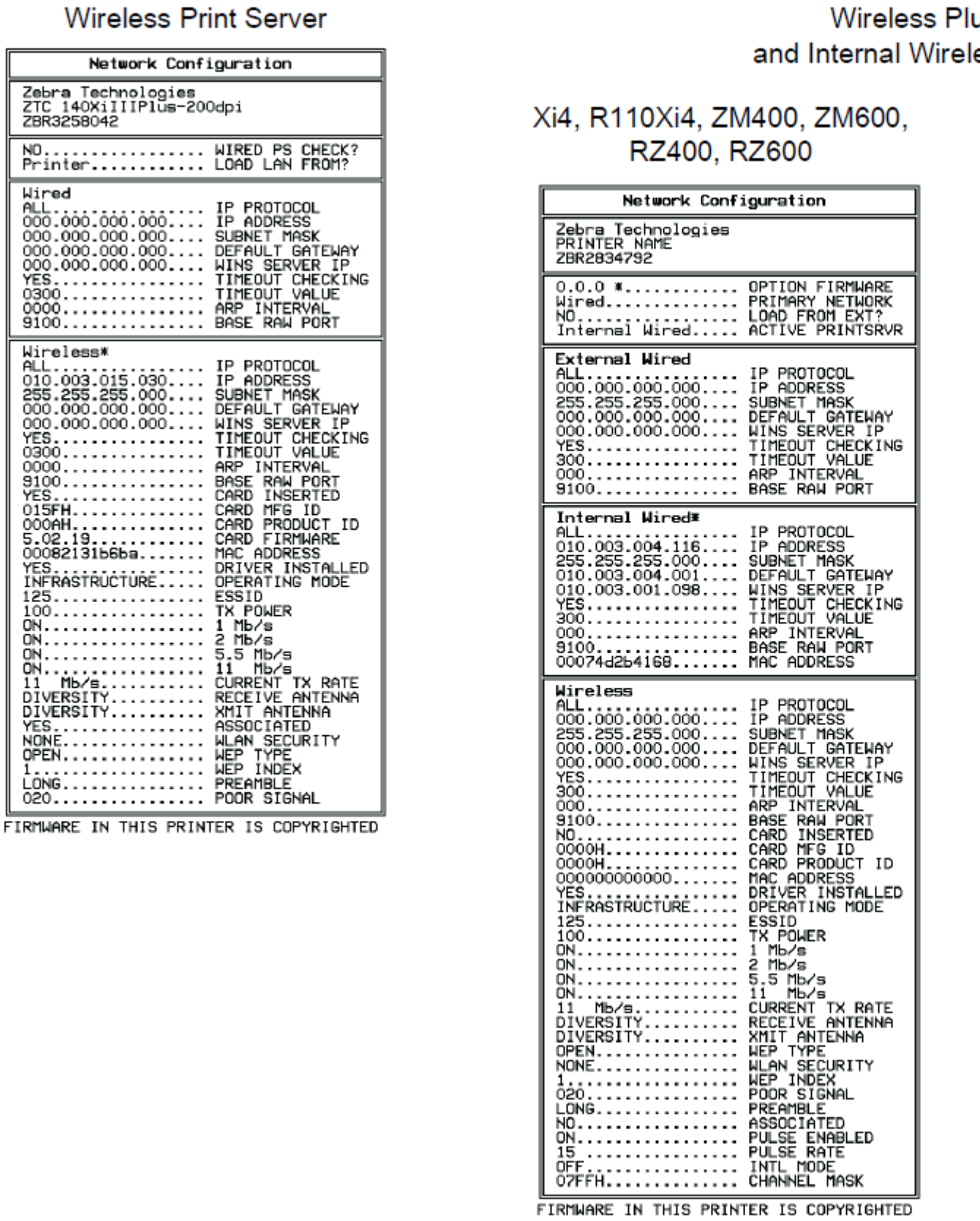
~WL - Print Network

Generates a network configuration label (Network Configuration Label).

Print Network Configuration Label

Format: ~WL

Figure 17 Network Configuration Label



^WP

Use this command to set the four-digit wireless password (not the same as the general printer password). If the wireless password is `0000`, the Wireless and Wireless Plus print servers run in an “unprotected” mode, which means that you do not need to enter the wireless password through the control panel to view or modify wireless settings.

Set Wireless Password

NOTE: This command does not apply to the S4M.

If a wireless password is set, the values for the following parameters will not appear through the control panel until the wireless password is entered:

- MAC Address
- ESSID
- WLAN Security
- WEP Type
- WEP Index
- Reset Network

Format: ^WPa,b

Parameters	Details
a = old wireless password	**Values:** 0000 through 9999 **Default:** 0000
b = new wireless password	**Values:** 0000 through 9999 **Default:** 0000

^WR - Set Transmit

Use this command to change the transmission rate for 802.11b wireless print servers.

Set Transmit Rate

Format: ^WRa,b,c,d,e

Parameters	Details
a = rate 1	Sets the 1 Mb/s transmit rate.Y (On), N (Off)
b = rate 2	Sets the 2 Mb/s transmit rate.Y (On), N (Off)
c = rate 5.5	Sets the 5.5 Mb/s transmit rate.Y (On), N (Off)
d = rate 11	Sets the 11 Mb/s transmit rate.Y (On), N (Off)
e = transmit power	1, 5, 20, 30, 50, 100

NOTE: This command is not valid for Link-OS printers and is only supported in selected other models.

~WR - Reset Wireless

Use this command to reinitialize the wireless radio card and the print server (wired or wireless) when the Wireless or Wireless Plus print server is running. The command also causes any wireless radio card in the printer to reassociate to the wireless network.

Reset Wireless Radio Card and Print Server

Format: ~WR

^WS

Use this command to set the wireless radio card values for ESSID, Operating Mode, and Card Preamble.

Set Wireless Radio Card Values

Format: ^WSe,o,p,h,i,j,k

Parameters	Details
e = ESSID value	**Values:** Any value up to 32 characters, including all ASCII and Extended ASCII characters, including the space character. When this parameter is left blank, the ESSID is not changed. **Default:** 125
o = operating mode	**Values:** I (Infrastructure), A (Adhoc) **Default:** I
p = wireless radio card preamble	**Values:** L = long S = short **Default:** L
h = wireless pulse .15↑ This parameter is supported in firmware version V60.15.x, V50.15.x, R6x.15.x, R53.15.x, ZSPx, or later.	Adds a pulse to the network traffic generated by the printer. This pulse is necessary with some network configurations to keep the printer online.**Values:** 0 = disabled 1 = enabled **Default:** 1
i = wireless pulse interval .15↑ This parameter is supported in firmware version V60.15.x, V50.15.x, R6x.15.x, R53.15.x, ZSPx, or later.	Sets the interval at which the wireless pulse is sent when the wireless pulse feature is enabled. **Values:** 5 to 300 seconds **Default:** 15
j = channel mask .15↑ This parameter is supported in firmware version X60.15.x, V50.15.x, or later.	For commonly used channel masks, see **Table 13 on page 424**. **Values:** 4 Hexadecimal digits preceded by “0x” (0x0000 to 0xFFFF) **Default:** 0x7FF

Parameters	Details
k = international mode This parameter is supported in firmware version X60.15.x, V50.15.x, or later.	In international mode, the printer uses the channel set by the access point. **Values:** 0 (Disabled), 1 (Enabled) **Default:** 0

Table 14 Channel Mask Settings

Region	Channel Mask
United States, Canada, Latin America	0x7FF
Europe, Middle East, Africa, other	0x1FFF
Japan	0x3FFF

^WX

Use this command to configure the wireless security settings for your printer. Values entered for this command must match what is configured on your WLAN and must be supported by the wireless radio card that you are using.

Configure Wireless Securities

The ^WX command replaces individual ZPL commands for different security types.

NOTE:

When using certificate files, your printer supports:

- Using Privacy Enhanced Mail (PEM) formatted certificate files.
- Using the client certificate and private key as two files, each downloaded separately.
- Using exportable PAC files for EAP-FAST.
- The supporting parameters that are required vary based on the security type that you select. See **Supporting Parameters for Different Security Types on page 429** for instructions for each security type.

The values 2, 3 for the security type (a) parameter, b, c, d, e, f, g and h parameters are ignored for printer running Link-OS 6.0 or later versions.

NOTE: important: When using certificate files, the time on the printer must be set correctly for the websocket connection to succeed, as the time is used in the certificate validation.

Format:`^WXa,[zero or more supporting parameters]`

Parameters	Details
a = security type	Enter the two-digit code for the security type that your WLAN uses. For which supporting parameters (b through n) to use with the different security types, see **Supporting Parameters for Differnt Security Types on page 429**. **NOTE:** Configuring the printer for WPA also allows the printer to be used in WPA2 environments. **Values:** 01 to 15 01 - No wireless security is active 02 = WEP 40-bit 03 = WEP 128-bit 04 = EAP-TLS 05 = EAP-TTLS 06 = EAP-FAST 07 = PEAP 08 = LEAP 09 = WPA PSK (R6x15.x, R53.15.x, ZSPx, and later.) 10 = WPA EAP-TLS 11 = WPA EAP-TTLS 12 = WPA EAP-FAST 13 = WPA PEAP 14 = WPA LEAP 15 = Kerberos **Default:** 01
b = WEP encryption index*	Specifies which encryption key to use for WEP encryption. A value must be specified if using WEP 40-bit or WEP 128-bit. **Values:** 1, 2, 3, 4 **Default:** 1
c= WEP authentication type*	Enables the WEP key authentication type. A value must be specified if using WEP 40-bit or WEP 128-bit. **Values:** 0 or S 0 = open system S = shared key **Default:** 0
d = WEP key type*	Specifies the format of the WEP key. A value must be specified if using WEP 40-bit or WEP 128-bit. **Values:** H or S H = hex key storage S = string key storage **Default:** S

Parameters	Details
e,f,g,h = WEP encryption keys 1 through 4*	Specifies the actual values of any WEP encryption keys to be used. A value must be specified for at least one WEP encryption key if you specify 40-bit or 128-bit WEP encryption for the security type. **NOTE: important:** Be careful to include the exact number of commas required for this command when setting encryption keys (parameters e through h). A missing or extra comma will cause the keys to be stored in the wrong slots and can prevent the printer from joining the wireless network. The encryption mode affects what can be entered for the encryption keys: • For 40-bit, encryption keys can be set to any 5 hex pairs or any 10 alphanumeric characters. • For 128-bit, encryption keys can be set to any 13 hex pairs or any 26 alphanumeric characters. **NOTE:** When using hex storage, do not add a leading 0x on the WEP key. **Values:** The actual value for the encryption key **Default:** None
• i = user ID*	Specifies a user ID for security types that require one. A value must be specified if using the following security types: • EAP-TTLS • LEAP • WPA LEAP • PEAP • WPA PEAP • WPA EAP-TTLS • Kerberos **Values:** The actual value for the user ID. **Default:** user

Parameters	Details
• j = password*	Specifies a password for security types that require one. A value must be specified if using the following security types: • EAP-TTLS • LEAP • WPA LEAP • PEAP • WPA PEAP • WPA EAP-TTLS • Kerberos **Values:** The actual value for the password. **Default:** password
• k = optional private key password*	Specifies an optional private key password for security types that require one. A value must be specified if using the following security types: • EAP-TLS • EAP-FAST • WPA EAP-TLS • WPA EAP-FAST **Values:** The actual value for the optional private key. **Default:** None
• l = realm*	Specifies the realm for security types that require it. A value must be specified if using Kerberos. **Values:** The actual value for the realm. **Default:** kerberos
• m = Key Distribution Center (KDC)*	Specifies the KDC for security types that require it. A value must be specified if using Kerberos. **Values:** The actual value for the KDC. **Default:** krbtgt"
• n = Pre-Shared Key (PSK) value*	Enter the PSK value. This value is calculated and must be the same for each device on the WLAN. Use ZebraNet Bridge to generate the PSK value. A value must be specified if using WPA PSK. **NOTE: important:** Do not enter a pass phrase for this field in this command. To use a pass phrase, use the ZebraNet Bridge Enterprise Wireless Setup Wizard. **Values:** a minimum of 64 hexadecimal digits**Default:** None
* Not used for all security types	

Supporting Parameters for Different Security Types

The supporting parameters required for this command vary based on the security type that you select. You should not use all of the supporting parameters each time that you use this command, nor will you use

extra commas to separate unused fields. Follow the example and format for your specific security type in this section, substituting your own wireless network data.

Security Type 01: No Wireless Security Active

Format: ^WX01

Example: This example turns off all wireless securities controlled under this command, but it does not reset the printer's wireless settings to their defaults.

```
^XA
^WX01
^JUS^XZ
```

Security Type 02: WEP 40-Bit

Format: ^WX02,b,c,d,e,f,g,h

Example: This example configures the printer for WEP 40-bit encryption using index key 1, open authentication, and a hexadecimal WEP key with a value of "A1B2C3D4F5."

```
^XA
^WX02,1,O,H,A1B2C3D4F5,,,
^JUS
^XZ
```

NOTE: This is no longer valid for Link OS 6 printers.

Security Type 03: WEP 128-Bit

Format: ^WX03,b,c,d,e,f,g,h

Example: This example configures the printer for WEP 128-bit encryption using index key 2, open authentication, and four hexadecimal WEP keys.

```
^XA
^WX03,2,O,H,001122334455667788,112233445566778899,223344556677889900,334455667788990
^XZ
```

NOTE: This command is not valid for printers running Link OS 6 or later versions.

Security Type 04: EAP-TLS

Format: ^WX04,k

Example: This example configures the printer for EAP-TLS authentication with an optional private key password with a value of "`private`."

```
^XA
^WX04,private
^JUS
^XZ
```

Security Type 05: EAP-TTLS

Format: ^WX05,i,j

Example: This example configures the printer for EAP-TTLS authentication, including a user ID of "`user`" and a password of "`password`."

```
^XA
^WX05,user,password
^JUS
^XZ
```

Security Type 06: EAP-FAST

Format: ^WX06,i,j,k

Example: This example configures the printer for EAP-FAST authentication, including a user ID of "`user`," a password of "`password`," and an optional private key of "`private`."

```
^XA
^WX06,user,password,private
^JUS
^XZ
```

Security Type 07: PEAP

Format: ^WX07,i,j

Example: This example configures the printer for PEAP authentication, including a user ID with a value of "`user`" and a password with a value of "`password`."

```
^XA
^WX07,user,password
^JUS
^XZ
```

Security Type 08: LEAP

Format: ^WX08,i,j

Example: This example configures the printer for LEAP authentication, including a user ID with a value of "`user`" and a password with a value of "`password`."

```
^XA
^WX08,user,password
^JUS
^XZ
```

Security Type 09: WPA PSK

NOTE: Configuring the printer for WPA also allows the printer to be used in WPA2 environments (R6x15.x, R53.15.x, ZSPx, and later.)

Format: ^WX09,n

Example: This example configures the printer for WPA PSK authentication with a PSK value of all zeroes (64 hexadecimal digits).

```
^XA
^WX09,00000000...^JUS
^XZ
```

Security Type 10: WPA EAP-TLS

NOTE: Configuring the printer for WPA also allows the printer to be used in WPA2 environments.

Format: ^WX10,k

Example: This example configures the printer for WPA EAP-TLS authentication with an optional private key password with a value of "`private`."

```
^XA
^WX10,private
^JUS
^XZ
```

Security Type 11: WPA EAP-TTLS

NOTE: Configuring the printer for WPA also allows the printer to be used in WPA2 environments.

Format: ^WX11,i,j

Example: This example configures the printer for WPA EAP-TTLS authentication, including a user ID with a value of "`user`" and a password with a value of "`password`."

```
^XA
^WX11,user,password
^JUS
^XZ
```

Security Type 12: WPA EAP-FAST

NOTE: Configuring the printer for WPA also allows the printer to be used in WPA2 environments.

Format: ^WX12,i,j,k

Example: This example configures the printer for WPA EAP-FAST authentication, including a user ID of "`user`," a password of "`password`," and an optional private key of "`private`."

```
^XA
^WX12,user,password,private
^JUS
^XZ
```

Security Type 13: WPA PEAP

NOTE: Configuring the printer for WPA also allows the printer to be used in WPA2 environments.

Format: ^WX13,i,j

Example: This example configures the printer for WPA PEAP authentication, including a user ID with a value of "`user`" and a password with a value of "`password`."

```
^XA
^WX13,user,password
^JUS
^XZ
```

Security Type 14: WPA LEAP

NOTE: Configuring the printer for WPA also allows the printer to be used in WPA2 environments.

Format: ^WX14,i,j

Example: This example configures the printer for WPA LEAP authentication, including a user ID with a value of "`user`" and a password with a value of "`password`."

```
^XA
^WX14,user,password
^JUS
^XZ
```

Security Type 15: Kerberos

Format: ^WX15,i,j,l,m

Example: This example configures the printer for Kerberos encryption, including a Kerberos user ID with a value of "`user`," a Kerberos password with a value of "`password`," a realm of "`zebra`," and a KDC of "`krbtgt`."

```
^XA
^WX15,user,password,zebra,krbtgt
^JUS
^XZ
```

ZPL RFID Commands

This section contains the ZPL II commands for RFID-specific applications.

For additional information, refer to the RFID Programming Guide for your printer. A copy is available at www.zebra.com/manuals.

^HL or ~HL

The printer can log RFID data and store it in the printer's RAM. These commands request that the RFID data log be returned to the host computer. The ~HL command is processed immediately, while the ^HL command is processed after all of the previous formats (^XA ... ^XZ) have been processed.

Return RFID Data Log to Host

The firmware version determines the way that these commands function:

- In firmware X.20.16Z and later, for security, logging is disabled by default. The ^HL command clears the current data log and restarts data recording. The ~HL command does not automatically clear the data log. The RFID host logs can be enabled or disabled by the `rfid.log.enabled` SGD command (see rfid.log.enabled on page 1539).
- In firmware X.20.15Z and earlier, logging is enabled by default. Both commands clear the current data log and restart data recording.

Format: ^HL or ~HL

In the log, RFID data displays in this format:

```
[date&time][RFID operation],[program position],[antenna element],
[read or write power], [RFID status],[data]
```

where

- `[date&time]*`

  a time stamp for the log entry * With some older versions of firmware, this parameter does not display.

- `[RFID operation]`

  B = a ^RLB command was issued (see ^RLB – Permanently Lock Specified Memory Sections on page 429)

  E = log file reset

  L = lock

  M = a ^RLM command was issued (see ^RLM – Lock/Unlock the Specified Memory Bank on page 429)

  R = read

  S = RFID settings

  W = write

- `[program position],[antenna element],[read or write power]*`

  Additional information about the program position, the antenna, and the read or write power follows the RFID operation.

  Such as:

```
R,F1,D3,27,00000000,DATA
```

where F1 = the program position, D3 = the antenna, and 27 is the write power. * With some older versions of firmware, these parameters do not display.

- [RFID status]

  #### or ######## = an RFID error code (See the RFID Programming Guide for your printer for more information on error codes. You can download a copy from www.zebra.com/manuals.)

  RPWR = read power

  WPWR = write power

  ANT = antenna

  PPOS = program position

  FFFFFFFF (or limited to length FFFF for some printers) = indicates that the log file was reset

- [data]

  the data read or written

Comments:

- Data is shown in the format specified by the ^RFW command (ASCII, Hex, or EPC).
- If the RFID data log exceeds the maximum size, the following occurs:
  - In firmware X.20.16Z and later, when the data log reaches 1500K, one or more older entries are deleted to make room for the newest entry.
  - In firmware X.20.15Z and earlier, when the data log reaches 64K, the RFID data log is cleared automatically, and data recording restarts. When this happens, the following appears in the log:

    ```
    E,FFFFFFFF,Logfile automatically reset
    ```
- In firmware X.20.15Z and earlier, If the printer loses power, the log is lost. If the log results are important to you, retrieve the information frequently.

^HR

Use this command to initiate tag calibration for RFID media. During the tag calibration process (which can take up to 5 minutes on some printers, depending on the type of RFID inlay and the label size), the printer moves the media, reads the tag's TID to determine chip type, calibrates the RFID tag position, and determines the optimal settings for the RFID media being used. Depending on the printer, these settings include the programming position, the antenna element to use, and the read/write power level to use.

Calibrate RFID Tag Position

Results of the ^HR tag calibration are returned to the host computer. The `"run"` option in the `rfid.tag.calibrate` SGD command performs the same calibration but does not create a results table. To restore the printer's default programming position at any time, use the `"restore"` option in the `rfid.tag.calibrate` SGD command (see rfid.tag.calibrate on page 1524).

Before running this command, load the printer with RFID media, calibrate your printer, close the printhead, and feed at least one label to make sure that tag calibration will begin from the correct position. For more information on media calibration, refer to the User Guide for your printer.

IMPORTANT: Consider the following before using this command:

- This command is not supported by all printers or firmware.
- For the R110Xi4 and all Link-OS RFID printers, leave all transponders before and after the tag that is being calibrated. This allows the printer to determine RFID settings that do not encode the adjacent tag. Allow a portion of media to extend out the front of the printer to allow for backfeed during the tag calibration procedure.
- With some printers, you should not perform transponder calibration for RFID media that meets the transponder placement specifications for your environment. With some printers, you should not perform transponder calibration for RFID media that meets the transponder placement specifications for your environment; because doing so will slow the printer's throughput unnecessarily. For more information about tag calibration, refer to the RFID Programming Guide for your printer. You can download a copy from www.zebra.com/manuals.

Format: `^HRa,b,c,d,e`

Parameters	Details
a = start string	This parameter specifies the user text to appear before the results table. **Values:** any string less than 65 characters **Default:** start
b = end string	This parameter specifies the user text to appear after the results table. **Values:** any string less than 65 characters **Default:** end

Parameters	Details
c = start position	This parameter specifies the start position of the calibration range. All numeric values are in millimeters. Forward or backward designations assume that the label's initial position is with the leading edge at the print line. **Values:** • **Forward:** F0 to Fxxx (where xxx is the label length in millimeters or 999, whichever is less) The printer feeds the label forward for the specified distance and then begins tag calibration. • **Backward:** B0 to B30 The printer backfeeds the label for the specified distance and then begins tag calibration. To account for the backfeed, allow empty media liner to extend out of the front of the printer when using a backward programming position. For printers that do not use backfeed during RFID calibration, the media is moved forward until it is in the same relative position for the following label. **Default:** • For ZT400 Series and ZT600 Series printers with RFID option: B30 • For R110Xi4, ZD500R, ZQ511/ZQ521, and ZQ630 printers with RFID option: B20 • For all other supported printers: F0—The printer moves the media to the start position relative to the leading edge of the label and then performs the RFID tag calibration.
d = end position	This parameter specifies the end position of the calibration range (last program position to check). All numeric values are in millimeters. Forward or backward designations assume that the label's initial position is with the leading edge at the print line. **Values:** • **Forward:** F0 to Fxxx (where xxx is the label length in millimeters or 999, whichever is less) The printer performs tag calibration until it reaches the specified end position and then ends the process. • **Backward:** B0 to B30 The printer performs tag calibration until it reaches the specified end position and then ends the process. Valid only with a backward start position that is greater than the end position. • **Automatic:** A The printer automatically ends the tag calibration process after successfully reading and encoding a consecutive range of 5 mm on the label. The printer also ensures that no other tags can be programmed at the programming position with the calibration-determined power levels. **Default:** For R110Xi4 and all Link-OS RFID printers: A For all other supported printers: Label length as shown on the printer configuration label

Parameters	Details
e = antenna and read/write power level detection	This parameter specifies whether to select the antenna and read/write power levels automatically or manually. **NOTE:** This parameter is not valid on all RFID printers. The ZD500R, ZQ511/ZQ521, and ZQ630 printers have only one antenna, so this parameter applies only to the read/write power level settings. **Values:** • A = Automatic. The printer automatically scans through the antennas and read/write power during calibration. • M = Manual. The printer uses the current antenna and read/write power level settings. **Default:** A

Example: When the printer is using Absolute mode and the following command is sent to the printer:

```
^XA^HR^XZ
```

the printer starts the transponder calibration and returns a results table such as the following:

```
start
position=195
215, ,
214, ,
213, ,
212, ,
211, ,
210, ,W
209,R,
208, ,
207, ,
206, ,W
205,R,
204, ,
203, ,
202, ,W
201,R,W
200,R,W
199,R,W
198,R,W
197,R,W
196,R,W
195,R,W <---****
194,R,W
193,R,W
192,R,W
191,R,W
190,R,W
189,R,
188, ,
```

```
187, ,
186, ,
185, ,
.
.
.
end
```

Each line in the results table appears as:

```
    Row, Read Result, Write Result
```

where

```
    Row = the dot row where calibration occurred
    Read Result = results of calibration (R = read, “ ” = unable to read)
    Write Result = results of calibration (W = write, “ ” = unable to write)
```

The optimal programming position is 195. This is identified at the top of the table (`position=195`) and with an the arrow (`<---****`) in the table.

Example: When the printer is using Relative mode and the following command is sent to the printer:

```
^HRstart,end,B20,F42,M
```

the printer starts the tag calibration and returns a results table such as the following:

```
start
position=F0 MM
leading edge
B20, ,
B19, ,
B18, ,
B17, ,
…
B8, ,
B7, ,
B6, ,
B5, ,
B4,R,W
B3,R,W
B2,R,W
B1,R,W
F0,R,W<---**** F0 MM
F1,R,W
F2,R,W
F3,R,W
F4, ,
F5, ,
F6, ,
F7, ,
F8, ,
```

```
F9, ,
F10, ,
…
F38, ,
F39, ,
F40, ,
F41, ,
F42, ,
trailing edge
end
```

Each line in the results table appears as:

```
    Row, Read Result, Write Result
```

where

```
    Row = the position from the leading edge of the label where calibration
 occurred
    Read Result = results of calibration (R = read, “ ” = unable to read)
    Write Result = results of calibration (W = write, “ ” = unable to write)
```

The optimal programming position is F0 (program with the leading edge of the label at the print line). This is identified at the top of the table (`position=F0 MM`) and with an the arrow (`<---****`) in the table.

Example: When the ^HR command is sent to the printer, the printer performs tag calibration and returns a results table such as the following:

```
start
position=B14 MM,A1,18,25
tid information=E200.3414:Alien
leading edge
    Tag 1   ,Tag 2   ,Tag 3   ,Tag 4   ,Tag 5   ,Tag 1   ,Tag 2   ,Tag 3
  ,Tag 4   ,Tag 5   ,
EPC,7109    ,BA29    ,6FD0    ,58AE    ,9CDE    ,7109    ,BA29    ,6FD0
  ,58AE    ,9CDE    ,
B30,A1,12,18,A1,29,  ,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,17,24,B1,  ,
  ,B1,  ,  ,B1,  ,  ,
B29,A1,13,18,A1,25,  ,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,14,19,B1,  ,
  ,B1,  ,  ,B1,  ,  ,
B28,A1,15,20,A1,23,29,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,09,15,B1,  ,
  ,B1,  ,  ,B1,  ,  ,
B27,A1,17,22,A1,23,29,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,08,14,B1,  ,
  ,B1,  ,  ,B1,  ,  ,
B26,A1,19,25,A1,  ,  ,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,09,15,B1,28,
  ,B1,  ,  ,B1,  ,  ,
B25,A1,22,28,A1,22,27,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,11,18,B1,26,
  ,B1,  ,  ,B1,  ,  ,
B24,A1,26,  ,A1,13,19,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,15,21,B1,27,
  ,B1,  ,  ,B1,  ,  ,
B23,A1,  ,  ,A1,08,14,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,18,24,B1,  ,
  ,B1,  ,  ,B1,  ,  ,
```

```
B22,A1,  ,  ,A1,05,11,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,
  ,B1,21,28,B1,19,24,B1,  ,  ,B1,  ,  ,
B21,A1,  ,  ,A1,05,11,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,25,
  ,B1,11,17,B1,  ,  ,B1,  ,  ,
B20,A1,  ,  ,A1,06,12,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,30,
  ,B1,07,13,B1,  ,  ,B1,  ,  ,
B19,A1,  ,  ,A1,08,15,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,05,11,B1,  ,  ,B1,  ,  ,
B18,A1,  ,  ,A1,15,22,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,05,10,B1,  ,  ,B1,  ,  ,
B17,A1,  ,  ,A1,22,28,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,05,11,B1,  ,  ,B1,  ,  ,
B16,A1,  ,  ,A1,16,23,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,07,13,B1,  ,  ,B1,  ,  ,
B15,A1,  ,  ,A1,13,19,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,13,20,B1,  ,  ,B1,  ,  ,
B14,A1,  ,  ,A1,12,19,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,18,23,B1,  ,  ,B1,  ,  ,<---****A1
B13,A1,  ,  ,A1,14,20,A1,24,30,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,10,16,B1,  ,  ,B1,  ,  ,
B12,A1,  ,  ,A1,15,22,A1,22,29,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,08,14,B1,  ,  ,B1,  ,  ,
B11,A1,  ,  ,A1,18,25,A1,26,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,08,14,B1,  ,  ,B1,  ,  ,
B10,A1,  ,  ,A1,21,27,A1,26,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,11,17,B1,26,  ,B1,  ,  ,
B09,A1,  ,  ,A1,24,  ,A1,15,21,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,14,20,B1,25,  ,B1,  ,  ,
B08,A1,  ,  ,A1,28,  ,A1,09,15,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,17,23,B1,  ,  ,B1,  ,  ,
B07,A1,  ,  ,A1,  ,  ,A1,06,11,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,20,26,B1,27,30,B1,  ,  ,
B06,A1,  ,  ,A1,  ,  ,A1,05,11,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,24,30,B1,16,19,B1,  ,  ,
B05,A1,  ,  ,A1,  ,  ,A1,05,11,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,  ,B1,28,
  ,B1,10,14,B1,  ,  ,
B04,A1,  ,  ,A1,  ,  ,A1,08,14,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,07,11,B1,  ,  ,
B03,A1,  ,  ,A1,  ,  ,A1,12,18,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,06,11,B1,  ,  ,
B02,A1,  ,  ,A1,  ,  ,A1,20,26,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,06,10,B1,  ,  ,
B01,A1,  ,  ,A1,  ,  ,A1,18,24,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,09,13,B1,  ,  ,
F00,A1,  ,  ,A1,  ,  ,A1,14,21,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,12,17,B1,  ,  ,
F01,A1,  ,  ,A1,  ,  ,A1,13,19,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,20,25,B1,  ,  ,
F02,A1,  ,  ,A1,  ,  ,A1,13,19,A1,27,  ,A1,  ,  ,B1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,16,20,B1,  ,  ,
F03,A1,  ,  ,A1,  ,  ,A1,14,21,A1,26,29,A1,  ,  ,B1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,11,16,B1,  ,  ,
F04,A1,  ,  ,A1,  ,  ,A1,17,24,A1,27,  ,A1,  ,  ,B1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,11,15,B1,  ,  ,
```

```
F05,A1,  ,  ,A1,  ,  ,A1,19,26,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,12,16,B1,25,  ,
F06,A1,  ,  ,A1,  ,  ,A1,22,29,A1,23,26,A1,  ,  ,B1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,15,18,B1,23,28,
F07,A1,  ,  ,A1,  ,  ,A1,26,  ,A1,15,19,A1,  ,  ,B1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,17,22,B1,23,29,
F08,A1,  ,  ,A1,  ,  ,A1,  ,  ,A1,10,14,A1,  ,  ,B1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,20,25,B1,  ,  ,
F09,A1,  ,  ,A1,  ,  ,A1,  ,  ,A1,08,12,A1,  ,  ,B1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,24,28,B1,21,26,
F10,A1,  ,  ,A1,  ,  ,A1,  ,  ,A1,08,11,A1,  ,  ,B1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,27,  ,B1,13,18,
trailing edge
end
```

In the results table, the tags visible to the antenna elements are numbered, and the EPC number that is unique to each tag is displayed.

Each line in the results table gives a row number followed by readings associated with RFID tags that are visible at that row. Multiple values on a line indicate that multiple tags were visible. The order of the RFID tags is arbitrary.

```
    [Row],[Antenna Element],[Min Read Power],[Min Write Power], [Antenna
 Element],[Min Read Power],[Min Write Power] ...
```

where

- Row = the position from the leading edge of the label where calibration occurred
- Antenna Element = the antenna used
- Minimum Read Power = calibration results (`0` – `30`) for a tag visible from that row
- Minimum Write Power = calibration results (`0` – `30`) for the same tag

The read and write power values are left empty (such as `A1, , ,`) when no tag is found.

In the sample results table for this example, at position B25 (25 mm behind the print line), two RFID tags are visible to the printer at antenna A1. Tag 1 (EPC 7109) can be read at power level 22 and written to at power level 28. Tag 2 (EPC BA29) can be read at power level 22 and written to at power level 27. At that position, Tags 2 and 3 are visible to antenna B1 while Tag 1 is not.

```
    Tag 1   ,Tag 2   ,Tag 3   ,Tag 4   ,Tag 5   ,Tag 1   ,Tag 2   ,Tag 3
  ,Tag 4   ,Tag 5   ,
EPC,7109    ,BA29    ,6FD0    ,58AE    ,9CDE    ,7109    ,BA29    ,6FD0
  ,58AE    ,9CDE    ,
...
B25,A1,22,28,A1,22,27,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,11,18,B1,26,
  ,B1,  ,  ,B1,  ,  ,
B24,A1,26,  ,A1,13,19,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,15,21,B1,27,
  ,B1,  ,  ,B1,  ,  ,
B23,A1,  ,  ,A1,08,14,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,18,24,B1,  ,
  ,B1,  ,  ,B1,  ,  ,
B22,A1,  ,  ,A1,05,11,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,
  ,B1,21,28,B1,19,24,B1,  ,  ,B1,  ,  ,
```

```
B21,A1,  ,  ,A1,05,11,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,25,
  ,B1,11,17,B1,  ,  ,B1,  ,  ,
...
```

At position B23, only Tag 2 is visible to antenna A1. Tag 1 is no longer visible.

```
    Tag 1   ,Tag 2   ,Tag 3   ,Tag 4   ,Tag 5   ,Tag 1   ,Tag 2   ,Tag 3
  ,Tag 4   ,Tag 5   ,
EPC,7109    ,BA29    ,6FD0    ,58AE    ,9CDE    ,7109    ,BA29    ,6FD0
  ,58AE    ,9CDE    ,
...
B25,A1,22,28,A1,22,27,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,11,18,B1,26,
  ,B1,  ,  ,B1,  ,  ,
B24,A1,26,  ,A1,13,19,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,15,21,B1,27,
  ,B1,  ,  ,B1,  ,  ,
B23,A1,  ,  ,A1,08,14,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,18,24,B1,  ,
  ,B1,  ,  ,B1,  ,  ,
B22,A1,  ,  ,A1,05,11,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,
  ,B1,21,28,B1,19,24,B1,  ,  ,B1,  ,  ,
B21,A1,  ,  ,A1,05,11,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,25,
  ,B1,11,17,B1,  ,  ,B1,  ,  ,
...
```

At position B13, Tag 3 (EPC 6FD0) becomes visible to antenna A1 and can be read with at power level 24 and written to at power level 30.

```
    Tag 1   ,Tag 2   ,Tag 3   ,Tag 4   ,Tag 5   ,Tag 1   ,Tag 2   ,Tag 3
  ,Tag 4   ,Tag 5   ,
EPC,7109    ,BA29    ,6FD0    ,58AE    ,9CDE    ,7109    ,BA29    ,6FD0
  ,58AE    ,9CDE    ,
...
B16,A1,  ,  ,A1,16,23,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,07,13,B1,  ,  ,B1,  ,  ,
B15,A1,  ,  ,A1,13,19,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,13,20,B1,  ,  ,B1,  ,  ,
B14,A1,  ,  ,A1,12,19,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,18,23,B1,  ,  ,B1,  ,  ,<---****A1
B13,A1,  ,  ,A1,14,20,A1,24,30,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,10,16,B1,  ,  ,B1,  ,  ,
...
```

The arrow (<---****) in the table indicates that a valid program position and power levels were found during calibration. The program position is identified at the top of the table as `position=B14 MM` (backfeed 14 millimeters). The optimal antenna element at that position is A1. The optimal read power is 18, and the optimal write power is 25.

```
start
position=B14 MM,A1,18,25
tid information=E200.3414:Alien
leading edge
```

```
...
B14,A1,  ,  ,A1,12,19,A1,  ,  ,A1,  ,  ,A1,  ,  ,B1,  ,  ,B1,  ,
  ,B1,18,23,B1,  ,  ,B1,  ,  ,<---****A1
...
```

^RB

Use this command to define the structure of EPC data, which can be read from or written to an RFID tag. For more information about EPC specifications, refer to the EPC Global web site. All parameters in this command are persistent and will be used in subsequent formats if not provided. The values are initially set to the default values.

Define EPC Data Structure

RFID tags can have different partitions defined. This command specifies the number of partitions and how many bits are in each partition.

Format: `^RBn,p0,p1,p2, ..., p15`

Parameters	Details
n = total bit size of the partitions	Specify the number of bits to include in the partitions. **Values:** 1 to n, where n is the bit size of the tag. **Default:** 96
`p0 ... p15` = partition sizes	Specify the number of bits to include in the individual partitions. The partition sizes must add up to the bit size specified for the previous parameter. The largest individual partition size is 64 bits. **Values:**1 to 64 **Default:**1

Example: The following command specifies that there are 96 bits used with three fields. Fields 1, 2, and 3 contain 10, 26, and 60 bits, respectively.

```
^RB96,10,26,60
```

The ZPL code to encode a tag with this format would look like this:

```
^RFW,E^FD1000.67108000.112292150460684697 6^FS
```

When the tag is being encoded, the tag stores the data in the following way:

- Field 1 contains `1000`. This value is stored in the first 10 bits
- Field 2 contains `67108000`. This value is stored in the next 26 bits.
- Field 3 contains `1122921504606846976`. This value is stored in the remaining 60 bits.

Example: The following command specifies that there are 64 bits used with eight 8-bit fields.

```
^RB64,8,8,8,8,8,8,8,8^FS
```

The ZPL code to encode a tag with this format would look like this:

```
^RFW,E^FD1.123.160.200.249.6.1.0^FS
```

When writing to the tag, each set of data is written in its respective 8-bit field.

Example: This example uses the SGTIN-96 standard, which defines 96-bit structure in the following way:

	Header	Filter Value	Partition	Company Prefix Index	Item Reference	Serial Number
SGTIN-96	8 bits	3 bits	3 bits	20–40 bits	24 bits	38 bits
	10 (binary value)	8 (decimal capacity)	8 (decimal capacity)	16,383 (decimal capacity)	9 to 1,048,575 (decimal capacity*)	33,554,431 (decimal capacity)
* Capacity of Item Reference field varies with the length of the company prefix.						

The ZPL code to encode a tag with this format would look like this:

```
^XA
^RB96,8,3,3,20,24,38^FS
^RFW,E^FD48,1,6,770289,10001025,1^FS
^XZ
```

These commands would put

- `48` in the header
- `1` as the filter value
- 6 as the partition (indicates a 20-bit prefix and 24-bit item reference)
- `770289` as the company prefix
- `10001025` as the item reference
- `1` as the serial number

To read this EPC data and print the results on the label, you would use the following code:

```
^XA
^RB96,8,3,3,20,24,38^FS
^FO50,50^A0N,40^FN0^FS
^FN0^RFR,E^FS
^XZ
```

The resulting label would look like this:

^RF

Use this command to read or write to (encode) an RFID tag or to specify the access password.

Read or Write RFID Format

When using this command to read a tag, you may use a field variable to print the tag data on the label or to return the data to the host. For more information on how memory is stored on a Gen 2 tag or for examples that use a field variable, refer to the RFID Programming Guide for your printer. A copy of the manual is available at http://www.zebra.com/manualshttp://www.zebra.com/manuals.

Format: `^RFo,f,b,n,m`

Parameters	Details
`o` = operation	Specifies the action to be performed. Values: `W` = write to (encode) the tag L = write with LOCK (if supported by tag type; Gen 2 tag type does not use this locking function)R = read the tag P = read password (Gen 2 tag type only. Not supported on all Gen 2 printers, including the ZD500R printer.) S = specify the access password **Default:** `W`
`f` = format	**Values:** A = ASCII H = Hexadecimal E = EPC (ensure proper setup with the ^RB command) **Default:** `H`

Parameters	Details
b = passwordORb = starting block number	**For Gen 2 tag type only:** What you specify for this parameter depends on what you enter for other parameters. **NOTE:** When the Gen 2 memory bank parameter is set to E (EPC 96-bit) or A (EPC and Auto adjust PC bits), W and R values are always set to 2. **If the Operation parameter value is...** **W** **Values:** P, which indicates that an access password, a kill password, or both follow in a ^FD command. Each password must be 8 hex characters. If the password is omitted, it is not written. An access password is used in subsequent lock commands in the format. 0 to n, which specifies the 16-bit starting block number, where n is the maximum number of blocks for the bank specified in the memory bank parameter. **Default:** 0 **R** **Values:** 0 to n, which specifies the 16-bit starting block number, where n is the maximum number of blocks for the bank specified in the memory bank parameter. **Default:** 0 **S** This parameter must be P and must be followed by the access password in a ^FD command. **For tag types other than Gen 2:** Specifies the starting block number. **Values:** 0 to n, where n is the maximum number of blocks for the tag. **Default:** 0
n = number of bytes to read or write	Specifies the number of bytes to read or write. **For high-frequency (HF) printers:** **Values:** 1 to n, where n is the maximum number of bytes for the tag. **Default:** 1 **For Gen 2 tag type only:** When E or A is specified for the memory bank parameter, this value is not required. **Values:** 1 to n, where n is the maximum number of bytes for the tag. **Default:** 1 **For all other printers and tag types:**This parameter applies only when the starting block number is 1. **Values:** 1 to n, where n is the maximum number of bytes for the tag. For UCODE EPC 1.19, n is 32. **Default:** 1

Parameters	Details
m = Gen 2 memory bank	**NOTE:** This parameter applies to Gen 2 tags only. Specifies the Gen 2 memory bank. For more information about Gen 2 memory, refer to the RFID Programming Guide for your printer. E = EPC 96-bit (When writing data, this parameter performs the operation on Gen 2 bit address 20_h and accesses 12 bytes of the EPC memory bank. When reading data, this parameter reads the amount of data specified in the PC bits on the tag.) A = EPC and Auto adjust PC bits (When writing data, this parameter performs the operation on Gen 2 bit address 20_h of the EPC memory bank and accesses the number of bytes specified in the ^FD. The PC bits will be updated to match the amount of data written to the tag. When reading data, this parameter reads the amount of data specified in the PC bits on the tag. This value is supported only by the ZD500R printer and ZT400 Series and ZT600_Series RFID printers. 0 = Reserved 1 = EPC 2 = TID (Tag ID) 3 = User **Default:** E

Example: This example encodes 96-bit data in ASCII format. (The ^RS command can be omitted for printers that use Gen 2 tag types only.)

```
^XA
^RS8
^RFW,A^FD00 my data^FS
^XZ
```

Example: This example encodes 96-bit EPC data, as specified by the ^RB command.

```
^XA
^RB96,8,3,3,20,24,38
^RFW,E^FD16,3,5,78742,146165,1234567891^FS
^XZ
```

Example: This example encodes 4 bytes of hexadecimal formatted data, starting in block 3 of Gen 2 EPC bank 1. (The ^RS command can be omitted for printers that use Gen 2 tag types only.)

```
^XA
^RS8
^RFW,H,3,4,1^FD11112222^FS
^XZ
```

Example: This example reads the extended Gen 2 tag ID (TID), which is not read by the ^RI command, and returns the results to the host computer. The results are labeled with the header "8-byte Tag ID Data." (The ^RS command can be omitted for printers that use Gen 2 tag types only.)

```
^XA
^RS8
^RFR,H,0,8,2^FN1^FS^HV1,,8-byte Tag ID Data:^FS
^XZ
```

Example: This command writes and specifies both the access password (12345678) and the kill password (88887777) separated by a comma.

```
^RFW,H,P^FD12345678,88887777^FS
```

This command writes the access password only:

```
^RFW,H,P^FD12345678^FS
```

This command writes the kill password only (a comma must be used before it to distinguish it from an access password):

```
^RFW,H,P^FD,88887777^FS
```

See the examples for ^RL for how this command would be used in a format.

Example: This command writes 1122334455667788 to the bit address 20h of the EPC memory and updates the PC bits bit address 10h to 14h to reflect 8 bytes (4 words) of data.

```
^RFW,H,,,A^FD1122334455667788^FS
```

Example: This command specifies the access password for the tag, which will be used in subsequent lock commands in the format. The access password specified must match the one stored on the tag. This command does not write the password to the tag. See the examples for ^RL for how this command would be used in a format.

```
^RFS,H,P^FD12345678^FS
```

^RL

Use this command to lock/unlock RFID tag memory.

Lock/Unlock RFID Tag Memory

The ^RL command has four distinct formats and functions:

- ^RLP – `Permanently Lock All Tag Memory` Locks all memory banks and/or passwords, as defined by the chip manufacturer.
- ^RLB – `Permanently Lock Specified Memory Sections` Locks blocks of user memory in an unwriteable state.
- ^RLM – `Lock/Unlock the Specified Memory Bank` Locks a password or an entire memory bank in a writeable or unwriteable state. These locks/unlocks can be permanent or reversible.

^RLP – Permanently Lock All Tag Memory

Some chip manufacturers have implemented an alternative permalocking mechanism that must be applied to all memory by permalocking certain banks and/or passwords. The RFID chips' datasheet may specify permalocking any combination of the Kill, Access, EPC, User, and TID memory to permalock the RFID tag. The ^RLP command automatically selects the required memory locations to permalock the RFID tag for a particular chip.

NOTE: The access password is not required for this command. The printer will use the default of `00000000`.

Format: ^RLP

^RLB – Permanently Lock Specified Memory Sections

The ^RLB command permanently locks (permalocks) one or more sections (individual sub-portions) in a tag's user memory. The section sizes for each tag is defined by the tag manufacturer.

Format: ^RLB,s,n

Parameters	Details
s = starting section	Specify the starting section of memory to lock.
n = number of sections	Specify the number of sections to lock.

^RLM – Lock/Unlock the Specified Memory Bank

The ^RLM command locks/unlocks the specified password or memory bank on an RFID tag. You can use this command to do the following:

- lock individual passwords, thereby preventing or allowing subsequent reads or writes of that password
- lock individual memory banks, thereby preventing or allowing subsequent writes to those banks
- Permanently lock (permalock) the lock status for a password or memory bank

Format: ^RLM,k,a,e,u

Parameters	Details
k = kill password function	**Values:** U = unlock the kill password* L = lock the kill password* 0 = permanently unlock (Open) the kill password P = permanently lock (Protected) the kill password
a = access password function	**Values:** U = unlock the access password* L = lock the access password* 0 = permanently unlock (Open) the access password P = permanently lock (Protected) the access password
e = EPC memory bank function	**Values:** U = unlock the EPC memory bank* L = lock the EPC memory bank* 0 = permanently unlock (Open) the EPC memory bank P = permanently lock (Protected) the EPC memory bank
u = USER memory bank function	**Values:** U = unlock the USER memory bank* L = lock the USER password bank* 0 = permanently unlock (Open) the USER memory bank P = permanently lock (Protected) the USER memory bank
* The access password must be set to something other than the default of `00000000` to use this value. See the examples for this command for guidance.	

Examples

^RLM Example 1: The following command locks all memory banks using a previously specified access password.

```
^RLM,L,L,L,L^FS
```

^RLM Example 2: The following command locks the user memory banks using a previously specified access password.

```
^RLM,,,,L^FS
```

^RLB Example: The following command permalocks sections 0 to 4 of user memory using a previously specified access password.

```
^RLB,0,4^FS
```

Combination ^RLM and ^RLB Example 1: This code does the following:

- writes 12 bytes to user memory

- writes 12345678 to the access password and 11223344 to the kill password
- permalocks 6 sections of user memory using 12345678 as the access password
- locks the kill and access passwords and permanently unlocks the EPC memory, using 12345678 as the access password

```
^XA
^RFW,H,0,12,3^FD112233445566778899001122^FS
^RFW,H,P^FD12345678,11223344^FS
^RLB,0,6^FS
^RLM,L,L,O^FS
^XZ
```

Combination ^RLM and ^RLB Example 2: This code does the following:

- writes 12 bytes to user memory
- permalocks 6 sections of user memory using 00000000 as the access password
- permalocks the kill password and access password using 00000000 as the access password

```
^XA
^RFW,H,0,12,3^FD112233445566778899001122^FS
^RLB,0,6^FS
^RLM,P,P^FS
^XZ
```

^RS

Use this command to set up RFID parameters including tag type; programming position; and error handling, such as setting the number of labels that will be attempted if an error occurs.

Set Up RFID Parameters

For example, if an RFID label fails to program correctly or if the transponder cannot be detected, the printer ejects the label and prints`VOID`across it. The printer will try to print another label with the same data and format for the number of labels specified (parameter`n`). If the problem persists, the printer follows the error handling instructions specified by the error handling parameter (parameter`e`): the printer may remove the problematic format from the print queue and proceed with the next format (if one exists in the buffer), or it may place the printer in Pause or Error mode.

IMPORTANT: Use care when using this command in combination with ^RF for reading tag data. Use care when using this command in combination with ^RT or ^RF for reading tag data. Problems can occur if the data read from the tag is going to be printed on the label. Any data read from the tag must be positioned to be printed above the read/write position. Failure to do this will prevent read data from being printed on the label.

Format: `^RSt,p,v,n,e,a,c,s`

Parameters	Details
`t` = tag type	**Values:** 8 = EPC Class 1, Generation 2 (Gen 2) **Default:** 8—Gen 2 is the only tag type supported by current RFID printers. For tag types supported by older printers, refer to the original RFID Programming Guide.

Parameters	Details
p = read/write position of the tag (programming position)	This parameter sets the read/write position of the tag. **IMPORTANT:** If a label format specifies a value for the programming position, this value will be used for the programming position for all labels until a new position is specified or until the tag calibration procedure is run. **For Link-OS printers:** **Values:** F0 to Fxxx (where xxx is the label length in millimeters or 999, whichever is less) The printer prints the first part of a label until it reaches the specified distance and then begins programming. After programming, the printer prints the remainder of the label. B0 to B30 The printer backfeeds the label for the specified distance and then begins programming. To account for the backfeed, allow empty media liner to extend out of the front of the printer when using a backward programming position. up = move to the next value down = move to the previous value **Default:** F0 (which moves the leading edge of the label to the print line) **For older RFID printers:** **Values:** **Absolute Mode** (all firmware versions): xxxx = 0 to label length (in dot rows). Move the media to the specified position xxxx on the label, measured in dot rows from the label top, before encoding. Set to 0 (no movement) if the tag is already in the effective area without moving the media. **Relative Mode** (firmware versions V53.17.6 and later): F0 to Fxxx (where xxx is the label length in millimeters or 999, whichever is less). The printer prints the first part of a label until it reaches the specified distance and then begins programming. After programming, the printer prints the remainder of the label. B0 to B30 (Does not apply to the RP4T printer.) The printer backfeeds the label for the specified distance and then begins programming. To account for the backfeed, allow empty media liner to extend out of the front of the printer when using a backward programming position. Default: For the R2844-Z and RPAX: 0 (no movement) For printers using V53.17.6, V74.19.6Z, and later: F0 (which moves the leading edge of the label to the print line) All others: label length minus 1 mm (1/16 in.)
v = length of void printout	Sets the length of the void printout in vertical (Y axis) dot rows. **Values:** 0 to label length **Default:** label length

Parameters	Details
n = number of labels to try encoding	The number of labels that will be attempted in case of read/encode failure. **Values:** 1 to 10 **Default:** 3
e = error handling	If an error persists after the specified number of labels are tried, perform this error handling action. **Values:** N = No action (printer drops the label format causing the error and moves to the next queued label) P = Place printer in Pause mode (label format stays in the queue until the user cancels) E = Place printer in Error mode (label format stays in the queue until the user cancels) **Defaults:** N **NOTE:** You can set the printer to send an error message to the host for each failure. To enable or disable this unsolicited error message, refer to the^SX and ^SQ ZPL commands. Use V for the condition type for an RFID error.
a = signals on applicator	**NOTE:** This parameter applies only to older RFID printers that have an applicator board. This parameter does not apply to the R2844-Z or to Link-OS printers. For the R4Mplus, this parameter applies only to printers with firmware version SP994X (R4Mplus European version). **Single Signal Mode** In this mode, one start print signal starts printing. Then, at the program position (parameter p), the printer automatically stops and encodes the tag. Printing continues, and a single end print signal signifies the completion of the label. **Double Signal Mode** With RFID, when there is a non-zero program position, the label is logically split into two parts. The first part is printed, the tag encodes, and then the second part prints. If this parameter is set to “D,” then the label is split into two and requires both portions of the label to be controlled by the applicator. This means that a start print signal triggers the first portion of the label, and then when the printer reaches the RFID program position (and the motor stops), an end print signal is provided. In this mode, a second start print signal is required to print the rest of the label. When the label is complete, a final end print signal is provided. **NOTE:** If parameter p is zero, then single signal mode is used (parameter ignored). If p is F0 (or B0) with backfeed-after, then single signal mode is used (parameter ignored). **Values:** S = single signal D = double signal (For the R110PAX4, Double mode will work only if the read/write position is changed from the default of zero.) **Default:** S
c = reserved	Not applicable.

Parameters	Details
s = void print speed	**NOTE:** This parameter is not supported on all printer models. If a label is voided, the speed at which “VOID” will be printed across the label. **Values:** any valid print speed **Default:** the printer’s maximum print speed

Example: The following are examples of Absolute Mode and Relative Mode for the tag position parameter (parameter p).

Absolute Mode

^RS,520 sets the encode position at 520 dots from the top edge of the label.

^RS,0 programs the tag without moving the media.

Relative Mode

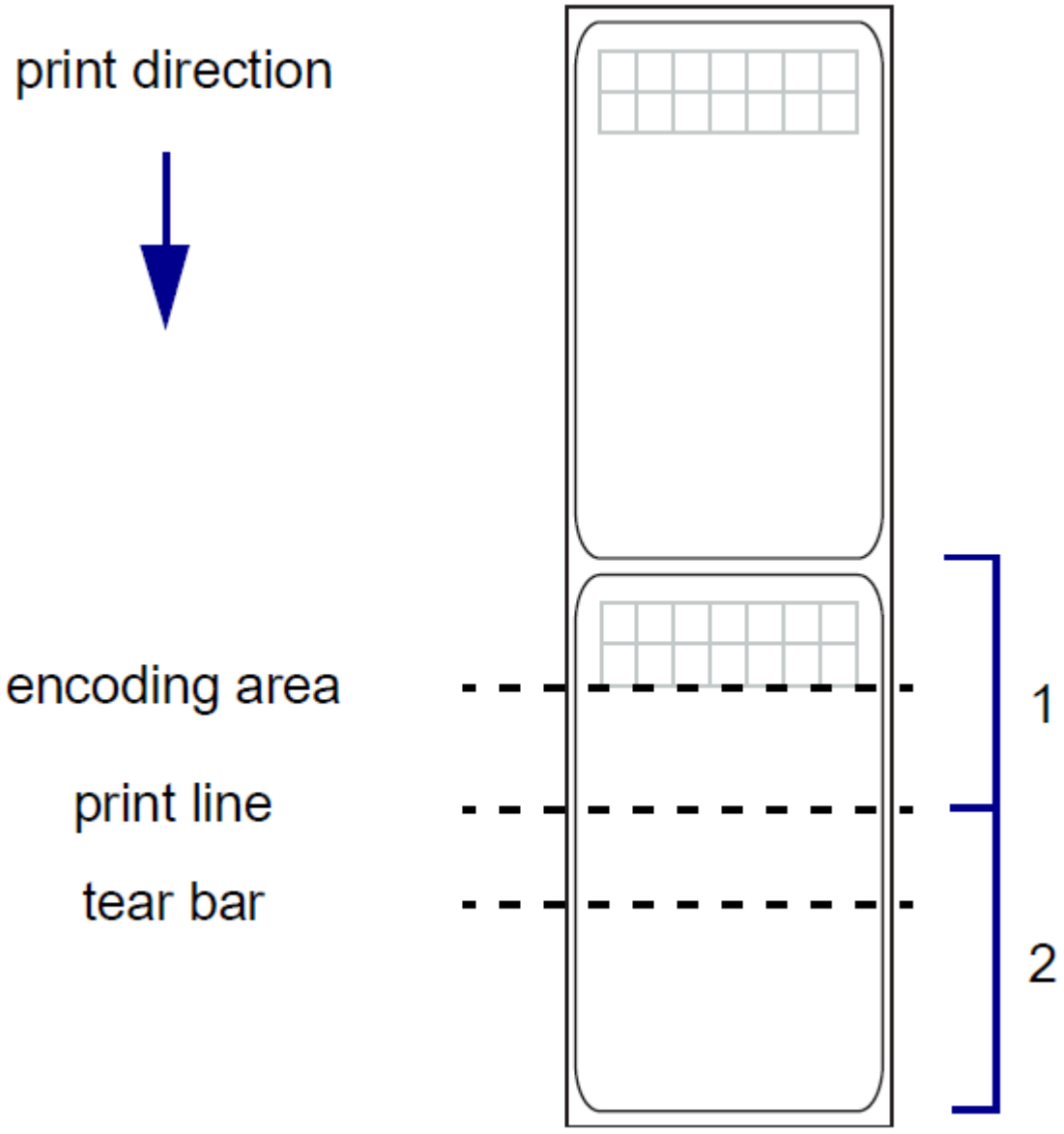
^RS,F1 sets the encode position 1 mm forward from the leading edge of the label.

^RS,B10 sets the encode position 10 mm backwards from the leading edge of the label.

^RS,F0 sets the encode position at the leading edge of the label.

^RS,B0 sets the encode position at the leading edge of the label.

Example: The following shows the difference between absolute and relative programming positions for the tag position parameter (parameter p) with a 6-inch (152-mm, 1216-dot) label length. The end results are that the tag is programmed with the label in the same position.



1	^RS, 496, Absolute Mode, 496 dots from the top of the label
2	^RS, F90, Relative Mode, 90 mm from the leading edge of the label

^RU

Use this command to read the TID (Tag ID) data from the current chip and format a unique 38-bit serial number, which will be placed in the lower (least significant) 38 bits of the EPC code.

Read Unique RFID Chip Serialization

Format: ^RUa,b

Parameters	Details
a = prefix	Specifies the prefix in ASCII Binary **Values:** Only ASCII characters 1 and 0 are accepted. Maximum of 38 characters. The number of bits in the value specifies the length of the prefix. The prefix is placed as the left-most (most significant) bits in the unique serial number. If nothing is specified, the default value will be used. **Default:** The MCS prefix is determined by the MDID in the TID of the chip read: 100 = EM Micro Impinj = 101 Alien = 110 NXP = 111
b = special character	Special character for serial number inclusion. **Values:** Any ASCII character other than the current Command character, Control character, Delimiter character, or any of the Real-Time Clock (RTC) characters. **Default:** #

NOTE: Serial number inclusion:

One of several data elements can be included into any ^FD data string in the same way that Real Time Clock data is included. Use any of the commands below to include a data pattern based on the serial number. These are defined using the default value for the Special Character.

#S = include 38-bit serial number derived from TID in decimal form.

#H = include 38-bit serial number derived from TID in hexadecimal form.

#E = include the entire 96-bit EPC code, including the 38-bit serial number derived from TID in decimal form.

#F = include the entire 96-bit EPC code, including the 38-bit serial number derived from TID in hexadecimal form.

#P = include the entire 96-bit EPC code, but use the tag's preprogrammed, 38-bit SGTIN serial number in decimal form.*

#Q = include the entire 96-bit EPC code, but use the tag's preprogrammed, 38-bit SGTIN serial number in hexadecimal form.*

* If the EPC has been preprogrammed (typically by the manufacturer) with the chip-based RFID serialization scheme, then the serialized data does not have to be written back to the EPC memory, which saves time. #P and #Q simply format the data that is read from the EPC memory bank.

Example: Read the TID from the tag, create a serial number based on the tag type, write `12<serial number (5 bytes)>000000000000` to the 96-bit EPC field, and print the serial number (in hex format) on the label.

```
^XA
^RU
^FO10,10^A0N,50,50^FDSerial Number: #H^FS
^RFW,H^FD12#H^FS
^XZ
```

Example: Read the TID from the tag, create a serial number based on the tag type, write the serial number to the EPC field (lower 38 bits) while maintaining the contents of the rest of the EPC memory, print `Serial Number: <serial number in hex format>` on the label, and return `Serial Number: <serial number in hex format>` to the host. Perform this operation on three label formats.

```
^XA
^RU
^FO10,10^A0N,50,50^FN1^FS
^FN1^FDSerial Number: #H^FS
^FH^HV1,24, ,_0D_0A,L^FS
^RFW,H^FD#F^FS
^PQ3
^XZ
```

Example: Read the full EPC (already serialized) from the tag, print `Serial Number: <full EPC in decimal format>` on the label, and return `Serial Number: <full EPC in decimal format>` to the host.

```
^XA
^RU
^FO10,10^A0N,50,50^FN1^FS
^FN1^FDSerial Number: #P^FS
^FH^HV1,44, ,_0D_0A,L^FS
^XZ
```

^RW

Use this command to set the RFID read and write power levels if the desired levels are not achieved through RFID tag calibration. If not enough power is applied, the tag may not have sufficient power for programming, and tag data will fail to encode. If too much power is applied, the extra power may cause data communication errors or may cause the wrong tag to be programmed.

Set RF Power Levels for Read and Write

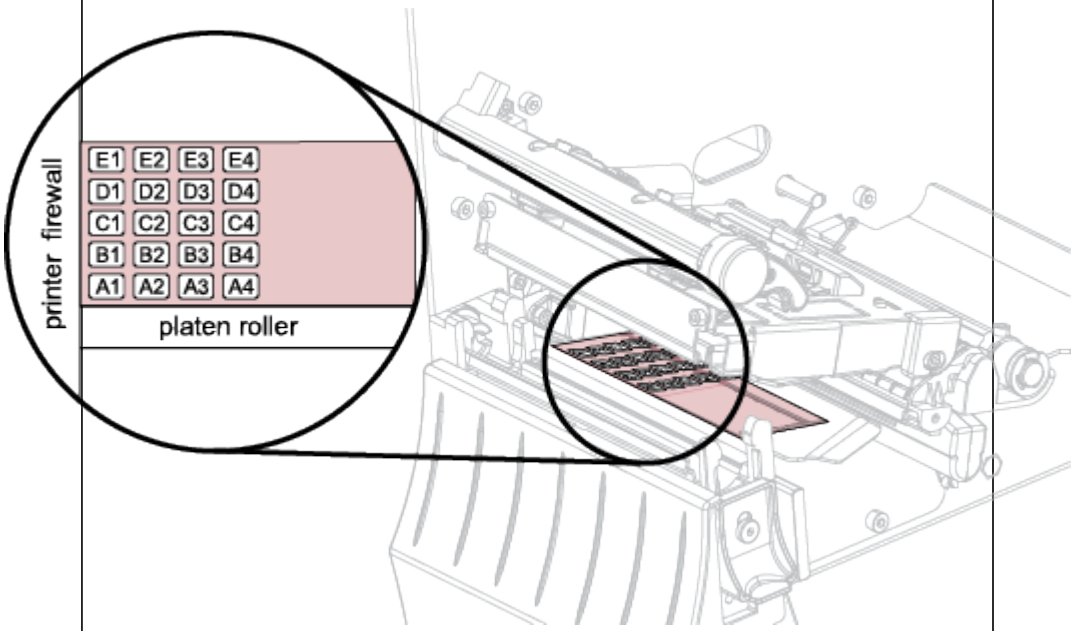
NOTE: Printers automatically select the best antenna element and read/write power levels for the media during RFID transponder calibration. The R110Xi4, ZT400 series, and ZT600 series printers also may set the levels during an adaptive antenna sweep. Use ^HL or ~HL on page 412 to view the antenna element and power settings being used.

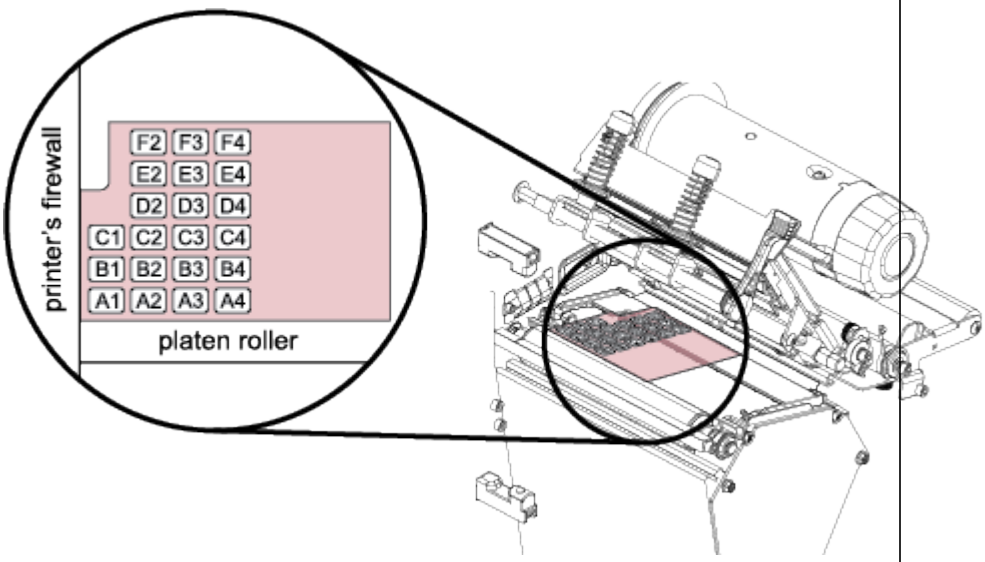
NOTE: For Japan, the printer's maximum RFID read and write power are limited to comply with local radio regulations. Any power setting of 24 or higher results in the same output.

Format: ^RWr,w,a

Parameters	Details
r = read power	This parameter sets the power level to match the desired output as calibrated in the factory. **R53.16.3, V53.17.5, and later:** **Values:** 0 to 30 **Default:** 16 **R60.16.4, R62.16.4, R63.16.4, SP994Q, SP999G, SP1027G, SP1056F, SP1082G, and later:** **Values:** 0 to 30, H (high), M (medium), L (low) **Default:** L **R65.X and older versions of other firmware:** **Values:** H = high M = medium L = low **Default:** L

Parameters	Details
w = write power	**NOTE:** This parameter is ignored on the R110Xi HF printer (firmware version R65.X) because read and write powers cannot be specified separately. The printer uses the value that you specified for read power for both the read and write power settings. This parameter sets the power level to match the desired output as calibrated in the factory. **R53.16.3, V53.17.5, and later:** **Values:** 0 to 30 **Default:** 16 **R60.16.4, R62.16.4, R63.16.4, SP994Q, SP999G, SP1027G, SP1056F, SP1082G, and later:** **Values:** 0 to 30, H (high), M (medium), L (low) **Default:** L **Older versions of firmware:** **Values:** H = high M = medium L = low **Default:** L

Parameters	Details
a = RFID antenna element selection	**ZD500R, ZQ511/ZQ521, and ZQ630:** This printer only has one antenna element, so the value used is always A1. **ZT400 and ZT600:** This parameter specifies the RFID antenna to be used for RFID operation. E1, E2, E3, E4 D1, D2, D3, D4 C1, C2, C3, C4 B1, B2, B3, B4 A1, A2, A3, A4  A4 **(Continued on next page)**

Parameters	Details
a = RFID antenna element selection	**(Continued from previous page)** **R110Xi4 (V53.17.5 and later):** This parameter specifies the RFID antenna to be used for RFID operation. **Values:** A1, A2, A3, A4, B1, B2, B3, B4, C1, C2, C3, C4, D2, D3, D4, E2, E3, E4, F2, F3, F4 (combinations D1, E1, and F1 are invalid)  **Default:** A4 **R110Xi HF (R65.X):** This parameter selects the antenna port that provides the best results for reading and writing. **Values:** 1 = antenna port 1 2 = antenna port 2 **Default:** 1

Example: The following command selects the antenna at row D, column 3 on an R110Xi4 printer:

```
^RW,,D3
```

Example: The following command sets the read/write power level to Medium and selects antenna 2 on an R110Xi HF printer:

```
^RWM,,2
```

Example: The following command sets the read and write power levels to High on an R110PAX4 printer:

```
^RWH,H
```

ZBI Commands

This section explains the Zebra Basic Interpreter, its commands, descriptions, formats, and parameters.

Introduction to Zebra Basic Interpreter (ZBI)

ZBI is an "on-the-printer" programing language that offers many of the functions found in ANSI BASIC. The ZBI language allows the user to create applications that are run on the printer to manipulate data streams. By using ZBI, it is possible to have the printer perform the same functions that a computer or programmable terminal might otherwise be used for.

With the connectivity options available on Zebra printers, you may not need a separate computer. Simply load a ZBI program on your printers, add them to your network, and let the printers serve as the gateway for moving data.

Here are some of the applications that can be written using ZBI:

- Connect a barcode scanner to the printer. Based on scanned data, reprint tags, verify printed output, and manage a list of items.
- Connect a scale to the printer and print labels, tags, or receipts based on the weight of an item.
- Connect the printer to a PC-based database and send queries from the printer to retrieve or upload data.
- Convert incoming data into the commands that can be used to print a label. This is useful for replacing other brands of printers with new Zebra units.
- Provide fail-over to another printer when the target printer is in an error state.

Printers, ZBI Keys, & ZBI Versions

Information about ZBI 1.x and ZBI 2.x:

ZBI versions 1.0 through 1.5:

ZBI 1.x was available on printers with X.10 or higher firmware (such as V48.10.x). To determine if the printer supports ZBI version 1, check the firmware version loaded on the printer. This can be determined by the absence of a "Z" in the firmware version number (for example, firmware V60.13.0.12 supports ZBI version 1, while V60.13.0.12Z does not).

ZBI-Developer can be used to create programs for use on printers that support ZBI version 1.x., however, the features that are only available in ZBI v2.x cannot be used with printers running ZBI v1.x. For example, "on-printer" debugging advanced file encryption and commands added in ZBI 2 are not supported in printers running ZBI 1.x. If you do not have a printer that meets this requirement, contact your reseller.

NOTE: Support for ZBI versions 1.0 through 1.5 is limited to syntax checking only. On-printer debugging is not supported for ZBI versions 1.0 through 1.5.

ZBI versions 2.0 and higher:

Printers with firmware versions X.16 or later (for example, V60.16.x and V53.16.x) can support ZBI version 2.0 and later.

These printers can be either ZBI-Ready or ZBI-Enabled, depending on whether or not a ZBI Key file has been loaded on the printer. ZBI Keys can be loaded onto printers during manufacturing or later purchased at www.zebrasoftware.com. A Downloader Utility/ZBI Key Manager software utility is available to assist in the task of sending ZBI Keys to printers.

The ZBI.nrd file is required to be present on the printer for ZBI 2.0 to be enabled. The ZBI Key is stored on the printer's E: memory location with the name ZBI.nrd. The file is persistent. It cannot be deleted even if the printer's memory is initialized. For example, if the ^JB command is used to initialize the location, the ZBI Key file will not be deleted.

When a printer is ZBI-Ready but not ZBI-Enabled, the firmware version will display a "Z" at the end of the version string (for example, V60.16.0Z). Additionally, the printer's configuration label will show that the printer is not ZBI-Enabled.

When a printer is ZBI-Enabled, the firmware version will not display a "Z" at the end of the version string (for example, V60.16.0Z). Additionally, the printer's configuration label will show that the printer is ZBI-Enabled.

NOTE: Each single ZBI Key can only be used once. When multiple printers are to be ZBI-Enabled, multiple Keys will be needed. The ZBI Key cannot be retrieved from printer to a host system.

Command and Function Reference Format

This section describes how commands and functions are presented in this document.

Function Rules

Functions built into this interpreter can be used in expressions only. The function names are not case sensitive.

If input parameters exist, they are enclosed in parentheses. If no parameters exist, no parentheses are used.

Variables referenced in the functions could be substituted by functions or expressions of the same type. If the function name ends with a $, it returns a string value. Otherwise, it returns a numeric value.

Command/Function NAME

Describes how the command is used, its capabilities, and its characteristics.

Format

The Format section explains how the command is arranged and its parameters. For example, the AUTONUM command starts the auto-numbering option. The format for the command is AUTONUM <A>,<B>. The <A> and <B> are parameters of this command and are replaced with values determined by the user.

For functions, parameters are enclosed within parentheses and separated by commas, such as EXTRACT $(A$,START$,STOP$).

Numeric parameters are written as a name, while string parameters are written as a name followed by a dollar sign.

Parameters

If a command has parameters that make a command or function more specific, they are listed under this heading. Still using the `AUTONUM` example, the <A> parameter is defined as:

<A> = number used to start the auto-numbering sequence

Return Value (functions only)

The return value is the result of evaluating the function or expression.

Example

When a command is best clarified in a programming context, an example of the ZBI code is provided. Text indicating parameters, exact code to be entered, or data returned from the host is printed in the `Courier` font to be easily recognizable.

An example of `PRINT` code is:

```
10 PRINT "HELLO WORLD"
RUN
HELLO WORLD
```

Comments

This section is reserved for notes that are of value to a programmer, warnings of potential command interactions, or command-specific information that should be taken into consideration. An example comment could be: This is a program command and must be preceded by a line number.

Section Organization

The sections in this guide are arranged based on programming topics. A brief description of the sections is listed below.

Editing Commands

This section describes the commands which are used to manipulate the interpreter and enter programs.

Running and Debugging

Outlines the control commands used to run and debug programs.

Base Types and Expressions

Fundamental structure for manipulating strings and computing numeric and boolean values.

Control and Flow

Commands to conditionally execute code and control the flow of the program

Input and Output

Outlines how to communicate with the physical ports, internal ports, and network.

File System

Shows how programs and formats can be saved and recalled

Comma Separated Values

Identifies how to load and store comma separated data

Events

Explains how to capture and trigger internal events in the printer

Systems

Contains miscellaneous systems interface functions

String Functions

Handles string manipulation

Math Functions

Handles mathematical calculations

Array Functions

Describes how to search, resize, and query arrays

Time and Date Functions

Functions to access the real time clock option

Set/Get/Do Interface

Functions to directly interface with the Set/Get/Do system

Example Programs

More examples to give a head start in creating your applications

Writing ZBI Programs

There are two main ways to develop ZBI programs. The preferred method is to use the ZBI-Developer application.

ZBI-Developer allows you to create and test programs before a printer is even turned on. In addition, many features of this program allow for quicker program creation and more meaningful debugging. ZBI-Developer can be downloaded from the Zebra web site.

An alternate method for developing a program is through a direct connection to the printer using a terminal emulation program.

Editing Commands

This section details the Editing Commands. This section describes the commands which are used to manipulate the interpreter and enter programs. These commands are used while controlling the ZBI environment from a console connection. Here is a quick list of these commands.

NEW

Clears out the program and variables currently in memory

REM and !

Comment commands

LIST

Lists the program currently in memory

AUTONUM

Automatically generates the next line number

RENUM

Renumbers the program currently in memory

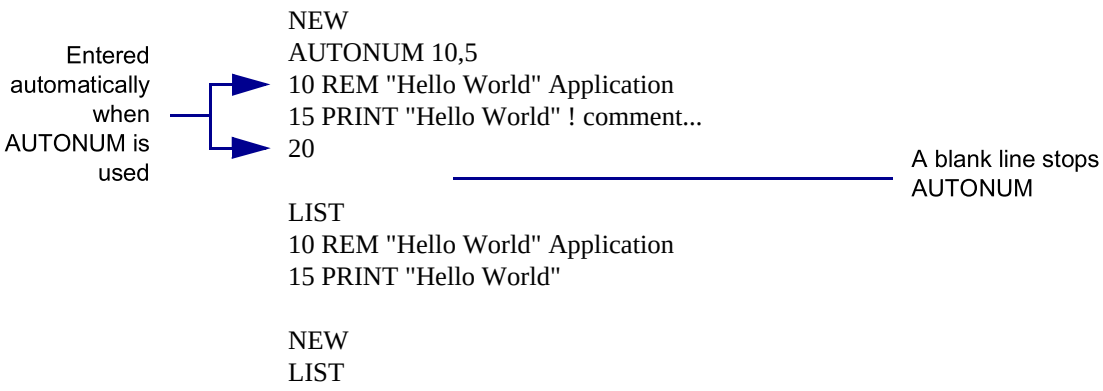
ECHO

Controls whether characters received on the console are echoed back

If you are using ZBI-Developer, the commands that will be most useful are AUTONUM and REM/!.

The following example shows the use of Editing commands from within a console connection.

Preview



Preview when viewed in ZBI-Developer

```
AUTONUM 10,5
REM "Hello World" Application
PRINT "Hello World" ! comment...
```

NEW

This command clears the interpreter's memory, including the line buffer and variables, but not any open ports. Use this command when creating code to restart the coding process or before resending a program from a file to the interpreter.

Format
NEW

Parameters
N/A

Example
This is an example of how to use the NEW command:

```
10 PRINT "Hello World"
RUN
Hello World

LIST
10 PRINT "Hello World"

NEW
LIST
```

Comments
This is an interactive command that takes effect as soon as it is received by the printer.

REM

A numbered remark line starts with REM and includes text in any form after it. This line is ignored by the interpreter.

Format

```
REM <comment>
```

Parameters

The comment string can contain any character and is terminated by a carriage return.

Example

This is an example of how to use the REM command:

```
10 REM COMMAND LINES 20-100 PRINT A LABEL
```

Comments

Remarks are used for program description and are included as a separate program line. To append a comment to the end of a program line, use the exclamation mark (!).

A useful method to keep comments in a stored file (but not in the printer) is to always start the REM line with the number 1. When all of the lines are sent to the printer, only the last REM line will stay resident in the printer. This will require less RAM for large programs.

Example

This is an example of how to re-use the REM command:

```
1 REM MYPROGRAM COPYRIGHT ME Inc. 2008
1 REM While debugging a port may be left open
5 CLOSE ALL
1 REM Open the ports this program will use
10 OPEN #0: NAME: "SER" ! Restart the console
```

! (EXCLAMATION MARK)

The exclamation mark is the marker for adding comments to the end of numbered programming lines. Any text following the ! is ignored when the line or command is processed.

Format

```
!<comment>
```

Parameters

The comment string can contain any character and is terminated by the carriage return.

Example

This is an example of how to use the ! (comments) command:

```
10 LET A=10 ! Indicates number of labels to print
```

Comments

None

LIST

This command lists the program lines currently in memory.

Format

`LIST`

`LIST <A>`

`LIST <A>-<B>`

Parameters

default = lists all lines in memory

<A> = line to start listing the program

<B> = line to stop listing the program. If not specified, only the line at <A> will print.

Example

This is an example of how to use the LIST command:

```
1 REM MYPROGRAM COPYRIGHT ME Inc. 2008
1 REM While debugging a port may be left open
5 CLOSE ALL
1 rem Open the ports this program will use
10 OPEN #0: NAME: "SER" ! Restart the console
20 PRINT #0: "Hello World"
LIST
1 REM Open the ports this program will use
5 CLOSE ALL
10 OPEN #0: NAME: "SER" ! Restart the console
20 PRINT #0: "Hello World"

LIST 1
1 REM Open the ports this program will use

LIST 5-10
5 CLOSE ALL
10 OPEN #0: NAME: "SER" ! Restart the console
```

Comments

The output of the LIST command may not match exactly what was entered. It is based on how the program lines are stored in memory. Notice that the last comment line the REM is entered in lower case characters. When it is listed, the REM is displayed in uppercase.

This is an interactive command that takes effect as soon as it is received by the printer.

AUTONUM

This command automatically generates sequential program line numbers.

Format

AUTONUM <A>,<B>

Parameters

A = the number used to start the auto-numbering sequence

B = the automatic increment between the new line numbers

Example

This example shows specifying the starting line number in the increment between new line number. Type the following at the prompt:

```
AUTONUM 10,5
SUB START
PRINT "HELLO WORLD"
GOTO START

LIST
```

Will produce:

```
AUTONUM 10,5
10 SUB START
15 PRINT "HELLO WORLD"
20 GOTO START
```

The three lines are automatically started with the AUTONUM parameters; in this case, the first line starts with 10 and each subsequent line increments by 5.

Comments

This feature is disabled by overwriting the current line number and entering the desired interactive mode commands, or leaving the line blank.

Use of the SUB command allows for GOTO and GOSUB statements that do not require line numbers in your program.

This is an interactive command that takes effect as soon as it is received by the printer.

RENUM

This command renumbers the lines of the program being edited. RENUM can reorganize code when line numbers become over- or under-spaced. The line references following GOTO and GOSUB statements are renumbered if they are constant numeric values. Renumbering does not occur if the line numbers are outside of the range limits of 1 to 10000.

Format

RENUM <A>,<B>

Parameters

<A> = the number to start the renumbering sequence

<B> = the automatic increment between the new line numbers

Example

This is an example of how to use the RENUM command:

```
LIST
13 LET A=6
15 LET B=10
17 GOTO 13
RENUM 10,5
LIST
10 LET A=6
15 LET B=10
20 GOTO 10
```

NOTE: The target of the GOTO command changes from 13 to 10 to reflect the renumbering.

Comments

This is an interactive command that takes effect as soon as it is received by the printer.

ECHO

When Console Mode is enabled, this command controls whether the printer echoes the characters back to the communications port. If ECHO ON is entered, keystroke results return to the screen. If ECHO OFF is entered, keystroke results do not return to the screen.

Format

```
ECHO ON
ECHO OFF
```

Parameters

<ON/OFF> = toggles the ECHO command on or off

Example

N/A

Comments

This can be an interactive command that takes effect as soon as it is received by the printer, or a program command that is preceded by a line number.

Running and Debugging Commands

The following commands were written before the development of the ZBI-Developer application. With that application, and when using ZBI version 1, the following commands are essentially obsolete. However, for those who started developing ZBI applications before ZBI-Developer, the following reference will be helpful.

RUN
Starts executing the program currently in memory at the first line of the program

CTRL-C
Sends an end-of-transmission character, ETX , to the console to terminate the ZBI program currently running.

RESTART
Starts executing the program currently in memory where it was last stopped

STEP
Executes one line of the program in memory where it was last stopped

DEBUG
This mode controls whether or not the TRACE and BREAK commands are processed

TRACE
Shows which lines have been executed and which variables have been changed

BREAK
Stops the currently running program

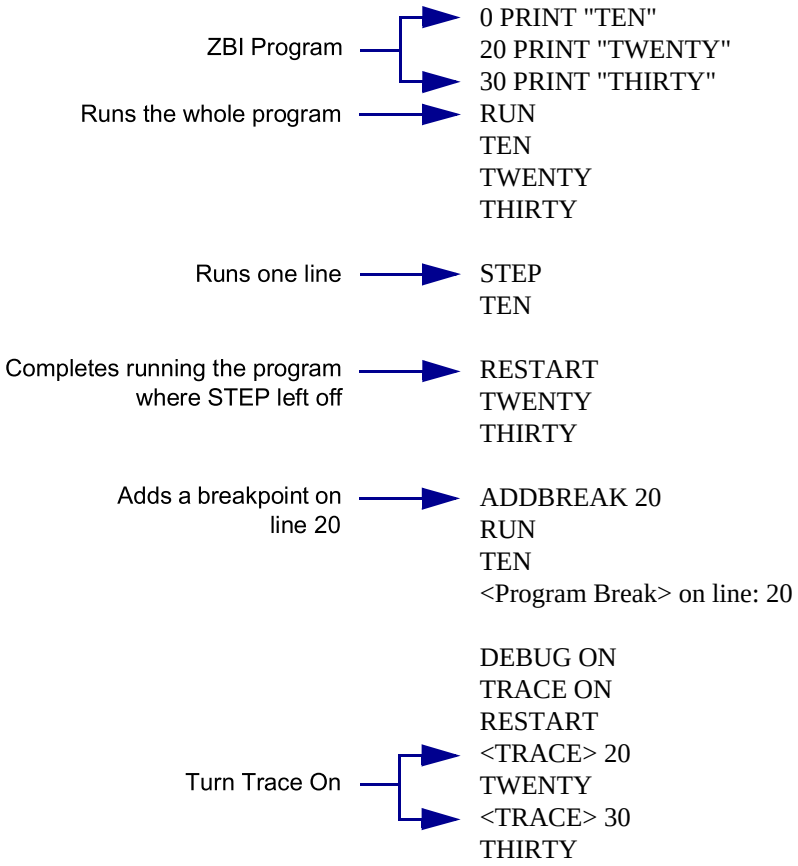
ADDBREAK
Adds a break to an existing line

DELBREAK
Deletes an existing break

ZPL
Terminates and exits the ZBI environment.

Example

This example shows many of the Running and Debug Commands in practice.



RUN

This command executes the current program, starting with the lowest line number. The interpreter will continue to execute the program lines in order unless a control statement directs the flow to a different point. When a higher line number does not exist or the END command is processed, the RUN command will stop.

Format

RUN

Parameters

N/A

Example

This is an example of how to use the RUN command:

```
10 PRINT "ZBI"
20 PRINT "Programming"
RUN
ZBI
Programming

15 END
RUN
ZBI
```

Comments

Ports that are open when the application is activated will remain open after the application has terminated. Variables also remain after the application has terminated.

To execute programs when the printer is powered on, use the ^JI command in the Autoexec.zpl file.

This is an interactive command that takes effect as soon as it is received by the printer.

CTRL-C

Sending an end-of-transmission character, ETX (3 in hex), to the console (port 0) terminates the ZBI program currently running.

Format
N/A

Parameters
N/A

Example
N/A

Comments
In most terminal programs, you terminate the program using the `Ctrl-C` key sequence. Another method is to store an ETX character in a file and have the terminal program send the file to the console port.

NOTE: It is not recommended to use RESTART after using a CTRL-C because a command may have been prematurely interrupted. Restarting will have an undefined result.

RESTART

If a program was halted by a break point or the BREAK command, the RESTART command can be used to reactivate the program at the point it stopped.

RESTART functions similar to RUN, except the program attempts to restart from the point where it was last terminated. It also works in conjunction with the STEP command, picking up where the STEP command ended.

Format

RESTART

Parameters

N/A

Example

An example of the RESTART command:

```
10 PRINT "TEN"
20 PRINT "TWENTY"
30 PRINT "THIRTY"
RUN
TEN
TWENTY
THIRTY

STEP
TEN

RESTART
TWENTY
THIRTY

ADDBREAK 20
RUN
TEN
<Program Break> on line: 20

DEBUG ON
TRACE ON
RESTART
<TRACE> 20
TWENTY
<TRACE> 30
THIRTY
```

Comments

If the program has not been run or has finished, RESTART runs the program from the beginning.

This is an interactive command that takes effect as soon as it is received by the printer.

STEP

If a program was stopped by a BREAK command, STEP attempts to execute the program one line from where it last ended. If the program has not been run or has been completed, this executes the lowest numbered line.

Format
STEP

Parameters
N/A

Example
This is an example of how to use the STEP command:

```
10 PRINT "Hello World"
20 Print "TWENTY"
STEP
Hello World

STEP
TWENTY
```

Comments
This is an interactive command that takes effect as soon as it is received by the printer.

DEBUG

DEBUG enables and disables the TRACE and BREAK commands.

Format

```
DEBUG ON

DEBUG OFF
```

Parameters

ON = turns the debug mode on enabling the TRACE and BREAK commands to be processed.

OFF = turns the debug mode off. This disables the TRACE mode and causes BREAK commands to be ignored.

Example

N/A

Comments

This command has no effect on the processing of break points in ZBI-Developer. It is recommended that you avoid using the DEBUG command when writing programs in the ZBI-Developer environment, instead use the Debug capabilities of ZBI-Developer.

TRACE

This command enables you to debug an application by outputting the executed line numbers and changed variables to the console.

Format

```
TRACE ON

TRACE OFF
```

Parameters

`<ON/OFF>` = controls whether TRACE is active (ON) or disabled (OFF).

If DEBUG is activated and the TRACE command is on, trace details are displayed. When any variables are changed, the new value displays as follows:

`<TRACE> Variable = New Value`

Every line processed has its line number printed as follows:

`<TRACE> Line Number`

Example

An example of TRACE command in use:

```
10 LET A=5
20 GOTO 40
30 PRINT "Error"
40 PRINT A
DEBUG ON
TRACE ON
RUN
<TRACE> 10
<TRACE> A=5
<TRACE> 20
<TRACE> 40
5
```

Comments

This can be an interactive command that takes effect as soon as it is received by the printer, or a program command that is preceded by a line number.

It is recommended that you avoid using the TRACE command when writing programs in the ZBI-Developer environment, instead use the Debug capabilities of ZBI-Developer.

BREAK

This command allows you to stop the program when the program reaches this line.

Format

BREAK

Parameters

N/A

Example

An example of BREAK command in use:

```
10 LET A=5
20 BREAK
30 PRINT A
DEBUG ON
TRACE ON
RUN
<TRACE> 10
<TRACE> A=5
<TRACE> 20
<USER BREAK>
```

Comments

This command is available only when the DEBUG function has been activated. When DEBUG is on, BREAK halts processing. RUN starts the program from the beginning. RESTART allows the program to continue from where it left off.

When using ZBI-Developer, this command will interfere with the debugging operations built into the application.

This is a program command that must be preceded by a line number.

ADDBREAK

This command allows you to stop the program when the program reaches a specified line.

Format

ADDBREAK <A>

Parameters

A = the line number to break on. If the number specified is not in the program, the program will not break.

Example

An example of the ADDBREAK command.

```
10 LET A=5
20 PRINT A
ADDBREAK 20
RUN
<PROGRAM BREAK> ON LINE:20

RESTART
5
```

Comments

This command is available only when the DEBUG function has been activated. When DEBUG is on, BREAK halts processing. RUN starts the program from the beginning. RESTART allows the program to continue from where it left off.

This is the command used internally by ZBI-Developer when the user right-clicks over a program line and adds a Breakpoint via the "Toggle Breakpoint" selection.

It is the recommended method for setting breakpoints in ZBI.

A maximum of 16 breakpoints can be set in an application.

This is an interactive command that takes effect as soon as it is received by the printer.

DELBREAK

This command allows you to remove existing breakpoints.

ZBI 2.01

Format

DELBREAK <A>

Parameters

A = the line number from which to remove the break. If 0 is specified, all break points will be removed. If the number specified is not a breakpoint, the command will have no effect.

Example

An example of the DELBREAK command:

```
10 LET A=5
20 PRINT A
ADDBREAK 20
DEBUG ON
TRACE ON
RUN
<TRACE> 10
<TRACE> A=5
<PROGRAM BREAK> ON LINE:20

RESTART
<TRACE> 20
5

DELBREAK 20
RUN
<TRACE> 10
<TRACE> A=5
<TRACE> 20
5
```

Comments

This command is available only when the DEBUG function has been activated. When DEBUG is on, BREAK halts processing, RUN starts the program from the beginning, and RESTART allows the program to continue where it left off.

This is the command used internally by ZBI-Developer when the user right-clicks over a program line and removes a Breakpoint via the "Toggle Breakpoint" selection.

A maximum of 16 breakpoints can be set in an application.

This is an interactive command that takes effect as soon as it is received by the printer.

ZPL

This command terminates and exits the ZBI environment.

Format
ZPL

Parameters
N/A

Example
An example of the ZPL command.

```
ZPL
ZBI TERMINATED
```

Comments
This is an interactive command that takes effect as soon as it is received by the printer.

Base Types and Expressions

There are two base types in the ZBI language. These types are Integers and Strings. Integers are whole numbers that contain no fractional part.

The range of values for integers is: -2,147,483,648 to +2,147,483,647

Strings are character arrays. The string length is only limited by the amount of memory in the system (version 2.0 and higher). Each character can have a value between 0 and 255 (version 2.0 and higher).

The use of control characters (0-31) may be difficult to debug based on the handling of control characters in different communications programs. In addition the ETX (3) will terminate a ZBI application when it is received on the console port. Use the CHR$ function when control characters must be placed into strings.

NOTE: In ZBI version 1.4 and lower, there was a string length limit of 255 characters.

This section is organized as follows:

- Variable Names
- Variable Declarations
- Constants
- Arrays
- Assignment
- Numeric Expressions
- String Concatenation (&)
- Sub-strings
- Boolean Expressions
- Combined Boolean Expressions

Variable Names

To distinguish strings from integers, string variable names must end in a $. Variable names must start with a letter and can include any sequence of letters, digits, and underscores. Function and command names must not be used as a variable name. Variable names are not case sensitive and are converted to uppercase by the interpreter.

A common mistake is to use a command or function name as a variable. To avoid using these reserved words, ZBI-Developer can be a useful resource. Reserved words are highlighted making it easier to spot this occurrence and thus, saving debugging time.

Valid variable names

I, J, K, VARNAME, VARSTR$, MYSTR$,MY_STR9$

Invalid Names

STR$ = Reserved word

ORD = Reserved word

VAL = Reserved word

W# = Invalid character (#)

9THSTR = Variable can not start with a number

Variable Declarations

ZBI will allow storage of up to 255 variables. If more variables are needed, consider using arrays to store data. The base array will take up one of the 255 variable slots, but it can be declared to allow for many indices.

Variables can be declared explicitly or implicitly. If a variable has not been used before, it will be declared when used. The default value for an integer will be zero and the default value of a string will be an empty string.

Explicit

```
DECLARE NUMERIC <variable_name>
DECLARE STRING <variable_name$>
```

If the variable existed before the DECLARE statement, it will be defaulted.

Implicit

```
LET <variable_name> = NUMERIC EXPRESSION
LET <variable_name$> = STRING EXPRESSION
```

The Interpreter is limited to 255 variables. If more variables are required, consider using arrays.

Constants

Integers are represented simply by numbers, such as 5, -10, 10000. Do not use commas in integer constants. Strings are enclosed by quotes. If a quote is required in the string, use double quotes, such as "Look here->""<-" would result in the string – Look here->"<-.

Arrays

An array is a collection of string or integer values used by a program. Array indices are accessed through parentheses. Array indices start at 1 and end at the length of an array (for example, MyArray(3) returns the value in the third location of the variable array). One- and two-dimensional arrays are allowed. Two-dimensional arrays are referenced with two indices in parentheses, separated by a comma.

Arrays must be allocated through the use of the DECLARE command. Arrays can be re-dimensioned by using DECLARE, however, this will replace the original array.

Array size is limited only by the size of the memory available.

Format

```
DECLARE STRING <ARRAYNAME$>(<SIZE>)
DECLARE STRING <ARRAYNAME$>(<ROWS>,<COLUMNS>)
DECLARE NUMERIC <ARRAYNAME>(<SIZE>)
DECLARE NUMERIC <ARRAYNAME>(<ROWS>,<COLUMNS>)
```

Parameters

```
<SIZE> = number of entries in a single dimension array
<ROWS> = number of rows in a two dimensional array
<COLUMNS> = number of columns in a two dimensional array
```

Example

An example of ARRAY code is:

```
10 DECLARE STRING INARRAY$(3)
20 FOR I = 1 TO 3
30 PRINT "Name "; I; ": ";
40 INPUT INARRAY$(I)
50 NEXT I
60 PRINT INARRAY$(1); ", "; INARRAY$(2); ", and "; INARRAY$(3);
70 PRINT " went to the park"
RUN
Name 1: Jim
Name 2: Jose
Name 3: Jack
Jim, Jose, and Jack went to the park
```

Comments

If you attempt to access an array outside of its allocated bounds, an error will occur.

Assignment

All lines must start with a command. In order to assign a value to a variable, use the LET command. Multiple variables can be placed before the =. The variable types must match the expression type.

The right side of the assignment is always calculated completely before the assignment is made. This allows a variable to be the target and source of the assignment.

When a value is assigned to a string variable with a sub-string qualifier, it replaces the value of the sub-string qualifier. The length of the value of the string variable may change as a result of this replacement.

Example

An ASSIGNMENT example:

```
10 LET A=5
20 LET B$="HELLO"
30 LET B$(5:5)=B$
```

LET

The LET command is used to assign value to a specific variable. The expression is evaluated and assigned to each variable in the variable list.

Format

`LET <variable> [,<variable>]* = <expression>`

The variable types must match the expression type or an error message will be displayed.

```
Error: Poorly formed expression.
```

When a value is assigned to a string variable with a sub-string qualifier, it replaces the value of the sub-string qualifier. The length of the value of the string variable may change as a result of this replacement.

Parameters

N/A

Example

This is an example of how to use the `LET` command:

```
10 LET A$= "1234"
15 LET A$(2:3)= "55" ! A$ NOW = 1554
20 LET A$(2:3)= "" ! A$ NOW = 14

10 LET A$= "1234"
15 LET A$(2:3)= A$(1:2) ! A$ NOW = 1124

10 LET A$= "1234"
15 LET A$(2:1)= "5" ! A$ NOW = 15234
```

Comments

This can be an interactive command that takes effect as soon as it is received by the printer, or a program command that is preceded by a line number.

Numeric Expressions

A base numerical expression can be either a constant, variable, or another numerical expression enclosed in parentheses. The five types used (addition, subtraction, multiplication, division, and exponentiation) are listed below. When evaluating an expression exceeding the maximum or minimum values at any point creates an undefined result. (maximum value: 2,147,487,647; minimum value: -2,147,483,648)

Floating point is not supported.

When using division, the number is always rounded down. For example, 5/2=2. Use `MOD` to determine the remainder.

Format

1. + (addition) Addition expressions use this format:

   <A>+<B>

   5+2 result = 7

   VAL ("25") +2 result =27

2. – (subtraction) Subtraction expressions use this format:

   <A>-<B>

   5-2 result = 3

   VAL ("25") -2 result =23

3. * (multiplication) Multiplication expressions use this format:

   <A>*<B>

   5*2 result = 10

   VAL ("25") *2 result =50

4. / (division) Division expressions use this format:

   <A>/<B>

   5/2 result = 2

   VAL ("25") /2 result =12

5. ^ (exponentiation) Exponentiation expressions use this format:

   <A>^<B>

   5^2 result = 25

   VAL ("25") ^2 result =625

Order of Precedence

In mathematics, the order of precedence describes the sequence that items in an expression are processed. All expressions have a predefined order of precedence.

The order of precedence is listed below:

1. Functions
2. Parenthetical Expressions ()
3. ^
4. * and /
5. + and -

The * and / have the same precedence, and the + and - have the same precedence. Items with the same order of precedence are processed from left to right.

For example, this expression 5+(8+2)/5 is processed as 8+2=10, followed by 10/5=2, then 5+2 to give a result of 7.

Functions and parenthetical expressions always have the highest order of precedence, meaning that they are processed first.

String Concatenation (&)

The basic string expression may be either a constant or a variable, and concatenation (&) is supported. Using the concatenation operator (&) adds the second string to the first string.

<A$> & <B$>

Example

This is an example of how to use the STRING CONCATENATION (&) command:

```
10 LET A$= "ZBI-"
20 LET B$= "Programming"
30 LET C$= A$ & B$
40 PRINT C$
RUN
ZBI-Programming
```

Sub-strings

Using a sub-string operator on a string allows a specific portion of the string to be accessed. This portion may be the target of an assignment operation or a reference to a portion of the string. To determine the coordinates of the string portion to be used, count the characters from the beginning to the end of the string, including spaces.

Format

LET <STRVAR$>(<A>:<B>)=<C$>

LET <C$> = <STRVAR$>(<A>:<B>)

Parameters

<A> = the position of the first character in the desired string

<B> = the position of the last character in the desired string.

<STRVAR$> = base string variable

If the A parameter is less than 1, it is automatically assigned a value of 1. Because the string is calculated starting with 1, the A parameter cannot be less than 1.

If B is greater than the length of the string, it is replaced with the length of the string.

If A is greater than B, a NULL string (""), which points to the location of the smaller of A or the end of the string, is returned. This is used when adding a string in the middle of another string without removing a character.

Example

This is an example of a sub-string reference:

```
LET A$="Zebra Quality Printers"
LET B$=A$(1:13)
PRINT B$
Zebra Quality
```

This is an example of a sub-string assignment.

```
LET A$= "1234"
LET A$(2:3)= "55" ! A$ NOW = 1554
LET A$(2:3)= "" ! A$ NOW = 14

LET A$= "1234"
LET A$(2:3)= A$(1:2) ! A$ NOW = 1124

LET A$= "1234"
LET A$(2:1)= "5" ! A$ NOW = 15234
```

The best way to think of assignment to a sub-string is as follows: an assignment is like selecting a word, and pasting over the selection with the new string.

Boolean Expressions

A Boolean expression holds 0 (zero) as false and non-zero as true.

Formats

```
<STRING EXPRESSION> <BOOLEAN COMPARE> <STRING EXPRESSION>
<NUMERIC EXPRESSION> <BOOLEAN COMPARE> <NUMERIC EXPRESSION>
NOT(<BOOLEAN EXPRESSION>)
```

Parameters

`<STRING EXPRESSION>` = a string variable, string constant or any combination with concatenation

`<NUMERIC EXPRESSION>` = any mathematical operation

Comments

A numeric expression cannot be compared to a string expression.

Numeric expressions can substitute a Boolean expression where a value of 0 (zero) represents false and a non-zero value represents true.

Base Boolean expressions:

Table 15 < (less than)

Expression	Result
1< 2	true
2<2	false
2<1	false

Table 16 <= (less than or equal to)

Expression	Result
1<=2	true
2<=2	true
2<=1	false

Table 17 > (greater than)

Expression	Result
1> 2	false
2>2	false
2>1	true

Table 18 >= (greater than or equal to)

Expression	Result
1>=2	false
2>=2	true
2>=1	true

Table 19 = (equal to)

Expression	Result
1=2	false
2=2	true
"A"="AA"	false
"A"="A"	true

Table 20 <> (not equal to)

Expression	Result
1<>2	true

Combined Boolean Expressions

AND, OR, and NOT can be used in conjunction with base Boolean expressions to recreate expanded Boolean expressions.

Table 21 NOT — Negate the target expression.

Expression	Result
NOT 1=2	true
NOT 1=1	false

Table 22 AND — Both expressions must be true for a true result.

Expression	Result
1=2 AND 1=2	false
2=2 AND 1=2	false
1=2 AND 2=2	false
2=2 AND 2=2	true

Table 23 OR — If either expression is true, the result will be true.

Expression	Result
1=2 OR 1=2	false
1=2 OR 2=2	true
2=2 OR 1=2	true
2=2 OR 2=2	true

Order of Precedence

The order of precedence is listed below:

1. Expressions and Functions
2. Parenthetical expressions ()
3. <, <=, <>, =, =>, >
4. NOT, AND, OR

Control and Flow

This section outlines the commands to conditionally execute code and control the flow of the program. Here is a quick list of these commands:

IF Statements
Executes or skips a sequence of statements, depending on the value of a Boolean expression.

DO Loops
Repeats instructions based on the results of a comparison.

FOR Loops
A control flow statement which allows code to be executed iteratively.

GOTO/GOSUB
Causes an unconditional jump or transfer of control from one point in a program to another.

SUB
Allows you to "substitute" names instead of actual line numbers as the target of `GOSUB`s and `GOTO`s.

EXIT
Used to exit the `DO` and `FOR` loops.

END
Terminates any program currently running.

IF Statements

If the value of the `<Boolean expression>` in an `IF` statement is true and a program line follows the keyword `THEN`, this program line is executed. If the value of the Boolean expression is false and a program line follows the keyword `ELSE`, this program line is executed. If `ELSE` is not present, then execution continues in sequence, with the line following the `END IF` statement.

Nesting of blocks is permitted, subject to the same nesting constraints as `DO-LOOP`s (no overlapping blocks).

`ELSE IF` statements are treated as an `ELSE` line followed by an `IF` line, with the exception that the `ELSE IF` shares the `END IF` line of the original `IF` statement.

Format

```
IF <Boolean expression> THEN
~~BODY~~
[ELSE IF <Boolean expression> THEN
~~BODY~~]*
[ELSE
~~BODY~~]
END IF
```

Parameters
N/A

Example

This is an example of how to use the IF statement command:

```
10 IF A$="0" TyHEN
20 PRINT "ZBI IS FUN"
30 ELSE IF A$="1" THEN
40 PRINT "ZBI IS EASY"
50 ELSE IF TIME=1 THEN
60 PRINT "It is one second past midnight"
70 ELSE
80 PRINT "X=0"
90 END IF
```

DO Loops

Processing of the loop is controlled by a <WHILE/UNTIL> expression located on the DO or LOOP line.

Processing a WHILE statement is the same on either the DO or LOOP lines. The Boolean expression is evaluated and if the statement is true, the LOOP continues at the line after the DO statement. Otherwise, the line after the corresponding LOOP is the next line to be processed.

Processing an UNTIL statement is the same on either the DO or LOOP lines. The Boolean expression is evaluated and if the statement is false, the LOOP continues at the line after the DO statement. Otherwise, the line after the corresponding LOOP is the next to be processed.

If <WHILE/UNTIL> is on the LOOP line, the BODY of the loop is executed before the Boolean expression is evaluated.

If neither the DO or LOOP line has a <WHILE/UNTIL> statement, the loop continues indefinitely.

Some notes about DO-LOOPs:

- can be nested
- cannot overlap
- have two formats

Format

```
DO [<WHILE/UNTIL> <Boolean expression>]

~~BODY~~

LOOP [<WHILE/UNTIL> <Boolean expression>]
```

Example

This is an example of how to use the DO-LOOP command with the conditional on the DO line:

```
10 DO WHILE A$="70"
20 INPUT A$
30 LOOP
```

Example

This is an example of how to use the `DO UNTIL LOOP` command with conditional on the LOOP line:

```
10 DO
20 INPUT A$
30 LOOP UNTIL A$="EXIT"
```

Comments

This is a program command that is preceded by a line number.

FOR Loops

`FOR` loops are an easy way to iterate through a range of values and run a body of code for each value iterated.

Format

```
FOR <I> = <A> TO <B> [STEP <C>]

~~BODY~~

NEXT <I>
```

Parameters

<I> = indicates a numeric variable is used. <I> increments each time through the FOR#LOOP.

<A> = the value assigned to <I> the first time through the loop

<B> = the last value through the loop

<C> = (Optional) the amount <I> increments each time through the loop

Values of `I` for the following situations:

Statement	Result
`FOR I=1 TO 10`	`{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}`
`FOR I=10 TO 1`	`{10, 9, 8, 7, 6, 5, 4, 3, 2, 1}`
`FOR I=1 TO 10 STEP 2`	`{1, 3, 5, 7, 9}`
`FOR I=10 TO 1 STEP 2`	`{10, 8, 6, 4, 2}`
`FOR I=10 TO 1 STEP 2`	`{ } FOR LOOP skipped`

Example

This is an example of how to use the FOR LOOP command:

```
10 FOR X=1 TO 10 STEP 1
20 PRINT X; ":ZBI IS FUN"
30 NEXT X
```

Comments

FOR loops can be nested but cannot overlap. Variables cannot be reused by the nested loops.

GOTO/GOSUB

GOSUB is followed by a line number. The program will attempt to process the line the GOSUB command points to rather than the next line of the program. Upon executing the GOSUB statement, the interpreter continues running at the line number specified following GOSUB. If the line number referenced does not exist, an error will occur.

Before executing the next line, the GOSUB command stores the line number of the GOSUB line. When the RETURN statement is called, the program moves back to the next line following the GOSUB.

Executing a RETURN statement without a corresponding GOSUB statement causes an error.

GOSUB statements can be nested.

GOTO works the same way as GOSUB except that no return address will be stored.

Format

```
GOSUB <A>

RETURN

GOTO <A>
```

Parameters

<A> = the program location executed immediately after the GOTO or GOSUB.

Example

This is an example of how to use the GOSUB command:

```
10 PRINT "Call Subroutine"
20 GOSUB 1000
30 PRINT "Returned from Subroutine"
40 END
1000 PRINT "In Subroutine"
1010 RETURN
```

Example

This is an example of how to use the GOTO command:

```
10 PRINT "Prepare to Jump!"
20 GOTO 1000
30 PRINT "Jump Missed..."
1000 PRINT "Jump Successful"
1010 END
```

Comments

These are program commands and must be preceded by line numbers.

SUB

This command allows you to use names instead of actual line numbers as the target of GOSUBs and GOTOs. AUTONUM can be used at the beginning of a file and there is no need to compute the line number where the jump will go.

Format

```
10 SUB <A>
```

Parameters

<A> = the integer variable to use as a target for the GOTO/GOSUB

Example

This is an example of how to use the SUB command:

```
AUTONUM 1,1
GOSUB INITCOMM
DO
GOSUB GETINPUT
GOSUB PROCESSINPUT
LOOP
SUB INITCOMM
OPEN #1:NAME "SER"
RETURN
SUB GETINPUT
INPUT #1: A$
RETURN
SUB PROCESSINPUT
PRINT A$
RETURN
```

Comments

<A> is a numeric variable. If this variable is changed in the program, any GOSUB/GOTO to this variable may fail.

EXIT

This command is used to exit the DO and FOR loops.

Format

```
EXIT DO

EXIT FOR
```

Parameters

The specified loop type is exited. For the DO command, the program will continue execution on the line following the next LOOP. Likewise for the FOR command, the program will continue on the line after the next NEXT command.

Example

N/A

Comments

This is a program command that is preceded by a line number. To be explicit and reduce errors, it is recommended to use GOTO instead of EXIT.

END

The END command terminates any program currently running. When the END command is received, the interpreter returns to interpreting commands (>).

Format

END

Parameters

N/A

Example

This is an example of how to use the END command:

```
10 PRINT "THIS PROGRAM WILL TERMINATE"
20 PRINT "WHEN THE END COMMAND IS RECEIVED"
30 END
40 PRINT "THIS SHOULD NOT PRINT"
RUN
THIS PROGRAM WILL TERMINATE
WHEN THE END COMMAND IS RECEIVED
```

Comments

This is a program command and is preceded by a line number.

Input and Output

This section oulines how to communicate with physical ports, internal ports, and the network.

ZBI allows access to the physical and network connections in the printer. Most ports are, by default, connected to the ZPL processor. When a port is opened in ZBI, the port will be disconnected from ZPL and connected into the interpreter. Depending on the type of connection, there are two methods you may use to start the connection. For the static connections, the OPEN command should be used. These are the connections that you open when starting your program and leave open for the duration of your program. For dynamic connections, servers and clients are set up following the "Sockets" model. On servers, the actual connections are started upon successful calls to ACCEPT. Below are the available connections that can be made and the preferred accessors.

Available Ports

Port/Connection	ZBI Name	Preferred Access Commands/Functions
Serial	"SER"	OPEN, CLOSE
Parallel	"PAR"	OPEN, CLOSE
USB	"USB"	OPEN, CLOSE
ZPL parser	"ZPL"	OPEN, CLOSE
TCP Server	"TCP", "TCPX"	SERVERSOCKET, SERVERCLOSE, ACCEPT, CLOSE
TCP Client	"TCP"	CLIENTSOCKET, CLOSE
UDP Server	"UDP"	SERVERSOCKET, SERVERCLOSE, ACCEPT, CLOSE
UDP Client	"UDP"	CLIENTSOCKET, CLOSE
Email Sender	"EML"	OPEN, CLOSE
Bluetooth	"BLU"	OPEN, CLOSE

NOTE:

TCPx will not work on PS2 or PS100 print servers.

Creating Connections

Here is a list of the commands in this section:

OPEN
Opens a port for transmitting and receiving data.

CLOSE
Closes specific ports that are in use.

DATAREADY
Determines if there is data received on a specified port.

SERVERSOCKET
Opens a listening socket for incoming UDP packets or TCP connections.

SERVERCLOSE
Closes a listening server socket.

CLIENTSOCKET
Creates an outgoing TCP connection or sets up UDP transmissions.

ACCEPT

Accepts incoming TCP or UDP connections and assigns a channel for the connection.

OPEN

This command is used to open a port for transmitting and receiving data.

Format

`OPEN #<CHANNEL>: NAME <PORT$>`

Parameters

`<CHANNEL>` = a number to use as a handle to the port for all future communications

Values

0 to 9

Default

a port must be specified

`<PORT$> =` port name to open.

Example

This is an example of how to use the OPEN command:

```
10 OPEN #1: NAME "ZPL"
```

The port being opened no longer allows data to pass directly into its buffer, it disconnects, and the interpreter now controls the data flow.

Data already in the buffer stays in the buffer.

Comments

This can be an interactive command that takes effect as soon as it is received by the printer, or a program command that is preceded by a line number.

CLOSE

This command is implemented to close specific ports that are in use. If a port is open on a channel and the CLOSE command is entered, the port closes and returns to communicating with the ZPL buffer.

Format

CLOSE #<A>

CLOSE ALL

Parameters

<A> = Numeric value of port to close

Values

0 through 9

All = closes all open ports and network connections

NOTE: CLOSE ALL will close the console.

Example

This example shows the closing of channel 1:This example shows the closing of channel 1:

```
10 CLOSE #1
```

Comments

This can be an interactive command that takes effect as soon as it is received by the printer, or a program command that is preceded by a line number.

DATAREADY

This function is used to determine if there is data received on a specified port.

Format

`DATAREADY (A)`

Parameters

A = the port to check

Returns

1 if there is data, 0 if there is no data.

Example

This is an example of how to check if there is a data on a port:

```
10 PRINT DATAREADY(0)
RUN
```

The result, assuming no data is waiting, is:

```
0
```

Comments

If this command follows the `INPUT` command, it may return 1 if the line received was ended with a CRLF. In this case, `INBYTE` can be used to take the LF out of the buffer.

SERVERSOCKET

This function opens a listening socket for incoming UDP packets or TCP connections. It must be used in conjunction with the ACCEPT function.

Format

SERVERSOCKET (TYPE$,PORT)

Parameters

TYPE$ = listens for any of the following communication protocols:

- "TCP" = TCP – PORT parameter is ignored. The current port will be used.
- "TCPX" = TCP – any open port
- "UDP" = UDP – any open port

Returns

NUMERIC = returns the handle of the server upon success.

Example

See the examples for TCP Server on page 504 and UDP Server on page 505.

Comments

When using TCPX, care needs to be taken not to use a port that is already open on the printer. No error message will be returned until the ACCEPT function is called.

SERVERCLOSE

This function closes a listening server socket created by SERVERSOCKET.

Format

SERVERCLOSE(SOCKET)

Parameters

SOCKET = the socket handle returned from a successful SERVERSOCKET invocation.

Returns

Returns a 0 if the socket was already closed or a 1 if the socket was closed successfully.

Example

This example shows how to close a listening server socket.

```
10 LET SERVER_HANDLE = SERVERSOCKET("TCPX", 19100)
20 LET SCERR = SERVERCLOSE(SERVER_HANDLE)
```

CLIENTSOCKET

This function creates an outgoing TCP connection or sets up UDP transmissions. Once set up for UDP, packets can be sent by printing to the socket. Packets are sent when the size limit is met or a EOT character is written.

Format

```
CLIENTSOCKET (TYPE$, IPADDR$, PORT)
```

Parameters

- TYPE$ = set to "UDP" or "TCP".
- IPADDR$ = connects to this address.
- PORT = connects to this IP port.

Returns

The port number assigned to the connection.

Example

See the examples for TCP Server on page 504 and UDP Server on page 505.

Comments

Multiple communications connections can be made up to the maximum of 10. Each protocol may have a different limit based on the support of the print server used. Test the worst case situation based on your application's needs or use ONERROR to recover from failed connection attempts.

ACCEPT

This function will accept incoming TCP or UDP connections and assign a channel for the connection. SERVERSOCKET must be used to set up the listening socket before ACCEPT can be used.

Format

```
ACCEPT (SERVER, CLIENT_INFO$)
```

Parameters

- SERVER = the handle returned by the SERVERSOCKET call.
- CLIENT_INFO$ = string variable will have the connecting client's IP address and port separated by a space when using UDP.

Returns

The channel number to use to communicate with the client.

Example

See the examples for TCP Server on page 504 and UDP Server on page 505.

Comments

It is best to poll this function at regular intervals. When there is no connection waiting, this function will trigger an error. Follow this function with the ON ERROR command to divert to a section of code that handles an unsuccessful connection.

ACCEPT can be called before closing a previous connection. This allows for processing multiple incoming streams of data. There are limits on the number of simultaneous incoming connections based on the print server model on the printer.

Connection closure can be detected when any input or output command to the port triggers an error. These commands should be followed by an ON ERROR statement to send the program into a recovery state and to shutdown the connection cleanly.

Reading and Writing

This manual has detailed various functions to read and write to all of the ports. The following section gives an overview of the commands, functions, and when each should be used.

To start, it is important to understand the term "blocking". In communications code, a function or command is "blocking" if it waits for all of the requested data to be received before it returns.

INPUT (blocking)
Reads one line into each string specified.

PRINT (blocking)
Simple method to write specified expressions out.

OUTBYTE (blocking)
Writes one byte out.

INBYTE (blocking)
Reads in one byte.

READ (non-blocking)
Reads in all available data up to the maximum amount specified.

WRITE (non-blocking)
Writes out as much data as possible up to a maximum specified amount.

SEARCHTO$ (blocking)
Reads in data (does not keep) until a search parameter is found. Non-matching data can be redirected to another port.

INPUT

If the variable is numeric and the value entered cannot be converted to a number, it writes as 0. This operation scans the data from left to right, shifting any number into the variable. It ignores any non-numeric character except the return character, which terminates the input, or Ctrl-C (^C) which terminates the program. The variable can be in string or numeric form.

Format

`INPUT [<CHANNEL>:] <A$> [,<B$>]*`

`INPUT [<CHANNEL>:] <A>[,<B>]*`

If the `[<channel>:]` is omitted, the default port, 0, will be used.

Parameters

`<CHANNEL>` = read data from this port. Default = 0.

`<A,B,...,N>` = variables to write.

When using multiple variables as targets, a corresponding number of lines are read. String and numeric variables can be intermixed.

Example

This is an example of how to use the INPUT command:

```
10 OPEN #1: NAME "ZPL"
20 PRINT #1: "~HS"
30 FOR I = 1 TO 3
40 INPUT #1: A$
50 PRINT A$
60 NEXT I
```

In this example, a host status prints to the console after submitting the host status request ~HS to the ZPL port. The Input/Output command of the ZBI interpreter is limited to the communications ports. File I/O is not supported.

`INPUT` ends processing a line with a CR or LF. This leads to a tricky situation. There are many ways different systems end a line: CR, CRLF, LF. If the ZBI program only uses `INPUT`, the next execution of the `INPUT` command will remove the extra LF or CR, in case of LFCR. However, if the program instead uses `INBYTE`, `DATAREADY` or the other commands, the extra LF will show up on the port. Here's a simple workaround to explicitly look for the CRLF that is in use:

`SEARCHTO(<PORT>,CHR$(13)&CHR$(10),<INSTRING$>)`

NOTE: Note: The `INPUT` command does not accept control characters or the delete character. If these characters need to be processed, use the READ command.

Comments

This can be an interactive command that takes effect as soon as it is received by the printer, or a program command that is preceded by a line number.

If an invalid port is specified, `Error: Invalid port` is returned.

Example

This shows the input command reading in multiple lines.

```
10 INPUT A$,B,C,D$,E$
```

Five lines would be read in: 3 strings and 2 numbers.

PRINT

This command sends data to the printer to be printed.

Format

```
PRINT [CHANNEL:] <expression> [,or; <expression>]* [;]
```

Parameters

<CHANNEL> = write data to this port

<expression> = the value to write

The expression can be either a string or a numeric expression.

- Using a , to separate expressions adds a space between them.
- Using a ; to separate expressions does not put a space between them.
- Using a ; at the end of a line ends the print statement without adding a new line (CR/LF).

Example

This is an example of how to use the PRINT command:

```
10 LET A$ = "This is an example"
20 LET B$ = "of the PRINT Command."
30 PRINT A$, B$ ! adds a space between expressions
40 PRINT A$; B$ ! no space added
RUN
```

The result is:

- This is an example of the PRINT Command.
- This is an exampleof the PRINT Command.

Comments

This can be an interactive command that takes effect as soon as it is received by the printer, or a program command that is preceded by a line number.

OUTBYTE

This command outputs a byte to a port.

Format

`OUTBYTE [<CHANNEL>:] <A>`

`OUTBYTE [<CHANNEL>:] <A$>`

Parameters

<CHANNEL> = sends the byte to this port. Default = 0.

<A> = This is a numeric expression.

Values

0 through 255. If it is not within that range, it is truncated.

<A$> = This is the string expression. The first character is used. In the case of a NULL string, 0 is sent.

Example

This is an example of how to use the OUTBYTE command:

```
LET A$="Hello"
OUTBYTE A$
```

This would only print the H character to the console.

```
OUTBYTE 4
```

This would print the control character EOT to the console. See an ASCII table for a list of the control characters.

Comments

This can be an interactive command that takes effect as soon as it is received by the printer, or a program command that is preceded by a line number.

INBYTE

This command forces the interpreter to pause until data is available. Use the DATAREADY function to determine if there is data on the port.

Format

`INBYTE [<CHANNEL>:] <A>`

`INBYTE [<CHANNEL>:] <A$>`

<A> = integer value is set to the byte received.

Parameters

<CHANNEL> = reads from this port. Default = 0.

<A$> = A single byte string is created with the byte received.The first character is used. In the case of a NULL string, 0 is sent.

Example

This is an example of how to use the INBYTE to create an echo program:

```
10 INBYTE A$ !Takes one byte (char) from port #0
20 PRINT A$ !Prints the character to the console
30 GOTO 10
```

In this example, the interpreter pauses until the data is entered, then continues processing. This command enters all bytes in a string or integer, including control codes.

Comments

INBYTE will block until a byte is received on the specified port. This can be an interactive command that takes effect as soon as it is received by the printer, or a program command that is preceded by a line number.

READ

This is a non-blocking input function. It will read in all of the bytes available on the specified port.

Format

READ (<CHANNEL>, <A>, <MAXBYTES>)

Parameters

<CHANNEL> = reads from this port. Default = 0.

<A$> = the string where the data will be placed

<MAXBYTES> = the maximum number of bytes to read

Returns

The number of bytes read.

Example

This is an example of the READ command:

```
1 CLOSE ALL
2 LET INPORT = CLIENTSOCKET("TCP","192.168.0.1",9100)
3 ON ERROR GOTO RECOVERY
4 LET WATERMARK = 5000
5 DO WHILE 1
6 IF LEN(DATA$) < WATERMARK THEN
7 LET BYTESREAD = READ(INPORT,DATA$,500)
8 ON ERROR GOTO RECOVERY
9 END IF
10 IF (LEN(DATA$) > 0) THEN
11 LET BYTES_WRITTEN = WRITE(INPORT,DATA$,LEN(DATA$))
12 ON ERROR GOTO RECOVERY
13 LET DATA$(1,BYTES_WRITTEN) = ""
14 END IF
15 IF BYTESREAD = 0 AND BYTESWRITTEN = 0 THEN
16 SLEEP 1 ! DON'T BOMBARD IF IDLE
17 END IF
18 LOOP
19 SUB RECOVERY
20 CLOSE #INPORT
```

WRITE

This is a non-blocking output function. It will write as many bytes as the output buffer can hold.

Format

```
WRITE (<CHANNEL>, <A>, <BYTES>)
```

Parameters

<CHANNEL> = reads from this port. Default = 0.

<A$> = the string to write out.

<MAXBYTES> = The number of bytes to write

Returns

The number of bytes written.

Example

This is an example of `WRITE` command:

```
1 CLOSE ALL
2 LET INPORT = CLIENTSOCKET("TCP","192.168.0.1",9100)
3 ON ERROR GOTO RECOVERY
4 LET WATERMARK = 5000
5 DO WHILE 1
6 IF LEN(DATA$) < WATERMARK THEN
7 LET BYTESREAD = READ(INPORT,DATA$,500)
8 ON ERROR GOTO RECOVERY
9 END IF
10 IF (LEN(DATA$) > 0) THEN
11 LET BYTES_WRITTEN = WRITE(INPORT,DATA$,LEN(DATA$))
12 ON ERROR GOTO RECOVERY
13 LET DATA$(1,BYTES_WRITTEN) = ""
14 END IF
15 IF BYTESREAD = 0 AND BYTESWRITTEN = 0 THEN
16 SLEEP 1 ! DON'T BOMBARD IF IDLE
17 END IF
18 LOOP
19 SUB RECOVERY
20 CLOSE #INPORT
```

SEARCHTO$

This function performs a search until a specified string is found. The string the search yields is displayed.

Format

```
SEARCHTO$(A,B$)
SEARCHTO$(A,B$,C)
SEARCHTO$(A$,B$)
SEARCHTO$(A$,B$,C$)
```

Parameters

- A = port number (0 to 9) to which requested data is sent
- A$ = string to search for B$
- B$ = string variable or string array. If B$ is an array, this command searches for all non-null strings in the B$ array.
- C = a port in which the input is directed until B$ is found
- C$ = a string in which the characters in A$ are directed until B$ is found

Returns

The string found.

Example

This example shows how to use SEARCHTO to find a string on a port:

```
10 OPEN #1: NAME "SER"
20 LET A$ = SEARCHTO$(1,"^XA")
30 PRINT "FOUND:", A$
```

Example

This example shows how to search for an array of strings:

```
10 OPEN #1: NAME "SER"
20 DECLARE STRING FIND$(3)
30 LET FIND$(1) = "ONE"
40 LET FIND$(2) = "TWO"
50 LET FIND$(3) = "THREE"
60 LET A$ = SEARCHTO$(1,FIND$)
70 PRINT "FOUND:", A$
```

Example

This example shows unused data routed to a port.

```
10 OPEN #1: NAME "PAR"
20 OPEN #2: NAME "SER"
30 DECLARE STRING FIND$(3)
40 LET FIND$(1) = "ONE"
50 LET FIND$(2) = "TWO"
60 LET FIND$(3) = "THREE"
70 LET A$ = SEARCHTO$(1,FIND$,2)
80 PRINT "FOUND:", A$
```

Example

This example shows how to use SEARCHTO to find a string within a string and direct the unused part of the string to another string:

```
10 LET A$ = "The faster you
go, the shorter you are - Einstein"
20 LET B$ = SEARCHTO$(A$,"you", C$)
30 PRINT "FOUND:", B$
40 PRINT "DISCARDED:", C$
```

Comments

SEARCHTO will block (wait) until the search string is found. If you want to be able to run other code while doing something similar, consider using READ with POS.

When using SEARCHTO with ports, it will block (wait) until the search string is found. If you want to be able to run other code while doing something similar, consider using READ to place data into a string. That string can be passed to SEARCHTO for processing.

Port Usage Examples

Before diving into the syntax of all the commands, let's look at some simple applications using the different features of the communications systems in ZBI.

Physical Ports (Serial, Parallel, USB, Bluetooth®)

Though the types of devices interacting with the printer's ports may vary greatly, internal to the printer, the ports are all handled in the same way. These ports are opened with the ZBI `OPEN` command and closed with the ZBI `CLOSE` command. When one of these ports is opened, it is disconnected from the ZPL parser and any data in the buffer will be redirected to the ZBI environment.

Example

In the following example, `"SER"` could be replaced by `"PAR"`, `"USB"`, or `"BLU"` depending on the application.

```
10 CLOSE ALL
20 LET INPORT = 1
25 SLEEP 1
30 OPEN #INPORT: NAME "SER"
35 ON ERROR GOTO 25
40 PRINT #INPORT: "Enter your name:";
50 INPUT #INPORT: YOURNAME$
55 ON ERROR GOTO 70
60 PRINT #INPORT: "You entered: "; YOURNAME$
70 CLOSE #INPORT
```

ZPL Parser

To make a ZBI program print, it is necessary to create a connection from the program to the ZPL parser on the printer. The connection will function in the same way as a connection to a physical port, except that the connection will not automatically terminate. The ZPL parser in the printer can handle many incoming connections simultaneously. For example, a ZBI program could take control of the serial port and send label formats to the ZPL parser, while the parallel port (unopened by ZBI) could also be used to send label formats directly into the parser.

NOTE: The ZPL parser will lock onto one port once a format is started (via the ^XA command). So, in some cases, is it desirable to start and stop your communications to ZPL in one continuous sequence.

Another use of ZBI is to check printer status, while another application prints to another port.

Example

Here is how that can be done:

```
10 OPEN #1: NAME "ZPL"
20 PRINT #1: "~HS"
30 FOR I = 1 TO 3
40 INPUT #1: A$
```

```
50 PRINT A$
60 NEXT I
```

TCP Client

There are two methods for making a TCP connection to another server. The first method uses the OPEN command while the second method uses the CLIENTSOCKET method.

CLIENTSOCKET is the preferred method.

Example

The following example demonstrates this method:

```
10 CLOSE ALL
20 LET INPORT = CLIENTSOCKET("TCP","192.168.0.1",9100)
40 LET OUTSTR$ = "REQUESTING SERVER NAME";
50 DO WHILE (LEN(OUTSTR$) > 0)
60 LET BYTES_WRITTEN = WRITE(INPORT,OUTSTR$,LEN(OUTSTR$))
70 ON ERROR GOTO RECOVERY
80 LET OUTSTR$ = OUTSTR$(1+BYTES_WRITTEN:LEN(OUTSTR$))
90 LOOP
100 INPUT #INPORT: YOURNAME$
110 PRINT #INPORT: "Server returned: "; YOURNAME$
120 CLOSE #INPORT
130 SUB RECOVERY
140 END
```

TCP Server

Setting up a listening server in the printer can be accomplished with the SERVERSOCKET function. To connect to incoming TCP sessions, use the ACCEPT function.

When starting the application, call SERVERSOCKET. This function will create a handle for this listening server. Check for incoming connections at regular intervals with the ACCEPT function. If there are no pending sessions, the ACCEPT function will return with an error. Handle the error using the ON ERROR command and continue looking for other sessions later.

Depending on how the program is set up, it is possible to handle one or more sessions at a time. If the program is configured to allow only one session, the other connections will remain pending until they are shut down by the requesting client or the ZBI program connects them.

Example

Here is an example of the SERVERSOCKET and ACCEPT commands:

```
10 CLOSE ALL
20 LET SERVER_HANDLE = SERVERSOCKET("TCPX",19100)
30 REM There are no connections yet we are just listening for them
40 REM Lets loop until we get a connection
50 SLEEP 1
60 LET INPORT = ACCEPT(SERVER_HANDLE,CLIENT_INFO$)
70 ON ERROR GOTO 50
```

```
80 PRINT #INPORT: "You have successfully connected!"
90 PRINT #INPORT: "Login:";
100 INPUT #INPORT: LOGIN$
110 PRINT #INPORT: "Password:";
120 INPUT #INPORT: PASSWORD$
130 REM We will not be nice and reject the connection
130 PRINT #INPORT: "Login failed"
140 CLOSE #INPORT
150 GOTO 60 ! Go look for the next connection
160 END
```

UDP Client

There are also two methods for making a UDP connection to another server. The first method uses the OPEN command, while the second method uses the CLIENTSOCKET method. UDP is a one way communication medium, thus, you can only use output commands. Because UDP is connectionless, the output will be queued up until an EOT character is written or the maximum packet size is exceeded. Once the EOT character is written, the packet is formatted and sent.

With UDP, it is important to be careful about understanding what the network being used will support.

In many cases, there will be a limit to the size of the packet that can be used, typically between 1000 and 1500 bytes, but some networks cut this down into the 500 to 600 byte range. To be safe, keep your packets less than 500 bytes.

UDP does not guarantee transmission. See UDP specifications for more details.

Example

Since CLIENTSOCKET is the preferred method, an example is shown below.

```
10 CLOSE ALL
20 LET INPORT = CLIENTSOCKET("UDP","192.168.0.1",22222)
30 LET EOT$ = CHR$(4)
40 PRINT #INPORT: "Packet #"; I; EOT$;
50 LET I = I + 1
60 SLEEP 1
70 GOTO 40
```

UDP Server

Setting up a listening server in the printer can be accomplished with the SERVERSOCKET function. Then, to connect to incoming UDP packets, use the function ACCEPT. When starting your application, call SERVERSOCKET. This function will create a handle for this listening server.

Check for incoming packets at a regular interval with the ACCEPT function. If there are no pending sessions, the ACCEPT function will return with an error. Just handle the error using the ON ERROR command and continue looking for other sessions later. You will need to call ACCEPT for each incoming packet. When the accept is successful, all of the data will be available. Call READ with a MAX string size of 2000 and you will have the whole packet in your string. Close the port and wait for the next packet. You can only read in data using a UDP server.

Example

Here is an example of how to set up to receive UDP messages:

```
10 CLOSE ALL
20 LET ZPLPORT = 1
35 OPEN #ZPLPORT: NAME "ZPL"
40 LET SERVER_HANDLE = SERVERSOCKET("UDP",33333)
50 REM There are no connections yet: listening
60 REM Let's loop until we get a connection
70 SLEEP 1
80 LET INPORT = ACCEPT(SERVER_HANDLE,CLIENT_INFO$)
90 IF INPORT = -1 THEN
92 GOTO 70
94 END IF
100 LET PACKET_SIZE = READ(INPORT,PACKET$,2000)
110 PRINT #ZPLPORT: "^XA^FO100,100^A0N,40,40^FDPACKET FROM:";
115 PRINT #ZPLPORT: CLIENT_INFO$; "^FS"
120 PRINT #ZPLPORT: "^FO100,150^A0N,40,40^FDPACKET SIZE:";
125 PRINT #ZPLPORT: PACKET_SIZE; "^FS"
130 PRINT #ZPLPORT: "^FO100,200^A0N,40,40^FDPACKET DATA:";
135 PRINT #ZPLPORT: PACKET$; "^FS^XZ"
140 CLOSE #INPORT
150 GOTO 60 ! go look for the next connection
160 END
```

E-mail

ZBI can be used to enhance the printer's ability to send status via e-mail messages. The process is simple: open the email port "EML", send the recipient list, send the header, and send the body of the message.

The printer can only process a limited number of outgoing email messages at one time. For this reason, error handling should be used when opening the connection to wait for the printer to be ready to send the message. The EOT character is important for delimiting sections of the email message. If it is left out, the message will not be sent properly.

Before the following code will work, the email settings for the print server must be set up. Consult the print server manual to learn how to configure the unit.

Example

Here is an example of how to send e-mails:

```
1 REM EOT$ this is used to denote end of transmission
5 LET EOT$ = CHR$(4)
1 REM Open a connection to the e-mail port and if it errors
1 REM try again until complete
10 OPEN #1: NAME "EML"
15 ON ERROR GOTO 10
1 REM Specify address to send message to then end signal end
1 REM of recipients with EOT$
1 REM To send to multiple addressees separate addressees by
1 REM space
```

```
20 PRINT #1: "youraddress@yourdomain.com";EOT$;
1 REM Fill in the message information
30 PRINT #1: "From: HAL"
40 PRINT #1: "To: Dave"
50 PRINT #1: "Subject: A message from HAL"
60 PRINT #1: ""
70 PRINT #1: "Dave, I am sorry I can not let you do that."
80 PRINT #1: i
1 REM Terminate message
90 PRINT #1: "";EOT$
1 REM You must close the port, each open port is only good
1 REM for sending one message
100 CLOSE #1
```

File System

This section shows how programs and formats can be saved and recalled. Here's a quick list of these commands:

STORE
Saves the program currently in memory as the specified file name.

LOAD
Transfers a program file previously stored in the printer's memory and opens it in the ZBI Program Memory.

DIR
With no filter included, prompts the printer to list all of the ZBI programs residing in all printer memory locations.

DELETE
Removes a specified file from the printer's memory.

Runtime Access

The following example is a method to store runtime data in the printer memory. The file system in the printer is limited to writing one file at a time. Since only one component of the printer can have write access to the file system, the ZPL parser is the component with this access. For ZBI to use the ZPL parser as a gateway into printer memory, the ZPL comment command (^FX) is used.

Example

```
AUTONUM 1,1
REM ******* TEST FOR SUBROUTINES *********************
LET ZPLPORT = 1 OPEN #ZPLPORT: NAME "ZPL"
LET SIZE = 5
LET FILENAME$ = "R:TESTSYS.ZPL"
DECLARE STRING DATAIN$(SIZE)
LET DATAIN$(1) = "ONE"
LET DATAIN$(2) = "TWO"
LET DATAIN$(3) = "THREE"
LET DATAIN$(4) = "FOUR"
LET DATAIN$(5) = "FIVE"
GOSUB STOREDATA
GOSUB GETDATA
FOR I = 1 TO SIZE
IF DATAIN$(I) <> DATAOUT$(I) THEN
PRINT #ZPLPORT: "^XA^FO100,100^A0N,50,50^FDERROR:";
PRINT #ZPLPORT: DATAOUT$(I);"^XZ"
END IF
NEXT I
END
REM **** SUBROUTINE STOREDATA *************************
REM INPUT: ZPLPORT, DATAIN$, SIZE, FILENAME$ ***********
SUB STOREDATA
PRINT #ZPLPORT: "^XA^DF" & FILENAME$ & "^FS"
PRINT #ZPLPORT: "^FX"; SIZE; "^FS"
FOR I = 1 TO SIZE
PRINT #ZPLPORT: "^FX" & DATAIN$(I) & "^FS"
```

```
NEXT I
PRINT #ZPLPORT: "^XZ"
RETURN
REM **** SUBROUTINE GETDATA - *************************
REM INPUT: ZPLPORT, FILENAME$ *************************
REM ** OUTPUT: DECLARES AND FILLS DATAOUT$ AND FILLS SIZE
SUB GETDATA
PRINT #ZPLPORT: "^XA^HF" & FILENAME$ & "^XZ"
SLEEP 1
LET RESULT$ = ""
FOR J = 1 TO 25
LET A = READ(ZPLPORT,TEMP$,5000)
LET RESULT$ = RESULT$ & TEMP$
IF POS(RESULT$,"^XZ") <> 0 THEN
EXIT FOR
END IF
SLEEP 1
NEXT J
LET RESULT$(1:POS(RESULT$,"^FX")+2) = ""
LET SIZE = VAL(EXTRACT$(RESULT$,"","^"))
DECLARE STRING DATAOUT$(SIZE)
FOR I = 1 TO SIZE
LET RESULT$(1:POS(RESULT$,"^FX")+2) = ""
LET DATAOUT$(I) = EXTRACT$(RESULT$,"","^")
NEXT I
LET RESULT$ = ""
LET TEMP$ = ""
RETURN
```

STORE

This command saves the program currently in memory as the specified file name. The format listed below is used.

Format

`STORE <filename$>`

Parameters

`<filename$>` = the name of the file to be stored. Drive location and file name must be in quotation marks.

Example

This is an example of how to use the `STORE` command:

```
STORE "E:PROGRAM1.BAS"
```

Comments

For a file name to be valid, it must conform to the 8.3 Rule: each file must have no more than eight characters in the file name and have a three-character extension. Here the extension is always `.BAS` (for example, `MAXIMUM8.BAS`).

This is an interactive command that takes effect as soon as it is received by the printer.

The ZBI-Developer IDE will take care of this for you with the `SEND TO` option on your program.

LOAD

This command transfers a program file previously stored in the printer's memory and opens it in the ZBI Program Memory.

If the program file does not exist, the ZBI Program Memory is cleared and no program is opened.

Format

`LOAD <filename$>`

Parameters

`<filename$>` = the file name to be loaded into memory. Drive location and file name must be in quotation marks. If the drive location is not specified, all drives will be searched.

Example

Here are examples of how to use the LOAD command:

```
LOAD "PROGRAM1.BAS"
LOAD "E:PROGRAM1.BAS"
```

Comments

This is an interactive command that takes effect as soon as it is received by the printer.

DIR

This command, with no filter included, prompts the printer to list all of the ZBI programs residing in all printer memory locations.

Including a filter signals the printer to limit the search; including a drive location signals the printer to search in only one location.

Asterisks (*) are used as wild cards. A wild card (*) finds every incidence of a particular request. The example here, `DIR "B:*.BAS"`, signals the printer to search for every file with a `.BAS` extension in B: memory.

Format

`DIR [<filter$>]`

Parameters

`[<filter$>]` = the name of the file to be accessed (optional). Drive location and file name must be in quotation marks.

Default = `"*:*.bas"`

NOTE: Quotes must be around what you are doing.

Example

N/A

Comments

This is an interactive command that takes effect as soon as it is received by the printer.

DELETE

This command removes a specified file from the printer's memory.

Format

`DELETE <filename$>`

Parameters

`<filename$>` = the name of the file to be deleted. Drive location and filename must be in quotation marks.

Example

This is an example of deleting a specified file from printer memory:

```
DELETE "E:PROGRAM1.BAS"
```

Comments

This is an interactive command that takes effect as soon as it is received by the printer.

Comma Separated Values (CSV)

This section describes the functions to access CSV files and ASCII plain-text files. Here is a quick list of these commands:

Accessing Comma Separated Value (CSV) and Text File Functions

CSVLOAD
Loads the contents of a CSV file in a two dimensional string array.

CSVSTORE
Stores the contents of a two dimensional string array in a CSV file.

TXTLOAD
Loads the contents of an ASCII plain-text file into a string variable.

TXTSTORE
Stores the contents of a string variable in an ASCII plain text file.

CSVLOAD

This function will load the delimited values from a CSV file, defined by FILENAME$, and store them in the two-dimensional array, DEST$.

Format

```
CSVLOAD(DEST$, FILENAME$)

CSVLOAD(DEST$, FILENAME$, DELIM$)
```

Parameters

DEST$ = two dimensional array that will hold the rows and columns from the CSV file specified by the FILENAME$ variable. If there is not enough room in DEST$, or if it has the wrong size, it will be changed to fit the data from the file. The data originally in DEST$ will be overwritten.

FILENAME$ = name of the file to load. Drive location and file name must be in quotation marks. The file extension must be either ".CSV" or ".TXT".

DELIM$ = optional delimiter that is used in the CSV file instead of a comma. If DELIM$ is not provided a comma will be used by default. The delimiter must be a single character that is not a quote, carriage return, or newline.

Returns

The number of elements in each row of the CSV file. The function will return 0 if errors were detected in the CSV file, or if the file could not be read.

Example

This example shows how to print the values in a CSV file with a comma delimiter.

```
10  DECLARE STRING CSVDB$(1,2)
20  LET FILENAME$ = "E:RECORDS.CSV"
30  LET NUMOFCOLS = CSVLOAD(CSVDB$, FILENAME$)
40  LET NUMOFROWS = ROWSIZE(CSVDB$)
100 FOR I = 1 TO NUMOFROWS STEP 1
110     FOR J = 1 TO NUMOFCOLS STEP 1
120         PRINT CSVDB$(I, J), " ";
200     NEXT J
210     PRINT ""
300 NEXT I
```

Example

This example shows how to print the values in a CSV file that uses a '|' as a delimiter.

```
10  DECLARE STRING CSVDB$(1,2)
20  LET FILENAME$ = "E:EMPLOYEE.CSV"
30  LET NUMOFCOLS = CSVLOAD(CSVDB$, FILENAME$, "|")
40  LET NUMOFROWS = ROWSIZE(CSVDB$)
100 FOR I = 1 TO NUMOFROWS STEP 1
110     FOR J = 1 TO NUMOFCOLS STEP 1
120         PRINT CSVDB$(I, J), " ";
200     NEXT J
210     PRINT ""
300 NEXT I
```

Comments

The maximum CSV file size supported will vary based upon available RAM within the printer.

CSV File Information

The file format should follow the rules in IETF RFC 4180: http://tools.ietf.org/html/rfc4180

The maximum number of columns per row in a CSV file is 256.

Each row must be 2048 characters or less including the delimiter. The carriage return/line feed (CRLF) does not count toward the limit.

Each row in the CSV file must have the same number of elements. If there are any missing elements in the CSV file (indicated by two adjacent commas or a comma at the end of a row), they will be represented as empty strings.

If an element in the CSV file contains a quote, it should be represented as two quotes. Additionally, if an element contains a quote, a new line, a carriage return, or the delimiter character, the element must be within quotes. For example, a value that is used to store a measurement in feet and inches (4' 5") must be formatted as "4' 5""" within the CSV file.

CSVSTORE

This function will store the values of a two dimensional array into a CSV file on the file system. Each element within the array is treated as a single value within the CSV file.

Format

```
CSVSTORE(SRC$, FILENAME$)

CSVSTORE(SRC$, FILENAME$, DELIM$)
```

Parameters

- SRC$ = two dimensional array of strings to be written to a CSV file.
- FILENAME$ = name of the file to store the array contents. Drive location and file name must be in quotation marks. The file extension must be either ".CSV" or ".TXT".
- DELIM$ = optional delimiter that is used in the CSV file instead of a comma. If DELIM$ is not provided a comma will be used by default. The delimiter must be a single character that is not a quote, carriage return, or newline.

Returns

A 0 if there were no errors. A 1 is returned if SRC$ is not a string array, if the file could not be written, or if SRC$ contains errors that prevent the file from being stored.

Example

This example shows how to convert a comma delimited CSV file into a "^" delimited TXT file and print the contents.

```
10  DECLARE STRING CSVDB$(1,2)
20  LET NUMOFCOLS = CSVLOAD(CSVDB$, "E:RECORDS.CSV")
30  LET CSVERROR = CSVSTORE(CSVDB$, "E:NEWREC.TXT", "^")
40  LET NUMOFCOLS = CSVLOAD(CSVDB$, "E:NEWREC.TXT", "^")
50  LET NUMOFROWS = ROWSIZE(CSVDB$)
100 FOR I = 1 TO NUMOFROWS STEP 1
110     FOR J = 1 TO NUMOFCOLS STEP 1
120         PRINT CSVDB$(I, J), " ";
200     NEXT J
210     PRINT ""
300 NEXT I
```

Comments

The elements of the array should follow the rules in IETF RFC 4180: http://tools.ietf.org/html/rfc4180

There is no limit on the number of columns per row when storing to a CSV file. However, a file stored with rows that exceed the column limit imposed by CSVLOAD will not be loaded by the CSVLOAD function.

There is no limit on the size of a row when stored to a CSV file. However, a file stored with rows that exceed the size limit imposed by CSVLOAD will not be loaded by the CSVLOAD function.

TXTLOAD

This function will read the contents of an ASCII text file into a ZBI string variable.

Format

```
TXTLOAD(DEST$, FILENAME$)
```

Parameters

- DEST$ = string to store the contents of FILENAME$.
- FILENAME$ = name of the file to read. Drive location and file name must be in quotation marks. The file extension must be either ".CSV" or ".TXT".

Returns

The number of bytes read from the file. The function will return 0 if the file could not be read.

Example

This example shows how to print out the contents of a file.

```
10 LET TXTSIZE = TXTLOAD(TXTDATA$, "E:MYDATA.TXT")
20 PRINT STR$(TXTSIZE), "bytes:", TXTDATA$
```

Comments

The data originally in DEST$ will be overwritten upon completion of this function.

TXTSTORE

This function will store the contents of a ZBI string in an ASCII text file.

Format

`TXTSTORE(SRC$, FILENAME$)`

Parameters

- SRC$ = string to store to `FILENAME$`.
- `FILENAME$` = name of the file to store. Drive location and file name must be in quotation marks. The file extension must be either ".CSV" or ".TXT".

Returns

Returns a 0 if there were no errors, otherwise a 1 is returned.

Example

This example shows how to append a text file.

```
10 LET TXTSIZE = TXTLOAD(TXTDATA$, "E:MYDATA.TXT")
11 REM Append a date/time stamp to the file
20 LET TXTDATA$ = TXTDATA$ & " " & DATE$ & " " & TIME$
30 LET TXTSIZE = TXTSTORE(TXTDATA$, "E:MYDATA.TXT")
40 PRINT TXTDATA$
```

Events

This section explains how to capture and trigger internal events in the printer. Here is a quick list of these commands:

Available Events
A table that correlates a ZBI event with an identification number.

ZBI Key Names
Details the names of each printer's control panel buttons, ZBI names, and ZBI event ID.

REGISTEREVENT
Sets up the HANDLEEVENT function to receive notification when the specified event has occurred.

UNREGISTEREVENT
Allows events that are currently set to be captured by the program to no longer be captured.

HANDLEEVENT
Once events have been registered, this function is used to see what events have occurred.

TRIGGEREVENT
Allows for control panel buttons to be triggered programmatically.

There are certain events in the printer that a ZBI 2.0 program can receive. To do this, the program first registers for the event. On a regular basis, call a function to handle events. When an event occurs that the program is registered for, the function will return the event's identification number.

Available Events

ZBI Event ID	ZBI Event
1	menu key
2	pause key
3	feed key
4	cancel key
5	up arrow key
6	plus key
7	minus key
8	enter key
9	setup exit key
10	select key
11	cancel all event
12	config label
13	timer1
14	timer2
15	timer3
16	timer4
17	timer5
18	spare unused
19	previous key

ZBI Event ID	ZBI Event
20	next save key
21	calibrate key
22	paper out set
23	paper out clear
24	ribbon out set
25	ribbon out clear
26	head too hot set
27	head too hot clear
28	head cold set
29	head cold clear
30	head open set
31	head open clear
32	supply too hot set
33	supply too hot clear
34	ribbon in set
35	ribbon in clear
36	rewind full set
37	rewind full clear
38	cutter jammed set
39	cutter jammed clear
40	paused set
41	paused clear
42	pq completed set
43	pq completed clear
44	label ready set
45	label ready clear
46	head element bad set
47	head element bad clear
48	basic runtime set
49	basic runtime clear
50	basic forced set
51	basic forced clear
52	power on set
53	power on clear
54	clean printhead set

ZBI Event ID	ZBI Event
55	clean printhead clear
56	media low set
57	media low clear
58	ribbon low set
59	ribbon low clear
60	replace head set
61	replace head clear
62	battery low set
63	battery low clear
64	rfid error set
65	rfid error clear
66	any messages set
67	any messages clear
68	auto baud
69	factory default
70	networking default
71	networking factory
72	print width
73	darkness adjust
74	calibrate
75	scroll key
76	soft key 1
77	soft key 2
78	ribbon cartridge authentication error set
79	ribbon cartridge authentication error clear

ZBI Key Names

This section details the names to use for each printer's front panel buttons when creating ZBI 2.0 programs to capture the buttons.

ZT200/ZT400/ZT500/ZT600/ZD500/QIn

ZT2X0	ZT400/ ZT500/ ZT600	ZD500	QIn	ZBI Event ID	ZBI Name
Left Soft button				76	soft key 1
Right Soft Button				77	soft key 2
Plus	Up Arrow			6	plus key

ZT2X0	ZT400/ ZT500/ ZT600	ZD500	QIn	ZBI Event ID	ZBI Name
Minus	Down Arrow			7	minus key
Left Arrow				19	previous key
Right Arrow				20	next save key
Setup	OK	Check	OK	10	select key
Pause			no key	2	pause key
Feed				3	feed key
Cancel			no key	4	cancel key

Xi4/RXi4/XiIIIPlus/PAX4/105SL/ZE500

XiIIIPlus/PAX4/Xi4/ RXi4/ ZE500/105SL Plus Front Panel Key	105SL Front Panel Key	ZBI Event ID	ZBI Name
Right Oval	Plus (+)	6	plus key
Left Oval	Minus (-)	7	minus key
Previous		19	previous key
Next/Save		20	next save key
Setup/Exit		9	setup exit key
Pause		2	pause key
Feed		3	feed key
Cancel		4	cancel key
Calibrate		21	calibrate key

HC100

Front Panel Key	ZBI Event ID	ZBI Name
Pause	2	pause key
Feed	3	feed key
Eject		eject key

ZM400/ZM600/RZ400/RZ600/Z4Mplus/Z6Mplus

Front Panel Key	ZBI Event ID	ZBI Name
Feed	3	feed key
Pause	2	pause key
Cancel	4	cancel key
Setup/Exit	9	setup exit key
Select	10	select key
Plus (+)	6	plus key

Front Panel Key	ZBI Event ID	ZBI Name
Minus (-)	7	`minus key`

S4M

Front Panel Key	ZBI Event ID	ZBI Name
Menu	1	`menu key`
Enter	8	`enter key`
Cancel	4	`cancel key`
Feed	3	`feed key`
Pause	2	`pause key`
Left Arrow	4	`cancel key`
Right Arrow	3	`feed key`
Up Arrow	5	`up arrow key`
Down Arrow	2	`pause key`

G-Series

Front Panel Key	ZBI Event ID	ZBI Name
Feed key	3	`Feed key`
Select key	10	`Select key`
Scroll key	75	`Scroll key`

KR403 / 2824 Plus Series

Front Panel Key	ZBI Event ID	ZBI Name
Feed key	3	`Feed key`

REGISTEREVENT

This function will set up the HANDLEEVENT function to receive notification when the specified event has occurred. Events can be registered for one time or until the program is exited.

NOTE: If an event occurs twice or more before the HANDLEEVENT function is called, only one event will be received.

Format

- REGISTEREVENT(X)
- REGISTEREVENT(X,Y)
- REGISTEREVENT(X,Y,Z)

Parameters

- (X) = This is the ID of the event being registered for.
- (Y) = If Y=1: the event happens once; If Y=0: the event stays registered for the duration of the program, or until it is unregistered.
- (Z) = For System Events: if Z=0, the event will still be handled by the printer. If Z=1, then only ZBI will receive the event.

  For Timer Events: this is the timer interval in mSec. If the interval is less than 0 or greater than 1,000,000,000, it is set to 1000.

Returns

The ID of the successfully registered event. If an event was not successfully registered, a -1 is returned.

Example

Here is an example of how to use the REGISTEREVENT command:

```
1 REM This example shows how to override the functionality of the feed
1 REM key
1 REM using the event system. After all why waste a label when you
1 REM could put
1 REM valuable information there
AUTONUM 1,1
CLOSE ALL
LET ZPLPORT = 1
OPEN #ZPLPORT: NAME "ZPL"
LET FEEDKEY = 3
LET TMP = REGISTEREVENT(FEEDKEY, 0, 1)
DO WHILE 1 = 1
LET EVT = HANDLEEVENT()
IF EVT = FEEDKEY THEN
GOSUB PRINTINFO
END IF
SLEEP 1
LOOP
REM **** SUBROUTINE PRINTINFO *** expects ZPLPORT *******
SUB PRINTINFO
PRINT #ZPLPORT: "^XA"
PRINT #ZPLPORT: "^FO30,30^A0N,50,50^FDZebra Technologies^FS"
PRINT #ZPLPORT: "^FO30,85^A0N,35,35^FDwww.zebra.com^FS"
PRINT #ZPLPORT: "^FO30,125^A0N,35,35^FDsupport.zebra.com^FS"
PRINT #ZPLPORT: "^FO30,165^A0N,35,35^FDFW Version: "
PRINT #ZPLPORT: GETVAR$("appl.name") & "^FS"
PRINT #ZPLPORT: "^FO30,205^A0N,35,35^FDPrinter Unique ID:"
PRINT #ZPLPORT: GETVAR$("device.unique_id") & "^FS"
PRINT #ZPLPORT: "^FO30,245^A0N,35,35^FDActive Network: "
PRINT #ZPLPORT: GETVAR$("ip.active_network") & "^FS"
PRINT #ZPLPORT: "^FO30,285^A0N,35,35^FDZBI Memory Usage: "
PRINT #ZPLPORT: GETVAR$("zbi.start_info.memory_alloc") & "^FS"
PRINT #ZPLPORT: "^FO30,325^A0N,35,35^FDOdometer: "
PRINT #ZPLPORT: GETVAR$("odometer.total_print_length") & "^FS"
PRINT #ZPLPORT: "^XZ"
```

Comments

None

UNREGISTEREVENT

This function allows events that are currently set to be captured by the program to no longer be captured. Once called events will return to the normal method of processing if the REGISTEREVENT function Z parameter was set to 1.

Format

`UNREGISTEREVENT(X)`

Parameters

`(X)` = the ID of the event to stop

Returns

0 if the event is a valid event to unregister. A -1 if the event does not exist.

Example

Here is an example of how to use the UNREGISTEREVENT command:

```
AUTONUM 1,1
LET OUTSTR$ = "Processing"
LET LOOPCTR = 200
LET TIMER5 = 17
LET TMP = REGISTEREVENT(TIMER5, 0, 1000)
DO WHILE LOOPCTR > 0
LET EVT = HANDLEEVENT()
IF EVT = TIMER5 THEN
LET A = SETVAR("device.frontpanel.line2",OUTSTR$)
LET OUTSTR$ = OUTSTR$ & "."
IF LEN(OUTSTR$) >16 THEN
LET OUTSTR$ = "Processing"
END IF
END IF
LET LOOPCTR = LOOPCTR - 1
SLEEP 1
LOOP
LET TMP = UNREGISTEREVENT(TIMER5)
LET A = SETVAR("device.frontpanel.line2","")
END
```

Comments

None

HANDLEEVENT

Once events have been registered, this function is used to see what events have occurred.

Format

`HANDLEEVENT( )`

Parameters

N/A

Returns

The ID of the event that occurred. One event at a time will be returned through this function. The order of the events are based on priority. The priority is based on the ID number of the event, with the exception of the timer events, which have the highest priority.

Example

Here are examples of how to use the HANDLEEVENT command:

```
1 REM This example shows how to override the feed key functionality
1 REM using the event system. Why waste a label when you could put
1 REM valuable information there
AUTONUM 1,1
CLOSE ALL
LET ZPLPORT = 1
OPEN #ZPLPORT: NAME "ZPL"
LET FEEDKEY = 3
LET TMP = REGISTEREVENT(FEEDKEY, 0, 1)
DO WHILE 1 = 1
LET EVT = HANDLEEVENT()
IF EVT = FEEDKEY THEN
GOSUB PRINTINFO
END IF
SLEEP 1
LOOP
REM ******** SUBROUTINE PRINTINFO ***
REM *** expects ZPLPORT *************
SUB PRINTINFO
PRINT #ZPLPORT: "^XA"
PRINT #ZPLPORT: "^FO30,30^A0N,50,50";
PRINT #ZPLPORT: "^FDZebra Technologies^FS"
PRINT #ZPLPORT: "^FO30,85^A0N,35,35";
PRINT #ZPLPORT: "^FDwww.zebra.com^FS"
PRINT #ZPLPORT: "^FO30,125^A0N,35,35";
PRINT #ZPLPORT: "^FDsupport.zebra.com^FS"
PRINT #ZPLPORT: "^FO30,165^A0N,35,35";
PRINT #ZPLPORT: "^FDFW Version: ";
PRINT #ZPLPORT: GETVAR$("appl.name") & "^FS"
PRINT #ZPLPORT: "^FO30,205^A0N,35,35";
PRINT #ZPLPORT: "^FDPrinter Unique ID:";
PRINT #ZPLPORT: GETVAR$("device.unique_id") & "^FS"
PRINT #ZPLPORT: "^FO30,245^A0N,35,35";
PRINT #ZPLPORT: "^FDActive Network: ";
PRINT #ZPLPORT: GETVAR$("ip.active_network") & "^FS"
PRINT #ZPLPORT: "^FO30,285^A0N,35,35";
PRINT #ZPLPORT: "^FDZBI Memory Usage: ";
PRINT #ZPLPORT: GETVAR$("zbi.start_info.memory_alloc") & "^FS"
PRINT #ZPLPORT: "^FO30,325^A0N,35,35";
PRINT #ZPLPORT: "^FDOdometer: ";
PRINT #ZPLPORT: GETVAR$("odometer.total_print_length") & "^FS"
PRINT #ZPLPORT: "^XZ"
```

Comments

None

TRIGGEREVENT

This function allows for front panel buttons to be triggered programatically.

Format

`TRIGGEREVENT(X)`

Parameters

X = the ID of the event from the possible event list to TRIGGER.

See the following printer tables for events that can be triggered by this command:

- Xi4/RXi4/XiIIIPlus/PAX4/105SL/ZE500 on page 523
- ZM400/ZM600/RZ400/RZ600/Z4Mplus/Z6Mplus on page 523
- S4M on page 524

Returns

Always returns 0.

Example

Here are examples of how to use the `TRIGGEREVENT` command:

```
1 REM THIS IS AN EXAMPLE OF HOW TO TRIGGER AN EVENT
AUTONUM 1,1
LET PAUSEKEY = 2
DO WHILE 1 = 1
LET A = TRIGGEREVENT(PAUSEKEY)
LET A = SETVAR("device.frontpanel.line2",str$(A))
SLEEP 2
LOOP
```

Comments

None

Systems

This section contain miscellaneous systems interface functions. Here's a quick list of these commands:

ISERROR

Returns a non-zero value if there is an internal error set in the printer.

ISWARNING

Returns a non-zero value if there is an internal warning set in the printer.

SLEEP

Specifies the time that the interpreter pauses.

SETERR

Sends a message to the printer to set the error flag.

CLRERR

Sends a message to the printer to clear the error flag.

ON ERROR

Prevents a program from halting in the event of an error.

ISERROR

This function returns a non-zero value if there is an internal error set in the printer. Otherwise, the numeral returned will 0.

Format
`ISERROR`

Parameters
N/A

Returns
0 for no errors; 1 if there is an error.

Example
Here is an example of the `ISERROR` command.

```
10 PRINT ISERROR
RUN
0
```

Comments
None

ISWARNING

This function returns a non-zero value if there is an internal warning set in the printer. Otherwise, the numeral returned will 0.

Format

`ISWARNING`

Parameters

N/A

Returns

0 for no errors; 1 if there is an error.

Example

Here is an example of the ISWARNING command.

```
10 PRINT ISWARNING
RUN
0
```

Comments

None

SLEEP

This command specifies the time that the interpreter pauses. This command could be sent to the printer after sending a label format to be printed. The interpreter pauses in its processing for the amount of time specified.

Format

SLEEP <A>

Parameters

<A> = the time in seconds (0 to 500) the interpreter pauses.

Example

This is an example of how to use the SLEEP command:

```
10
SLEEP 450
```

Comments

If a timer is needed, use the `Event` system. The timer will allow for processing other items, where SLEEP will stop execution of any ZBI commands for the specified SLEEP period.

This is a program command and must be preceded by a line number.

Calling SLEEP with <A> set to zero will force the ZBI task to yield to the rest of the system and allow any pending tasks to run (e.g., pending ZPL commands). If there are no pending tasks, ZBI will sleep for a minimum of 8 milliseconds.

SETERR

This command sends a message to the printer to set the error flag. A logical interpreter flag is triggered in the printer. This error is referenced as a BASIC Forced Error.

Format

SETERR

Parameters

N/A

Example

An example of the SETERR and CLRERR commands.

```
AUTONUM
1,1
OPEN #1:NAME "ZPL"
PRINT #1: "^XA^SXO,A,Y,Y^XZ"
CLOSE #1
FOR I=1 TO 10
SLEEP 5
IF MOD(I,2)=1 THEN
SETERR
ELSE
CLRERR
ENDIF
NEXT I
```

Comments

This is a program command and must be preceded by a line number.

CLRERR

This command sends a message to the printer to clear the error flag. A logical interpreter flag is cleared in the printer. This error is referenced as a BASIC Forced Error.

Format

```
10 CLRERR
```

Parameters

N/A

Example

See SETERR on page 534.

Comments

This is a program command that is preceded by a line number.

ON ERROR

The ON ERROR command can be used to prevent a program from halting in the event of an error. If an error occurs in a previous line during program execution, the ON ERROR statement calls the GOTO or GOSUB statement and allows the program to continue.

Format

```
ON ERROR GOTO <A>
```

```
ON ERROR GOSUB <A>
```

Parameters

<A> = the destination location in the program should an error be triggered on the previous line.

Example

This is an example of how to use the ON ERROR command:

```
30 LET A = B/C
40 ON ERROR GOTO 100
...
100 PRINT "DIVIDE BY ZERO OCCURRED"
110 LET A = 0
120 GOTO 50
...
```

See TCP Server on page 504 or UDP Server on page 505.

Comments

If there is no error, this line is ignored.

This is a program command that is preceded by a line number.

Applicator Functions

The printer applicator port option can be controlled in part or completely by ZBI 2. When ZBI takes control of a pin, the printer's built-in applicator functionality will not have access to that pin. This function will allow the printer to perform some of the functionality that a programmable logic controller (PLC) could.

AUXPORT_STEALPIN
Takes control of a pin and allows ZBI to perform other actions on the pin.

AUXPORT_SETPIN
Sets the output level on an applicator pin.

AUXPORT_GETPIN
Retrieves the state of the applicator pin.

AUXPORT_RELEASEPIN
Returns a pin controlled by ZBI to normal printer operation.

AUXPORT_STEALPIN

This function will take control of a pin and allow ZBI to perform other actions on the pin.

Format

AUXPORT_STEALPIN(X)

Parameters

X = perform action on this applicator port pin.

Returns

This function returns -1 upon failure and 0 upon success.

Example

This is an example of the AUXPORT_STEALPIN command:

```
1 REM Demo applicator to show control of applicator pins
1 REM on the printer
1 REM The application is to create a light pole with an
1 REM external feed button
AUTONUM 1,1
LET RED = 9
LET YELLOW = 10
LET GREEN = 11
LET BUTTON = 4
LET FEED_KEY = 3
LET TMP = AUXPORT_STEALPIN(RED)
LET TMP = AUXPORT_STEALPIN(YELLOW)
LET TMP = AUXPORT_STEALPIN(GREEN)
LET TMP = AUXPORT_STEALPIN(BUTTON)
DO WHILE 1 = 1
SLEEP 1
IF ISERROR = 1 THEN
LET TMP = AUXPORT_SETPIN(RED,1)
LET TMP = AUXPORT_SETPIN(YELLOW,0)
LET TMP = AUXPORT_SETPIN(GREEN,0)
ELSE IF ISWARNING = 1 THEN
LET TMP = AUXPORT_SETPIN(RED,0)
LET TMP = AUXPORT_SETPIN(YELLOW,1)
LET TMP = AUXPORT_SETPIN(GREEN,0)
ELSE
LET TMP = AUXPORT_SETPIN(RED,0)
LET TMP = AUXPORT_SETPIN(YELLOW,0)
LET TMP = AUXPORT_SETPIN(GREEN,1)
END IF
IF AUXPORT_GETPIN(BUTTON) = 1 THEN
LET A = TRIGGEREVENT(FEED_KEY)
END IF
LOOP
```

Comments

If this pin is not controlled via ZBI (power pin), this function will return -1.

AUXPORT_SETPIN

This function sets the output level on an applicator pin.

Format

AUXPORT_SETPIN(X,Y)

Parameters

X = perform action on this applicator port pin.

Y = The value to set on the pin (1 = high, 0 = low).

Returns

This function returns -1 upon failure and 0 upon success.

Example

See AUXPORT_STEALPIN on page 538.

Comments

If this pin is not controlled via ZBI (power pin), this function will return -1. See AUXPORT_STEALPIN on page 538.

AUXPORT_GETPIN

This function will retrieve the state of the applicator pin.

Format

AUXPORT_GETPIN(X)

Parameters

X = perform action on this applicator port pin.

Returns

This function returns 1 if pin is in high state, 0 in low state, and -1 upon failure.

Example

See AUXPORT_STEALPIN on page 538.

Comments

If this pin is not controlled via ZBI (power pin), this function will return -1. See AUXPORT_STEALPIN on page 538.

AUXPORT_RELEASEPIN

This function returns a pin controlled by ZBI to normal printer operation.

Format

AUXPORT_RELEASEPIN(X)

Parameters

X = perform action on this applicator port pin.

Returns

This function returns -1 upon failure and 0 upon success.

Example

This is an example of the AUXPORT_RELEASEPIN command:

```
90 LET TMP = AUXPORT_RELEASEPIN(X)
```

Comments

If this pin is not controlled via ZBI (power pin), this function will return -1. See AUXPORT_STEALPIN on page 538.

String Functions

This section identifies how to handle string manipulation. Here is a quick list of these commands:

LCASE$
Converts a string to all lowercase characters.

CHR$
Takes a value between 0 and 255 and puts that value into a string.

LTRIM$
Removes leading spaces from a string.

REPEAT$
Creates multiple copies of a string combined into a new string.

RTRIM$
Returns a string with trailing spaces removed

SPLIT
Splits a string into sub-strings

SPLITCOUNT
Returns the number of sub-strings that would be returned by the SPLIT function.

UCASE$
Converts a string to all uppercase characters

EXTRACT$
Searches for a string based on a starting and ending string.

ORD
Returns the ASCII value of the first character of string A$.

POS
Returns the location of the first occurrence of a search string in the target string.

LEN
Returns the length of a string.

LCASE$

This function will convert a string to all lowercase characters.

Format

`LCASE$ (A$)`

Parameters

`(A$)` = the string that will be converted

Returns

The characters in A$ converted to lowercase.

Example

This is an example of how to use the LCASE$ command.

```
10 LET B$=LCASE$ ("Hello World")
20 PRINT B$
RUN
hello world
```

Comments

This will only work on non-accented Latin characters, A-Z.

CHR$

This function takes a value between 0 and 255 and puts that value into a string.

Format

CHR$(VAL)

Parameters

(VAL)= The numeric value of the string character.

Returns

A single character string containing the value entered.

Example

This is an example of how to use the CHR$ command to easily put control characters into strings:

```
10 LET NULL$=CHR$(0)
20 LET STX$=CHR$(2)
30 LET ETX$=CHR$(3)
40 LET EOT$=CHR$(4)
```

Comments

None

LTRIM$

This function removes leading spaces from a string.

Format

`LTRIM$(A$)`

Parameters

`(A$) =` the string to convert.

Returns

The string in A$ with no spaces.

Example

This is an example of how to use the LTRIM$(A$)command:

```
10 LET A$=" Hello"
20 PRINT LTRIM$(A$)
RUN
Hello
```

Comments

None

REPEAT$

This function creates multiple copies of a string combined into a new string.

Format

`REPEAT$(A$,M)`

Parameters

- A$ = the base string to duplicate
- `M` = the number of times to duplicate A$

Returns

A string containing `M` copies of A$. Note: When M=0, an empty string is returned.

Example

This is an example of how to use the REPEAT$(A$,M)command:

```
10 PRINT REPEAT$("Hello",3)
RUN
HelloHelloHello
```

Comments

None

RTRIM$

This function returns a string with trailing spaces removed.

Format
`RTRIM$(A$)`

Parameters
`(A$)` = the base string

Returns
A$ with trailing spaces removed.

Example

This is an example of how to use the RTRIM$(A$)command:

```
10 LET A$="Hello "
20 LET B$="World"
30 PRINT A$ & B$
40 PRINT RTRIM$(A$)& B$
RUN
Hello World
HelloWorld
```

Comments
None

SPLIT

This function allows a string to be split into sub-strings.

Format

```
SPLIT(DEST$,SOURCE$,DELIMITER$)

SPLIT(DEST$,SOURCE$,DELIMITER$,MAXCOUNT)
```

Parameters

- `DEST$` = the array to populate with the sub-strings created by the split
- `SOURCE$` = the string that will be searched for the provided delimiter
- `DELIMITER$` = the delimiter string (may be more than one character) to search for
- `MAXCOUNT` = the maximum number of sub-strings the string should be split into. A negative value will return every sub-string created by the split. A value of zero will return empty strings in the array. If not specified, the limit will be the maximum size of the array.

Returns

The number of sub-strings placed into the DEST$ array. If the number of sub-strings is less than the size of DEST$, the remaining elements of the array will be set to empty strings.

Example

This is an example of how to use the SPLIT command:

```
1 REM Example - This example show how the SPLIT and SPLITCOUNT
1 REM commands can be
1 REM used to merge a comma separated variable string(CSV)
1 REM into a stored format
AUTONUM 1,1
SLEEP 10
DECLARE STRING TESTDATA$(5)
REM data format = <Format Name>,<VAR 1>,<VAR 2>,...,<VAR N>
LET TESTDATA$(1) = "E:PRICETAG.ZPL,FRED'S OATS,
$1.25,C:126789:325,123456789"
LET TESTDATA$(2) = "E:PRICETAG.ZPL,FRED'S OATS,
$2.25,C:126789:325,123456789"
LET TESTDATA$(3) = "E:PRICETAG.ZPL,FRED'S OATS,
$3.25,C:126789:325,123456789"
LET TESTDATA$(4) = "E:PRICETAG.ZPL,FRED'S OATS,
$4.25,C:123489:325,123456789"
LET TESTDATA$(5) = "E:PRICETAG.ZPL,FRED'S OATS,
$5.25,C:123459:325,123456789"
LET ZPLPORT = 2
OPEN #ZPLPORT: NAME "ZPL"
FOR T = 1 TO 5
LET DATA$ = TESTDATA$(T)
GOSUB CSVPRINTER
NEXT T
END
REM ******* Subroutine CSVPRINTER, expects DATA$ and ZPLPORT
 ***************
SUB CSVPRINTER
LET CNT = SPLITCOUNT(DATA$, ",")
DECLARE STRING SPLITSTRING$(CNT)
ON ERROR GOTO RECOVERY
LET CNT = SPLIT(SPLITSTRING$,DATA$,",")
PRINT #ZPLPORT: "^XA^XF";SPLITSTRING$(1);"^FS"
IF CNT >= 2 THEN
FOR I = 2 TO CNT
PRINT #ZPLPORT: "^FN";I-1;"^FD";SPLITSTRING$(I);"^FS"
NEXT I
END IF
PRINT #ZPLPORT: "^XZ"
SUB RECOVERY
RETURN
```

Example

This is an example of how to use the SPLIT command:

```
1 REM Example - Shows how the SPLIT and SPLITCOUNT commands can be used
 to
1 REM merge a comma separated variable string(CSV) into a stored
 format
AUTONUM 1,1
SLEEP 10
DECLARE STRING TESTDATA$(5)
REM data format = <Format Name>,<VAR 1>,<VAR 2>,...,<VAR N>
LET F$="E:PRICETAG.ZPL"
LET TESTDATA$(1) = F$&",FRED'S ROLLED OATS,
$1.25,C:123456789:325,123456789"
LET TESTDATA$(2) = F$&",FRED'S ROLLED OATS,
$2.25,C:123456789:325,123456789"
LET TESTDATA$(3) = F$&",FRED'S ROLLED OATS,
$3.25,C:123456789:325,123456789"
LET TESTDATA$(4) = F$&",FRED'S ROLLED OATS,
$4.25,C:123456789:325,123456789"
LET TESTDATA$(5) = F$&",FRED'S ROLLED OATS,
$5.25,C:123456789:325,123456789"
LET ZPLPORT = 2
OPEN #ZPLPORT: NAME "ZPL"
FOR T = 1 TO 5
LET DATA$ = TESTDATA$(T)
GOSUB CSVPRINTER
NEXT T
END
REM ******* Subroutine CSVPRINTER, expects DATA$ and ZPLPORT
 *****************
SUB CSVPRINTER
LET CNT = SPLITCOUNT(DATA$, ",")
DECLARE STRING SPLITSTRING$(CNT)
ON ERROR GOTO RECOVERY
LET CNT = SPLIT(SPLITSTRING$,DATA$,",")
PRINT #ZPLPORT: "^XA^XF";SPLITSTRING$(1);"^FS"
IF CNT >= 2 THEN
FOR I = 2 TO CNT
PRINT #ZPLPORT: "^FN";I-1;"^FD";SPLITSTRING$(I);"^FS"
NEXT I
END IF
PRINT #ZPLPORT: "^XZ"
SUB RECOVERY
RETURN
```

Comments

If the delimiter is an empty string, or does not appear in the SOURCE$ string, the first entry of the array will be the source string and all other elements will be empty strings.

When the SPLIT function encounters a delimiter at the beginning or end of the source string, or two delimiters in a row, it populates the corresponding array element with an empty string.

If MAXCOUNT is larger than the number of returned sub-strings (N), the last MAXCOUNT - N array elements will be empty strings. If MAXCOUNT is larger than the destination array or is negative, the size of the array will be used as the MAXCOUNT. Therefore, the smallest value among the value of MAXCOUNT, the size of the return array, or the number of sub-strings found determines the maximum number of sub-strings that will be returned.

If MAXCOUNT is less than the number of delimiters in a string the last string in the array will hold the end of the string starting from where the last delimiter was found. For example, if SOURCE$ = "one,two,three,four,five", DELIMITER$ = ",", and MAXCOUNT = 2, the output would be two strings: "one" and "two,three,four,five".

If a two dimensional array is provided for DEST$, the array will be filled linearly. For example, an array that is 2 x 3 (for example, DECLARE STRING MYARRAY$(2,3)) will be filled from (0,0), then (0,1) up to (2,3).

SPLITCOUNT

This function returns the number of sub-strings that would be returned by the SPLIT function.

Format

`SPLITCOUNT(SOURCE$, DELIMITER$)`

Parameters

`SOURCE$` = the string that will be searched for the provided delimiter.

`DELIMITER$` =5

Returns

The number of sub-strings that would be returned by the `SPLITCOUNT` function.

Example

This function shows how to determine the number of sub-strings that the `SPLITCOUNT` command would produce

```
10 LET CNT = SPLITCOUNT("ONE,,,FOUR,FIVE,,SEVEN,", ",")
20 PRINT "Number of sub-strings returned is", STR$(CNT)
RUN
Number of sub-strings returned is 8
```

Comments

None

UCASE$

This function converts a string to all uppercase characters.

Format

UCASE$(A$)

Parameters

A$ = the base string to convert

Returns

A$ converted to uppercase.

Example

This is an example of how to use the UCASE$(A$)command:

```
10 LET A$="Zebra Technologies"
20 PRINT UCASE$(A$)
RUN
ZEBRA TECHNOLOGIES
```

Example

This is an example of how to capitalize a line.

```
10 LET A$="The Cow jUmped Over THE Moon."
20 LET A$=LCASE$(A$)
30 LET A$(1:1)=UCASE$(A$(1:1))
40 PRINT A$
RUN
The cow jumped over the moon.
```

Comments

This will only convert non-accented Latin characters, a-z.

EXTRACT$

This function searches for a string based on a starting and ending string. When these two strings are found, the string between them is returned.

NOTE: If the EXTRACT$ command encounters a carriage return line feed before encountering the beginning character or the ending character, it returns null.

Format

```
EXTRACT$ (CHANNEL, START$, STOP$)
```

```
EXTRACT$ (A$, START$, STOP$)
```

Parameters

- CHANNEL = extracts data from this channel
- A$ = the source string
- START$ = Once this string is found, the extract pulls characters immediately following.
- STOP$ = the extraction stops when this string is found

Example

This example shows how to extract the word Technologies from this string: Zebra,Technologies,Corporation.

This is what the program looks like to accomplish this:

```
10 LET A$ = "Zebra,Technologies,Corporation,"
20 LET DATA$ = EXTRACT$(A$,",",",",")
```

Example

This example shows how the EXTRACT$ command works from an open port:

```
10 OPEN #1: NAME "SER"
20 LET DATA$ = EXTRACT$(1,",",",",")
```

Notice how the quotes are used to show a literal character, in this case a comma.

Example

This example shows how the start and stop points are variable; a variable name is used instead of the literal:

```
10 LET B$ = ","
20 LET A$ = "Zebra,Technologies,Corporation"
30 LET DATA$ = EXTRACT$(A$,B$,B$)
40 PRINT DATA$
RUN
Technologies
```

Example

This example shows how an empty string can be used to extract from the start of the input string to the end string:

```
10 LET IN$ = "BLAH BLAH <END>"
20 LET B$ = EXTRACT$(IN$, "", "<END>")
30 PRINT B$
RUN
BLAH BLAH
```

Example

This example will use an empty string to extract to the end of a line:

```
10 LET IN$ = "BLAH <START> THE DATA"
20 LET B$ = EXTRACT$(IN$, "<START>", "")
30 PRINT B$
RUN
THE DATA
```

Comments

EXTRACT$ reads in and discards data until the start criteria is met. Then, all data is returned up to the stop criteria.

ORD

This function returns the ASCII value of the first character of string A$.

Format

`ORD(A$)`

Parameters

A$ = Input string: only the first character will be used.

Returns

The ASCII value of the first character.

Example

This is an example of how to use the ORD(A$)command:

```
10 LET A$="ABC"
20 PRINT ORD(A$)
RUN
65
```

Comments

None

POS

This function returns the location of the first occurrence of a search string in the target string. It can be assigned an index.

Format

`POS(A$,B$)`

`POS(A$,B$,M)`

Parameters

- A$ = the target string to search
- B$ = the search string to find in A$
- `M` = The index to start looking for B$. If omitted, the search will start at the beginning of the string. `M` must be greater than zero.

Returns

The location of the string. If the string is not found, this will return 0.

Example

This is an example of how to use the POS command:

```
10 LET A$="Hello World"
20 LET B$="o"
30 PRINT POS(A$,B$)
40 PRINT POS(A$,B$,1)
50 PRINT POS(A$,B$,6)
RUN
5
5
8
```

Comments

None

LEN

This function returns the length of a string.

Format

`LEN(A$)`

Parameters

A$ = the target string from which to determine the length.

Returns

The length of the string.

Example

This example identifies the length of a string. Hello World is 11 characters, as follows:

```
10 LET A$="Hello World"
20 PRINT LEN(A$)
RUN
11
```

Comments

None

Math Functions

This section identifies how to handle mathematical calculations. Here is a quick list of these commands:

STR$
Converts a number to a string.

MAX
Returns the greater value between two numbers.

MIN
Returns the smaller value of two numbers.

MAXNUM
returns the largest number permitted by this machine.

MOD
Computes the remainder from division.

VAL
Evaluates the number represented by a string.

INTTOHEX$
Takes a numeric value and converts it into a hexadecimal string.

HEXTOINT
Converts hexadecimal strings to integers.

STR$

This function converts a number to a string.

Format
STR$(X)

Parameters
X = the number to convert to a string

Returns
A string representing X.

Example

This is an example of how to use the STR$(X)command:

```
10 LET A=53
20 PRINT STR$(A)
RUN
53
```

Comments
None

MAX

This function returns the greater value between two numbers.

Format

`MAX(X,Y)`

Parameters

X = the first number to compare

Y = the second number to compare

Returns

The greater of X or Y.

Example

This is an example of how to use the MAX(X,Y)command:

```
10 LET A=-2
20 LET B=1
30 PRINT MAX(A,B)
RUN
1
```

Comments

None

MIN

This function returns the smaller value of two numbers.

Format

`MIN(X,Y)`

Parameters

X = the first number to compare

Y = the second number to compare

Returns

The smaller of X or Y.

Example

This is an example of how to use the MIN(X,Y)command:

```
10 LET A=-2
20 LET B=0
30 PRINT MIN(A,B)
RUN
-2
```

Comments

None

MAXNUM

This function returns the largest number permitted by this machine: 2,147,483,647.

Format
MAXNUM

Parameters
N/A

Returns
The largest number that the NUMERIC type can handle (2,147,483,647).

Example
This is an example of how to use the MAXNUM command:

```
10 PRINT MAXNUM
RUN
2147483647
```

Comments
None

MOD

This function computes the remainder from division. (This is known as the modulus.)

Format

`MOD(X,Y)`

Parameters

X = the value to be modulated (numerator).

Y = the base number or divisor (denominator).

Returns

The remainder of the division (X/Y).

Example

This is an example of how to use the MOD(X,Y)command:

```
10 PRINT MOD(25,10)
20 PRINT MOD(2,1)
30 PRINT MOD(3,2)
40 PRINT MOD(9,2)
50 PRINT MOD(-2,9)
60 PRINT MOD(2,0)
RUN
5
0
1
1
-2
ERROR OCCURRED ON LINE 60:DIVIDE BY ZERO
```

Comments

None

VAL

This function evaluates the number represented by a string.

Format

VAL(A$)

Parameters

A$ = This is the input string to pull the number from. Non-numbers are ignored.

Returns

The numeric representation of the string.

Example

This is an example of how to use the VAL(A$)command:

```
10 LET A$="123"
20 LET C=VAL(A$)
30 PRINT C
RUN
123

PRINT VAL("321A123")
321123
```

Comments

None

INTTOHEX$

This function will take a numeric value and convert it into a hexadecimal string. The range of values for integers is: -2,147,483,648 to +2,147,483,647

Format

```
INTTOHEX$(A)
```

Parameters

A = The numeric value to convert.

Returns

A string representing the integer in hex.

Example

These print statements show the output of the `INTTOHEX$` function given different values. These print statements show the output of the `INTTOHEX$` function given different values.

```
PRINT INTTOHEX$(1)
1
```

```
PRINT INTTOHEX$(10)
A
```

```
PRINT INTTOHEX$(16)
10
```

```
PRINT INTTOHEX$(20)
14
```

```
PRINT INTTOHEX$(30)
1E
```

```
PRINT INTTOHEX$(100)
64
```

```
PRINT INTTOHEX$(123124)
1EOF4
```

```
PRINT INTTOHEX$(-5)
0
```

```
PRINT INTTOHEX$(-99)
0
```

Comments

Negative values will be returned as 0.

HEXTOINT

This function will convert hexadecimal strings to integers.

Format
HEXTOINT(A$)

Parameters
A$ = The hex string to convert.

Returns
A integer string computed from the hexadecimal string.

Example
These print statements show the output of the `INTTOHEX` function given different values.

```
PRINT HEXTOINT("0")
0
```

```
PRINT HEXTOINT("A")
10
```

```
PRINT HEXTOINT("a")
10
```

```
PRINT HEXTOINT("1A")
26
```

```
PRINT HEXTOINT("10")
16
```

```
PRINT HEXTOINT("AaAa")
43690
```

```
PRINT HEXTOINT("AAAA")
43690
```

```
PRINT HEXTOINT("-1")
0
```

```
PRINT HEXTOINT("-A")
0
```

Comments

Negative values will be returned as 0.

Array Functions

This section describes the functions to search, resize, and query arrays.

REDIM
Changes the size of an array.

INSERTROW
Inserts a new row into an existing array.

DELROW
Deletes a new row from an existing array

ROWSIZE
Returns the number of rows in an array.

COLUMNSIZE
Returns the number of columns in an array.

FIND
Searches a string array for an occurrence of a sub-string.

REDIM

This command will change the dimensions of an array.

Format

```
REDIM <ARRAYNAME>(<SIZE>)
REDIM <ARRAYNAME>(<ROWS>,<COLUMNS>)
REDIM <ARRAYNAME$>(<SIZE>)
REDIM <ARRAYNAME$>(<ROWS>,<COLUMNS>)
```

Parameters

<SIZE> = new number of entries in a single dimension array.

<ROWS> = new number of rows in a two dimensional array.

<ROWS> = new number of rows in a two dimensional array.

<COLUMNS> = new number of columns in a two dimensional array.

Example

This example shows how to change a one dimensional numeric array.

Example

This example shows how to change a two dimensional string array.

```
10  DECLARE STRING NAMEAGES$(3,2)
20  LET NAMEAGES$(1,1) = "Abraham"
30  LET NAMEAGES$(1,2) = "Lincoln"
40  LET NAMEAGES$(2,1) = "Dwight"
50  LET NAMEAGES$(2,2) = "Eisenhower"
60  LET NAMEAGES$(3,1) = "Theodore"
70  LET NAMEAGES$(3,2) = "Roosevelt"
80  REDIM NAMEAGES$(5,2) ! Make room for more
```

Comments

The `REDIM` must have the same number of dimensions as the original declaration of the array.

- If the array has two dimensions, the second array bound cannot change. It must have the same value as the original declaration.
- If `REDIM` makes an array smaller, elements (or rows, for a two dimensional array) at the end of the array are discarded.
- If `REDIM` makes an array larger, elements (or rows) are added at the end of the array, and initialized as they would be with a `DECLARE`.

This can be an interactive command that takes effect as soon as it is received by the printer, or a program command that is preceded by a line number.

INSERTROW

This command will insert a new row into an existing array.

Format

`INSERTROW (<ARRAYNAME>, <INDEX>)`

Parameters

`<ARRAYNAME>` = array where the row will be inserted

`<INDEX>` = index of the row in the array that the new row will be inserted before

Example

This example shows how to insert a row into the middle of an array.

```
10  DECLARE NUMERIC SCORES(3)
20  LET SCORES(1) = 85
30  LET SCORES(2) = 92
40  LET SCORES(3) = 98
50  INSERTROW(SCORES, 2)
60  LET SCORES(2) = 100
```

Example

This example shows how to add a row into the end of an array.

```
10  DECLARE NUMERIC SCORES(3)
20  LET SCORES(1) = 85
30  LET SCORES(2) = 92
40  LET SCORES(3) = 98
50  INSERTROW(SCORES, 4)
60  LET SCORES(4) = 100
```

Comments

Inserting a row increases the size of the array by one row, and moves all the rows from `INDEX` to the end of the array up one row, leaving an empty row at position `INDEX`.

`INDEX` cannot be any larger the number of rows in the array plus one. If the number of rows plus one is provided, the new row will be added to the end of the array.

This can be an interactive command that takes effect as soon as it is received by the printer, or a program command that is preceded by a line number.

DELROW

This command will delete a row from an existing array.

Format

`DELROW (<ARRAYNAME>, <INDEX>)`

Parameters

`<ARRAYNAME>` = the array where the row will be deleted

`<INDEX>` = index of the row to delete from the array

Example

This example shows how to delete a row from the middle of an array.

```
10  DECLARE NUMERIC SCORES(5)
20  LET SCORES(1) = 85
30  LET SCORES(2) = 92
40  LET SCORES(3) = 98
50  LET SCORES(4) = 45
60  LET SCORES(5) = 100
70  DELROW(SCORES, 4) ! Remove the low score
```

Comments

This decreases the size of A by one row, and moves all the rows from INDEX to the end of the array down by one, overwriting the row at position INDEX.

`INDEX` cannot be any larger the number of rows in the array.

If the array only has one row, that row may not be deleted.

This can be an interactive command that takes effect as soon as it is received by the printer, or a program command that is preceded by a line number.

ROWSIZE

This function will return the number of rows in an array.

Format

ROWSIZE(A)

ROWSIZE(A$)

Parameters

A = integer array to query for the number of rows.

A$ = string array to query for the number of rows.

Returns

Returns a 0 if the variable is not an array. Returns the number of elements in the array if the array has only one dimension. Returns the size of the first dimension if the array has two dimensions.

Example

This example shows how to determine the number of elements in a one dimensional string array.

```
10 DECLARE STRING NAMES$(3)
20 LET NAMES$(1) = "Fred"
30 LET NAMES$(2) = "Wilma"
40 LET NAMES$(3) = "Barney"
50 REDIM NAMES$(4) ! Make room for Betty
60 LET NAMES$(4) = "Betty"
70 LET NUMOFNAMES = ROWSIZE(NAMES$)
80 PRINT NUMOFNAMES
```

Example

This example shows how to determine the number of rows in a two dimensional numeric array.

```
10 DECLARE NUMERIC SQROFTWOLOOKUP(3,2)
20 LET SQROFTWOLOOKUP (1,1) = 1
30 LET SQROFTWOLOOKUP (1,2) = 2
40 LET SQROFTWOLOOKUP (2,1) = 2
50 LET SQROFTWOLOOKUP (2,2) = 4
60 LET SQROFTWOLOOKUP (3,1) = 3
70 LET SQROFTWOLOOKUP (3,2) = 8
80 LET NUMOFSQRS = ROWSIZE(SQROFTWOLOOKUP)
90 PRINT NUMOFSQRS
```

COLUMNSIZE

This function will return the number of columns in an array.

Format

COLUMNSIZE(A)

COLUMNSIZE(A$)

Parameters

A = integer array to query for the number of columns.

A$ = string array to query for the number of columns.

Returns

A 0 if the variable is not an array. Returns 1 if the array has only one dimension. Returns the size of the second dimension if the array has two dimensions.

Example

This example shows how to determine the number of elements in a one dimensional string array.

```
10 DECLARE STRING NAMES$(3)
20 LET NAMES$(1) = "Fred"
30 LET NAMES$(2) = "Wilma"
40 LET NAMES$(3) = "Barney"
50 REDIM NAMES$(4) ! Make room for Betty
60 LET NAMES$(4) = "Betty"
70 LET NUMOFCOLS = COLUMNSIZE(NAMES$)
80 PRINT NUMOFCOLS
```

Example

This example shows how to determine the number of columns in a two dimensional numeric array.

```
10 DECLARE NUMERIC SQROFTWOLOOKUP(3,2)
20 LET SQROFTWOLOOKUP (1,1) = 1
30 LET SQROFTWOLOOKUP (1,2) = 2
40 LET SQROFTWOLOOKUP (2,1) = 2
50 LET SQROFTWOLOOKUP (2,2) = 4
60 LET SQROFTWOLOOKUP (3,1) = 3
70 LET SQROFTWOLOOKUP (3,2) = 8
80 LET COLCNT = COLUMNSIZE(SQROFTWOLOOKUP)
90 PRINT COLCNT
```

FIND

This function will find an element of a string array that contains an identified search string.

Format

FIND(A$, B$)

FIND(A$, B$, START)

FIND(A$, COLUMN, B$)

FIND(A$, COLUMN, B$, START)

Parameters

- A$ = string array to search for B$.
- B$ = string to search for within A$.
- START = index within a single dimensional array, or row for a two dimensional array, to start the search.
- COLUMN = column to isolate search to in a two dimensional array. This must be supplied if A$ is a two dimensional array.

Returns

Returns a 0 if B$ is not found or if there was an error. Otherwise, returns the index that contains the first occurrence of the string B$ (the element index for one dimensional arrays, the row for two dimensional arrays).

Example

This example shows how to find a string in a one dimensional array.

```
10 DECLARE STRING NAMES$(4)
20 LET NAMES$(1) = "Fred"
30 LET NAMES$(2) = "Wilma"
40 LET NAMES$(3) = "Barney"
50 LET NAMES$(4) = "Betty"
60 LET BARNEYIX = FIND(NAMES$, "Bar")
70 PRINT "Found Barney in element "; STR$(BARNEYIX)
```

Example

This example shows how to find a string that occurs more than once in a two dimensional array.

```
10  DECLARE STRING CLOTHING$(5,2)
20  LET TYPECOL     = 1
30  LET MATERIALCOL = 2
40  LET CLOTHING$(1,1) = "Gloves"
50  LET CLOTHING$(1,2) = "Knit"
60  LET CLOTHING$(2,1) = "Pants"
70  LET CLOTHING$(2,2) = "Cotton"
80  LET CLOTHING$(3,1) = "Gloves"
90  LET CLOTHING$(3,2) = "Leather"
100 LET CLOTHING$(4,2) = "Shirts"
110 LET CLOTHING$(4,2) = "Polyester"
120 LET CLOTHING$(5,2) = "Pants"
130 LET CLOTHING$(5,2) = "Denim"
140 LET GLOVEIX = 1
150 DO
160 LET GLOVEIX = FIND(CLOTHING$, TYPECOL, "Gloves", GLOVEIX)
170 IF NOT GLOVEIX = 0 THEN
180 PRINT CLOTHING$(GLOVEIX, MATERIALCOL), "gloves are available"
190 LET GLOVEIX = GLOVEIX + 1
200 END IF
210 LOOP WHILE NOT GLOVEIX = 0
```

Comments

COLUMN must be greater than 0.

If START is given, it must be greater than 0.

FIND will match the first occurrence of B$, even if it is a substring of a string within the A$ array. For example, "Coat" will be found in both locations 1 and 4.

```
5 DECLARE STRING A$(5)
10 LET A$(1) = "Over Coat"
20 LET A$(2) = "Hat"
30 LET A$(3) = "Jacket"
40 LET A$(4) = "Coat"
50 LET A$(5) = "Boots"
```

If an exact match is needed, FIND should be called until 0 is returned or the item is found and confirmed. To confirm, check the item against the expected item, it should match exactly. See CSV Program on page 587 for an example showing how to do this.

Time and Date Functions

This section describes the functions to access the real time clock option. Here is a quick list of these commands:

DATE$
Returns the date as a string

TIME$
Returns the current time in a string.

DATE
Gets the current date as a number.

TIME
Gets the current time as a number.

DATE$

This function returns the date as a string.

Format
DATE$

Parameters
N/A

Returns
The current date in string form YYYYMMDD. If the Real-Time Clock is not installed, an empty string is returned.

Example
This is an example of how to use the DATE$ command:

```
10 PRINT DATE$
RUN
```

The result, assuming the date is January 1, 2003 is:

```
20030101
```

Example
This is another example of the DATE$ command used with the sub-string operator to get the day of the month:

```
10 LET A$=DATE$(7:8)
20 IF A$ <> DATE$(7:8)
30 LET A$=DATE$(7:8)
40 IF A$="01"
50 PRINT "IT IS THE FIRST OF THE MONTH"
60 END IF
70 END IF
80 SLEEP 100
90 GOTO 20
```

Comments
None

TIME$

This function returns the current time in a string.

Format

`TIME$`

Parameters

N/A

Returns

This function returns the time of day in format HH:MM:SS (hours:minutes:seconds). If the Real-Time Clock is not installed, an empty string is returned.

Example

This is an example of how to use the TIME$ command:

```
10 PRINT TIME$
RUN
10:00:00
```

DATE

This function gets the current date as a number.

Format
DATE

Parameters
N/A

Returns
This function returns the current date in YYYYDDD format, where YYYY is the year and DDD is the number of days since the beginning of the year. If the Real-Time Clock is not installed, 0 is returned.

Example
This example assumes the current date is January 1, 2003:

```
10 PRINT DATE
RUN
2003001
```

TIME

This function gets the current time as a number.

Format

`TIME`

Parameters

N/A

Returns

This function returns the time past midnight (2400h) in seconds. If the Real-Time Clock is not installed, 0 is returned.

Example

This is an example of how to use the TIME command [assuming the time is one minute past midnight]:

```
10 PRINT TIME
RUN
60
```

Set/Get/Do Interactions

The printer's Set/Get/Do data can be directly accessed via ZBI. For a complete listing of what can be accessed, type the following:

```
! U1 getvar "allcv"
```

Here's a quick list of these commands:

SETVAR
Allows the direct setting of printer parameters.

GETVAR$
Retrieves printer parameters.

SETVAR

SETVAR allows the direct setting of printer parameters.

ZBI 2.0↑

Format

SETVAR (PARAM$, VALUE$)

Parameters

PARAM$ = The printer parameter to set.

VALUE$ = the value to set

Returns

Parameter dependent.

Example

This is an example of the SETVAR command:

```
AUTONUM 1,1
LET OUTSTR$ = "Processing"
LET LOOPCTR = 200
LET TIMER5 = 17
LET TMP = REGISTEREVENT(TIMER5, 0, 1000)
DO WHILE LOOPCTR > 0
LET EVT = HANDLEEVENT()
IF EVT = TIMER5 THEN
LET A = SETVAR("device.frontpanel.line2",OUTSTR$)
LET OUTSTR$ = OUTSTR$ & "."
IF LEN(OUTSTR$) >16 THEN
LET OUTSTR$ = "Processing"
END IF
END IF
LET LOOPCTR = LOOPCTR - 1
SLEEP 1
LOOP
LET TMP = UNREGISTEREVENT(TIMER5)
LET A = SETVAR("device.frontpanel.line2","")
END
```

Comments

None

GETVAR$

This function retrieves printer parameters.

Format

GETVAR$ (PARAM$)

Parameters

PARAM$ = the printer parameter to get.

Returns

The value of the parameter. Refer to the SGD commands for specific parameters.

Example

This is an example of the GETVAR$ command:

```
AUTONUM 1,1
LET SGDCOUNT = 7
DECLARE STRING SGDQUERY$(2,SGDCOUNT)
LET SGDQUERY$(1,1) = "appl.name"
LET SGDQUERY$(1,2) = "device.printhead.serialnum"
LET SGDQUERY$(1,3) = "internal_wired.ip.addr"
LET SGDQUERY$(1,4) = "internal_wired.ip.netmask"
LET SGDQUERY$(1,5) = "internal_wired.ip.gateway"
LET SGDQUERY$(1,6) = "internal_wired.ip.port"
LET SGDQUERY$(1,7) = "internal_wired.mac_addr"
FOR I = 1 TO SGDCOUNT
LET SGDQUERY$(2,I) = GETVAR$(SGDQUERY$(1,I))
NEXT I
OPEN #1: NAME "ZPL"
PRINT #1: "^XA"
FOR I = 1 TO SGDCOUNT
PRINT #1: "^FO50,";50*I;"^A0N,25,25^FD";SGDQUERY$(1,I);"=";
PRINT #1: SGDQUERY$(2,I);"^FS"
NEXT I
PRINT #1: "^XZ"
```

Comments

None

Example Programs

The next section provides example programs of common tasks using ZBI commands.

These programs are also available for download at: www.zebra.com/zbi

Array Program

This program prompts a user to enter first a name; when it is entered, it is added to an array of all names entered. The user is them prompted to enter an address, which is then added to an array of all addresses entered. After the user enters a total or five names and addresses, the program uses the arrays to print the entered data on five labels.

Example

This is an example of Array

```
1 rem ********************************************************
1 rem Zebra Technologies ZBI Sample Program
1 rem
1 rem Professional programming services are available. Please contact
1 rem ZBI-Experts@zebra.com for more information.
1 rem
1 rem This is an example of using arrays to store and use data within 1 rem
 ZBI.
1 rem ********************************************************
1 rem close all ports except for the console
1 rem********************************************************
10 for i = 1 to 9 step 1
20    close #i
30    next i
1 rem ********************************************************
1 rem open a port to the print engine
1 rem ********************************************************
40 open #1: name "ZPL"
1 rem ********************************************************
1 rem create string arrays five elements in size to hold names and
1 rem addresses
1 rem ********************************************************
50 declare string name$(5)
60 declare string address$(5)
1 rem ********************************************************
1 rem infinite loop to put name and address data from console into
1 rem arrays
1 rem ********************************************************
70 do
80 for i = 1 to 5 step 1
90    print "PLEASE ENTER THE NAME"
1 rem ********************************************************
1 rem get data from console; input command looks for CRLF
1 rem ********************************************************
100    input name$(i)
1 rem ********************************************************
```

```
1 rem if the user inputs end or END, the program will end
1 rem ******************************************************
110    if name$(i) = "END" or name$(i) = "end" then
120       end
130    end if
140    print "PLEASE ENTER THE ADDRESS"
150    input address$(i)
160    if address$(i) = "END" or address$(i) = "end" then
170       end
180    end if
190 next i
200 for index = 1 to 5 step 1 ! For loop To Print data no label
1 rem ******************************************************
1 rem semicolon at the end prints with no CRLF
1 rem ******************************************************
210    print #1: "^XA^FO30,30^A0N,30,30^FD"&NAME$(INDEX)&"^FS";
1 rem ******************************************************
1 rem ampersand used to concatenate data into strings
1 rem ******************************************************
220    print #1: "^FO30,70^A0N,30,30^FD"&ADDRESS$(INDEX)&"^FS^XZ"
230 next index
240 loop ! loops back To Line 60
250 end
```

CSV Program

The following program will initialize and then execute continuously, repeating the same series of operations; process events, read input from the serial port, write any processed data out to the ZPL port, and then process the data read from the serial port.

The program first loads the CSV database E:PRODUCTS.CSV (in PROGRAMINIT subroutine). Then, data read from the serial port is compared against the first column in the database. If an entry is found in the first column of a row (in FINDITEM subroutine), the data for the respective row is inserted into the ZPL format E:PRICELBL.ZPL and printed on a label.

Example

This is an example of a CVS program.

```
1 REM SUBROUTINES BELOW....
2 REM
3 REM
 ************************************************************************
4 REM                MAIN LOOP - DO NOT MODIFY
5 REM
 ************************************************************************
6 REM
7 GOSUB PROGRAMINIT
8 DO WHILE 1 = 1
9 GOSUB PROCESSEVENTS
10 GOSUB GETINPUT
11 GOSUB WRITEOUTPUT
12 GOSUB PROCESSDATA
```

```
13 LOOP
14 REM SUBROUTINES BELOW....
15 REM
16 REM
 ************************************************************************
17 REM                Program Init
18 REM
 ************************************************************************
19 REM
20 SUB PROGRAMINIT
21 LET INPORT = 1
22 LET OUTPORT = 2
23 LET ENDLINE$ = CHR$ ( 13 ) & CHR$ ( 10 )
24 OPEN # INPORT : NAME "SER"
25 OPEN # OUTPORT : NAME "ZPL"
26 DECLARE STRING DATABASE$ ( 1 , 1 )
27 LET COLUMNCOUNT = CSVLOAD ( DATABASE$ , "E:PRODUCTS.CSV" )
28 LET OUTDATA$ = "TABLE WITH " & STR$ ( COLUMNCOUNT ) & " COLUMNS LOADED" &
 ENDLINE$
29 RETURN
30 REM
31 REM
 ************************************************************************
32 REM                Process Events
33 REM
 ************************************************************************
34 REM
35 SUB PROCESSEVENTS
36 RETURN
37 REM
38 REM
 ************************************************************************
39 REM                Get Input
40 REM
41 REM Writes All Data from the serial port to the string INDATA$
42 REM
 ************************************************************************
43 REM
44 SUB GETINPUT
45 IF LEN ( INDATA$ ) < 5000 THEN
46 LET INCOUNT = READ ( INPORT , A$ , 1024 )
47 LET INDATA$ = INDATA$ & A$
48 END IF
49 RETURN
50 REM
51 REM
 ************************************************************************
52 REM                Write Output
53 REM
54 REM Writes All Data from the string OUTDATA$ to the ZPL Port
55 REM
 ************************************************************************
56 REM
57 SUB WRITEOUTPUT
```

```
58 LET OUTCOUNT = WRITE ( OUTPORT , OUTDATA$ , LEN ( OUTDATA$ ) )
59 IF OUTCOUNT > 0 THEN
60 LET OUTDATA$ ( 1 : OUTCOUNT ) = ""
61 END IF
62 RETURN
63 REM
64 REM
 ************************************************************************
65 REM                Process Data
66 REM
67 REM Parse the data in the string INDATA$ and write output to OUTDATA$
68 REM
 ************************************************************************
69 REM
70 SUB PROCESSDATA
71 IF LEN ( OUTDATA$ ) > 1000 THEN
72 RETURN
73 END IF
74 REM REMOVE ALL LINE FEEDS
75 DO
76 LET LOC = POS ( INDATA$ , CHR$ ( 10 ) )
77 LET INDATA$ ( LOC : LOC ) = ""
78 LOOP WHILE LOC > 0
79 REM COMPLETED LINE FEED REMOVAL
80 LET LOC = POS ( INDATA$ , CHR$ ( 13 ) ) ! Line ends with CR
81 IF LOC > 0 THEN
82 LET INLINE$ = INDATA$ ( 1 : LOC - 1 )
83 LET INDATA$ ( 1 : LOC ) = ""
84 GOSUB FINDITEM
85 IF ROW > 0 THEN
86 LET OUTDATA$ = OUTDATA$ & "^XA^XFE:PRICELBL.ZPL^FS" & ENDLINE$
87 LET OUTDATA$ = OUTDATA$ & "^FN1^FD" & DATABASE$ ( ROW , 1 ) & "^FS" &
 ENDLINE$
88 LET OUTDATA$ = OUTDATA$ & "^FN2^FD" & DATABASE$ ( ROW , 2 ) & "^FS" &
 ENDLINE$
89 LET OUTDATA$ = OUTDATA$ & "^FN3^FD" & DATABASE$ ( ROW , 3 ) & "^FS^XZ" &
 ENDLINE$
90 END IF
91 END IF
92 RETURN
93 REM
94 REM
 ************************************************************************
95 REM                Find Item
96 REM
97 REM Search the first column of the database for the exact item requested
98 REM
 ************************************************************************
99 REM
100 SUB FINDITEM
101 LET ROW = 0
102 LET EXPECTED$ = INLINE$
103 DO
104 LET FOUNDENTRY$ = ""
```

```
105 LET ROW = FIND ( DATABASE$ , 1 , EXPECTED$ , ROW + 1 )
106 IF ROW <> 0 THEN
107 LET FOUNDENTRY$ = DATABASE$ ( ROW , 1 )
108 END IF
109 LOOP WHILE ( ROW <> 0 AND FOUNDENTRY$ <> EXPECTED$ )
110 RETURN
```

DPI Conversion Program

This program converts a ZPL format being sent to the printer on the parallel port to 300 dpi (dots per inch) from 200 dpi (dots per inch). This is done by searching for and extracting ZPL commands with resolution-dependent arguments and scaling the arguments for a 300 dpi printer.

Example

This is an example of dpi conversion:

```
1 rem ************************************************
1 rem Zebra Technologies ZBI Sample Program
1 rem
1 rem Professional programming services are available. Please contact
1 rem ZBI-Experts@zebra.com for more information.
1 rem
1 rem This is an example of converting a printer from 200 dpi (dots
1 rem per inch
1 rem to 300 dpi. This example covers only some of the ZPL commands
1 rem that
1 rem could be affected by converting from 200 to 300 dpi printing.
1 rem ************************************************
1 rem open the ports for input and output
1 rem ************************************************
10 close #1
20 close #2
30 open #1 : name "PAR"
40 open #2 : name "ZPL"
1 rem ************************************************
1 rem create an array with the search parameters
1 rem ************************************************
50 declare string find$(20)
60 let find$(1) = "^FO"
70 let find$(2) = "^A0"
80 let find$(3) = "^GB"
90 let find$(4) = "^XZ"
100 let find$(5) = "^A@"
110 let find$(6) = "^LL"
120 let find$(7) = "^LH"
130 let find$(8) = "FO"
140 let find$(9) = "A0"
150 let find$(10) = "GB"
160 let find$(11) = "XZ"
170 let find$(12) = "A@"
180 let find$(14) = "LH"
190 let find$(15) = "^BY"
```

```
200 let find$(16) = "BY"
210 let find$(17) = "^B3"
220 let find$(18) = "B3"
1 rem ******************************************************
1 rem search for the parameters
1 rem ******************************************************
300 do
310    let in$ = searchto$(1, find$, 2)
1 rem ******************************************************
1 rem once a parameter is found, determine how to handle it
1 rem ******************************************************
320    if in$ = "^FO" or in$ = "FO" then
330       gosub 520
340    else if in$ = "^LH" or in$ = "LH" then
350       gosub 520
360    else if in$ = "^A0" or in$ = "A0" then
370       gosub 700
380    else if in$ = "^A@" or in$ = "A@" then
390       gosub 700
400    else if in$ = "^GB" or in$ = "GB" then
410       gosub 1100
420    else if in$ = "^LL" then
430       gosub 1300
440    else if in$ = "^BY" or in$ = "BY" then
450       gosub 1400
460    else if in$ = "^B3" or in$ = "B3" then
470       gosub 1600
480    else if in$ = "^XZ" then
490       print #2: in$;
500    end if
510 loop
1 rem ******************************************************
1 rem convert the ^FO and ^LH commands from 200 to 300 dpi
1 rem ******************************************************
520 inbyte #1: a$
530 let a = ord(a$)
540 if a >= 65 then
550    print #2: in$&a$;
560    goto 660
570 end if
580 let x$ = extract$(1, "", ", ")
590 let x2$ = a$&x$
600 let y$ = extract$(1, "", "^")
610 let x = val(x2$)
620 let y = val(y$)
630 let x2 = (x/2)+x
640 let y2 = (y/2)+y
650 print #2: in$; x2; ","; y2; "^";
660 return
1 rem ******************************************************
1 rem convert the ^A0 and ^A@ commands from 200 to 300 dpi
1 rem ******************************************************
700 inbyte #1: a$
710 let a = ord(a$)
```

```
720 let b = 0
730 let c = 0
740 if a >= 65 then
750    print #2: in$&a$; ",";
760    let b = 1
770 end if
780 inbyte #1: a$
790 let h$ = extract$(1, "", ",")
800 if in$ = "^A@" or in$ = "A@" then
810    let c = 1
820    let w$ = extract$(1, "", ",")
830    let m$ = extract$(1, "", "^")
840 else
850    let w$ = extract$(1, "", "^")
860 end if
870 let h = val(h$)
880 let w = val(w$)
900 let h2 = (h/2) + h
910 let w2 = (w/2) + w
920 if b = 1 then
930    print #2: h2; ","; w2;
940 else
950    print #2: in$&"N,"; h2; ","; w2;
960 end if
970 if c = 1 then
980    print #2: ","; m$;
990 end if
1000 print #2: "^";
1010 return
1 rem ********************************************************
1 rem convert the ^GB command from 200 to 300 dpi
1 rem ********************************************************
1020 let w$ = extract$(1, "", ",")
1030 let h$ = extract$(1, "", ",")
1040 let t$ = extract$(1, "", "^")
1050 let h = val(h$)
1060 let w = val(w$)
1070 let t = val(t$)
1080 let h2 = (h/2)+ h
1090 let w2 = (w/2)+ w
1100 let t2 = (t/2)+ t
1110 print #2: in$; w2; ","; h2; ","; t2; "^";
1120 return
1 rem ********************************************************
1 rem convert the ^LL command from 200 to 300 dpi
1 rem ********************************************************
1300 let l$ = extract$(1, "", "^")
1310 let l = VAL(l$)
1320 let l2 = (l/2) + l
1330 print #2: in$; l2; "^";
1340 return
1 rem ********************************************************
1 rem convert the ^BY command from 200 to 300 dpi
1 rem ********************************************************
```

```
1400 inbyte #1: a$
1410 let a = ord(a$)
1420 if a >= 48 and a <= 57 then
1460    let x$ = extract$(1, "", ", ")
1470    let x2$ = a$&x$
1480    let x = val(x2$)
1490    let x2 = (x/2) + x
1500       if x2 > 10 then
1510       let x2 = 10
1520       end if
1530    print #2: in$; x2; ",";
1540 else
1550    print #2: in$; a$;
1560 end if
1570 return
1 rem ******************************************************
1 rem convert the ^B3 command from 200 to 300 dpi
1 rem ******************************************************
1600 let o$ = extract$(1, "", ", ")
1610 let e$ = extract$(1, "", ", ")
1620 let h$ = extract$(1, "", ", ")
1630 let h = val(h$)
1640 let h2 = (h/2) + h
1650 print #2: in$; o$; ","; e$; ","; h2; ",";
1660 return
```

Email Program

This program sends a simple email message to user@domain.com, assuming a valid email server is set up by identifying the SMTP server on the print server. In order to write email via ZBI, the port written to must be named "EML".

Example

This is an example of email

```
1 rem ************************************************
1 rem Zebra Technologies ZBI Sample Program
1 rem
1 rem Professional programming services are available. Please contact
1 rem ZBI-Experts@zebra.com for more information.
1 rem
1 rem This is an example of connecting to an email server to send
1 rem email.
1 rem ************************************************
1 rem EOT$ is the special character used to denote end of transmission
1 rem ************************************************
5 let EOT$ = chr$(4)
1 rem ************************************************
1 rem Open a connection to the email port; if there is an error, try
1 rem again
1 rem ************************************************
10 open #1: name "EML"
```

```
15 on error goto 10
1 rem ************************************************
1 rem Specify address to send message to, signal end of recipients
1 rem with EOT$
1 rem Note: To send to multiple addressees, separate addressees with
1 rem a space
1 rem ************************************************
20 print #1: "user@domain.com";EOT$;
1 rem ************************************************
1 rem Fill in the message information
1 rem ************************************************
30 print #1: "From: Sample User"
40 print #1: "To: Recipient"
50 print #1: "Subject: This is a test"
60 print #1: ""
70 print #1: "Hello!"
80 print #1: i
1 rem ************************************************
1 rem Terminate message
1 rem ************************************************
90 print #1: "";EOT$
1 rem ************************************************
1 rem Close the port, since each open port is only good for sending
1 rem one message
1 rem ************************************************
100 close #1
110 sleep 2
120 let i = i + 1
130 goto 10
```

Extraction 1 Program

This program finds and stores data of interest, which in this case is found in a format after the string "DATA = ". The extract command is used to get the data from the input stream, and it is inserted into a simple ZPL format to be printed.

Example

This is an example of Extraction 1.

```
1 rem ************************************************
1 rem Zebra Technologies ZBI Sample Program
1 rem
1 rem Professional programming services are available. Please contact
1 rem ZBI-Experts@zebra.com for more information.
1 rem
1 rem This is an example of using ZBI for data extraction.
1 rem There are two methods for doing extraction; this example shows
1 rem data extraction using a string.
1 rem
1 rem The data to extract is as follows:
1 rem START
1 rem DATA = "hello":
```

```
1 rem DATA = "goodbye":
1 rem END
1 rem ************************************************
1 rem close ports except console, open channels to parallel and serial
1 rem ports
1 rem ************************************************
05 for i = 1 to 9 step 1
10    close #i
20 next i
30 open #1: name "PAR"
40 open #2: name "ZPL"
1 rem ************************************************
1 rem create string array to hold data
1 rem ************************************************
50 declare string format$(3)
60 let format$(1) = "START     "
70 let format$(2) = "END     "
80 let format$(3) = "DATA"
1 rem ************************************************
1 rem main program; look for "START" keyword, if found print ^XA to ZPL port
1 rem ************************************************
90 do
100    let begin$ = searchto$(1,format$,2)
110    if begin$ = "START" then
120    print #2: "^XA";
1 rem ************************************************
1 rem if "DATA" keyword is found, get two data strings
1 rem ************************************************
130    else if begin$ = "DATA" then
140       input #1: data_string1$
150       input #1: data_string2$
1 rem ************************************************
1 rem get data from between quotes and print to ZPL port with formatting
1 rem ************************************************
160       let extracted_data1$ = extract$(data_string1$,"""","""")
170       let extracted_data2$ = extract$(data_string2$,"""","""")
180       print #2:"^FO30,30^A0N,30,30^FD"&extracted_data1$&"^FS";
190       print #2:"^FO30,70^A0N,30,30^FD"&extracted_data2$&"^FS";
200    else if begin$ = "END" then
210       print #2: "^XZ        "
220    end if
230 loop
```

Extraction 2 Program

This program finds and stores data of interest, which in this case is found in a format after the string "DATA = ". The input command is used to get the data from the input stream, and it is inserted into a simple ZPL format to be printed.

Example

This is an example of Extraction 2.

```
1 rem*****************************************************
1 rem Zebra Technologies ZBI Sample Program
1 rem
1 rem Professional programming services are available. Please contact
1 rem ZBI-Experts@zebra.com for more information.
1 rem
1 rem This is an example of using ZBI for data extraction.
1 rem There are two methods for doing extraction; this example shows
1 rem data extraction from the port directly.
1 rem
1 rem The data to extract is as follows:
1 rem START
1 rem DATA = "hello":
1 rem DATA = "goodbye":
1 rem END
1 rem*****************************************************
1 rem close ports except console, open channels to parallel and serial ports
1 rem*****************************************************
05 for i = 1 to 9 step 1
10    close #i
20 next i
30 open #1: name "PAR"
40 open #2: name "ZPL"
1 rem*****************************************************
1 rem create string array to hold data
1 rem*******quotes and print to ZPL port with formatting
1 rem*****************************************************

50 declare string format$(3)
60 let format$(1) = "START"
70 let format$(2) = "END"
80 let format$(3) = "DATA"
1 rem*****************************************************
1 rem main program; look for "START" keyword, if found print ^XA to ZPL port
1 rem*****************************************************

90 do
100    let begin$ = searchto$(1, format$, 2)
110    if begin$ = "START" then
120       print #2: "^XA";
1 rem*****************************************************
1 rem if "DATA" keyword is found, get two data strings
1 rem*****************************************************
130    else if begin$ = "DATA" then
1 rem************************************************
1 rem get data from between q
140       let extracted_data1$ = extract$(1,"""","""")
150       input #1: junk$
170       let extracted_data2$ = extract$(1,"""","""")
180       print #2:"^FO30,30^A0N,30,30^FD" &extracted_data1$& "^FS";
190       print #2:"^FO30,70^A0N,30,30^FD" &extracted_data2$& "^FS";
200    else if begin$ = "END" then
210       print #2: "^XZ"
```

```
220    end if
230 loop
```

Front Panel Control

This example shows how to intercept front panel button presses and write to the display to create a simple menu. The buttons used in this demo are set up for a Z4M/Z6M, ZM400/ZM600, or RZ400/RZ600. This could be reconfigured to work with any other printer.

Example

This is an example of front panel control.

```
1 REM This example shows how to override the functionality of the feed key
1 REM and use the front panel display to show a option list
AUTONUM 1,1
REM CLOSE ALL
DECLARE STRING OPTIONS$(5)
FOR I = 1 TO 5
LET OPTIONS$(I) = "Option " & STR$(I)
NEXT I
LET ZPLPORT = 1
OPEN #ZPLPORT: NAME "ZPL"
LET FEEDKEY   = 3
LET SELECTKEY = 10
LET PLUSKEY   = 6
LET MINUSKEY  = 7
LET EXITKEY   = 9
LET TMP = REGISTEREVENT(FEEDKEY, 0, 1)
SUB NORMALLOOP
DO WHILE 1 = 1
LET EVT = HANDLEEVENT()
IF EVT = FEEDKEY THEN
LET INDEX = 1
GOSUB REGISTERKEYS
GOSUB SHOWMENU
GOTO FEEDLOOP
END IF
SLEEP 1
LOOP
SUB FEEDLOOP
DO WHILE 1 = 1
LET EVT = HANDLEEVENT()
IF EVT = FEEDKEY THEN
GOSUB RELEASEKEYS
GOSUB HIDEMENU
GOTO NORMALLOOP
ELSE IF EVT = SELECTKEY THEN
GOSUB HANDLEOPTION
ELSE IF EVT = PLUSKEY THEN
LET INDEX = INDEX + 1
IF INDEX > 5 THEN
LET INDEX = 1
```

```
END IF

GOSUB SHOWMENU
ELSE IF EVT = MINUSKEY THEN
LET INDEX = INDEX - 1
IF INDEX < 1 THEN
LET INDEX = 5
END IF
GOSUB SHOWMENU
ELSE IF EVT = EXITKEY THEN
GOSUB RELEASEKEYS
GOSUB HIDEMENU
GOTO NORMALLOOP
END IF
SLEEP 1
LOOP
REM ******** SUBROUTINE SHOWMENU ***
SUB SHOWMENU
LET LINE1$ = "FEED DISPLAY"
LET LINE2$ = OPTIONS$(INDEX)
GOSUB UPDATEDISPLAY
RETURN
REM ******** SUBROUTINE HIDEMENU ***
SUB HIDEMENU
LET LINE1$ = ""
LET LINE2$ = ""
GOSUB UPDATEDISPLAY
RETURN
SUB UPDATEDISPLAY
LET A = SETVAR("device.frontpanel.line1",LINE1$)
LET A = SETVAR("device.frontpanel.line2",LINE2$)
RETURN
SUB REGISTERKEYS
LET TMP = REGISTEREVENT(SELECTKEY, 0, 1)
LET TMP = REGISTEREVENT(PLUSKEY, 0, 1)
LET TMP = REGISTEREVENT(MINUSKEY, 0, 1)
LET TMP = REGISTEREVENT(EXITKEY, 0, 1)
RETURN
SUB RELEASEKEYS
LET TMP = UNREGISTEREVENT(SELECTKEY)
LET TMP = UNREGISTEREVENT(PLUSKEY)
LET TMP = UNREGISTEREVENT(MINUSKEY)
LET TMP = UNREGISTEREVENT(EXITKEY)
RETURN
SUB HANDLEOPTION
PRINT #ZPLPORT: "^XA^FO100,100^A0N,100,100^FD"; OPTIONS$(INDEX);"^XZ"
RETURN
```

Recall Program

This program searches for a ZPL format named "FORMAT.ZPL" that is already saved in printer memory. If the format is found, a number within the format is extracted and shown on the console. The user is then prompted to enter a new number, which is then substituted into the format.

Example

This is an example of Recall.zpl

```
1 rem ********************************************************
1 rem Zebra Technologies ZBI Sample Program
1 rem
1 rem Professional programming services are available. Please contact
1 rem ZBI-Experts@zebra.com for more information.
1 rem
1 rem This is an example of recalling a ZPL format and extracting data
1 rem from it.
1 rem ********************************************************
1 rem close ports except console, open ZPL port and declare search
1 rem array
1 rem ********************************************************
10 for i = 1 to 9 step 1 ! Close all ports
20    close #i
30 next i
40 let zplport = 2
50 open #zplport: name "ZPL"
60 declare string search_zpl$(2)
70 let search_zpl$(1) = chr$(03)
80 let search_zpl$(2) = "FORMAT.ZPL"
1 rem ********************************************************
1 rem main program; look for format to recall on printer
1 rem ********************************************************
90 do
100    print #zplport: "^XA^HWE:*.ZPL^FS^XZ"
110       let present = 0
115    let find$ = ""
120    do until find$ = chr$(03)
130       let find$ = searchto$(zplport, search_zpl$)
140       if  find$ = "FORMAT.ZPL" then
150         let present = 1 ! format is present
160       end if
170    loop

1 rem ********************************************************
1 rem if format is not found, create a format and set data value to
1 rem 000
1 rem ********************************************************
180    if present = 0 then
190       print #zplport:"^XA^DFE:FORMAT.ZPL^FS";
200       print #zplport:"^FX000^FS^XZ"
210       let counter$ = "000"
1 rem ********************************************************
```

```
1 rem if format is found, extract the data from ^FX field
1 rem ********************************************************
220    else
230       print #zplport:"^XA^HFE:FORMAT.ZPL^FS^XZ"
240       let stop$ = searchto$(zplport, "^FX")
250       let counter$ = extract$(zplport, "", "^FS")
260       let stop$ = searchto$(zplport, "^XZ")
270    end if
1 rem ********************************************************
1 rem print current data value, prompt user to replace data
1 rem ********************************************************
280    print ""
290    print "Current number in format is " & counter$
300    print "Please enter new number (type EXIT to end) ";
310    input new_counter$
320    if new_counter$ = "EXIT" then
330       print "Program ending"
340       end
350    else
360       print #zplport:"^XA^DFE:FORMAT.ZPL^FS";
370       print #zplport:"^FX" & new_counter$ & "^FS^XZ "
380    end if
390 loop
```

Scale Program

This program reads data from a scale connected to the serial port by sending a "W" to the scale and waiting for a weight to be returned. When the weight is received, it is inserted into a simple label format and printed.

Example

This is an example of Scale

```
1 rem ********************************************************
1 rem Zebra Technologies ZBI Sample Program
1 rem
1 rem Professional programming services are available. Please contact
1 rem ZBI-Experts@zebra.com for more information.
1 rem
1 rem This is an example of using ZBI to read scale data from the
1 rem serial port.
1 rem ********************************************************
1 rem close all ports except console, open channels to parallel and
1 rem serial ports
1 rem ********************************************************
05 for i = 1 to 9 step 1
10    close #i
20    next i
30 open # 2 : name "SER"
40 open # 1 : name "ZPL"
1 rem ********************************************************
1 rem main program; send serial port a 'W' in order to get a weight
```

```
1 rem ******************************************************
50 do
60    do
70       sleep 1   ! sleep so scale is not bombarded with incoming
1 rem data
80       print # 2 : "W" ;   ! semicolon ends sent W without a CRLF
1 rem ******************************************************
1 rem get response from scale; note that input requires a CRLF to be
1 rem entered
1 rem ******************************************************
90       input # 2 : a$
100       if a$ = "EXIT" then ! back door exit - if EXIT is received, ZBI
 ends
110          close # 2
120          print #1: "^XZ"
130          close #1
140          end
150       end if


1 rem ******************************************************
1 rem loop until valid weight is received, then print on label
1 rem ******************************************************
160    loop while pos ( a$ , "000.00" ) = 1 or pos ( a$ , "?" ) = 1
170    print # 1 : "~SD25^XA^FS";
180    print # 1 : "^LH0,0^FS";
190    print # 1 : "^FO56,47^A0N,69,58^FDThis weighs^FS";
1 rem ******************************************************
1 rem print weight on label; & character concatenates strings
1 rem ******************************************************
200    print # 1 : "^FO56,150^A0N,69,58^FD" & A$ & " lbs^FS";
210    print # 1 : "^PQ1,0,0,N";
220    print # 1 : "^XZ"
1 rem ******************************************************
1 rem loop until weight is off scale, then repeat for next item
1 rem weighed
1 rem ******************************************************
230    do
240       print # 2 : "W" ;
250       input # 2 : A$
260    loop until pos(A$ , "000.00") = 1 or pos(A$ , "?") = 1
270 loop
```

About SGD Printer Commands

This chapter provides a high-level overview of printer setting Set / Get / Do (SGD) commands.

.15↑

SGD commands are available in printers with the following firmware versions or later:	
• V66.17.4Z or later • V61.15.xZ or later • V60.16.2Z or later • V60.15.xZ or later • V50.15.xZ or later • V56.15.xZ or later • V53.16.x or later	• V53.15.2Z or later • R53.16.3Z or later • R60.15.8Z or later • R62.15.8Z or later • R63.15.8Z or later • R65.15.8Z or later

IMPORTANT: These are important points to note when using ZPL and SGD commands:

- SGD commands are case-sensitive.
- ZPL and SGD commands should be sent to the printer as separate files.
- Certain settings can be controlled by both ZPL and SGD. Configuration changes made in ZPL can affect configuration changes made in SGD.
- Changes made with one command type (ZPL or SGD) will affect the data returned to the host in response to both ZPL and getvar commands. The command type (ZPL or SGD) that was sent last determines the current setting.
- Some RF cards do not support all of the SGD commands.

IMPORTANT: These are important points to note when using a Zebra G-Series printer:

- You can send instructions to the printer using multiple programming languages: EPL, ZPL, or SGD. EPL and ZPL commands configure the printer, print labels, and get device status information. SGD commands set and get configuration details. These three languages can be used without the need to send the printer instructions to switch from one language to another.
- EPL, ZPL, and SGD commands must be sent to the printer as separate files. They cannot be used together in one format, or set of commands. For example, if you send a series of SGD

commands to the printer and they are followed by a printable format, this needs to be done using separate files.

Overview

This section describes how and why to use the Set/Get/Do (SGD) commands. It also provides an example of a typical command structure.

SGD commands are commands that allow you to configure all printers with firmware versions V60.15.xZ, V50.15.xZ, V61.15.xZ, V56.15.xZ, V53.15.xZ, or later. The printer performs the specified function immediately after receiving the command. The commands are:

- setvar
- getvar
- do

IMPORTANT: SSGD commands must be terminated by a carriage return or a space and line feed, and the command, attributes, and values must be specified in lower case.

setvar Command

`Setvar` commands:

- are used to configure printer settings to specific values by setting them in the printer
- must be terminated by a space character or a CR/ LF (0x0D, 0x0A)

getvar Command

`Getvar` commands:

- are used to get the current value of the printer settings
- must be terminated by a space character or CR/LF (0x0D, 0x0A)

The printer responds with the printer setting of “?” if:

- the printer setting does not exist (usually due to incorrect spelling of the printer setting)
- it has not been configured yet

do Command

`Do` commands:

- are used to instruct the printer to perform predefined actions
- must be terminated by a space character or a CR/LF (0x0D, 0x0A)

Some `Do` commands require additional settings which must be enclosed in double quotes.

Command Structure

It is important to understand the structure of the command and its components. A command structure illustration is provided for each command in this guide.

This is an example of a command structure illustration:

! U1 setvar	"ip.addr"	"value"
1	2	3

1. Command—always preceded with an exclamation point (!) and must be specified in lower case. A space resides between the ! and U1 and between U1 and the command (setvar or getvar).
2. Attribute—always in double quotes and must be specified in lower case.
3. Chosen value—always in double quotes. Only applicable for setvar and do.

This command must be terminated by a space character or a CR/ LF (0x0D, 0x0A).

How to Send Multiple SGD Commands

For any getvar, setvar, or do command, if you issue the syntax without the "1" and use the END command followed by a space, multiple SGD commands are sent simultaneously.

This syntax shows how you can send multiple getvar commands:

```
! U getvar "ip.telnet.enable"
getvar "ip.dhcp.enable" getvar "ip.dhcp.cid_prefix"
END
```

1. The command portion of the string does not use the "1" after the "! U".

```
! U getvar "ip.telnet.enable"
```

2. Commands issued after the first command do not require the "! U".

```
getvar "ip.dhcp.enable" getvar "ip.dhcp.cid_prefix"
```

3. The string of commands is terminated by the word "END" with a space after the word, and by a carriage return/ line feed.

```
END
```

JSON (JavaScript Object Notation)

JSON (JavaScript Object Notation) is an open standard format that uses human- and machine-readable text for device management. It transmits data objects consisting of elements as attribute–value pairs.

You can use use JSON as an alternative to using the SGD (Set-Get-Do) mechanism when reading or writing parameters on QLn and iMZ mobile printers. JSON is a popular open standard for exchanging data objects and is well suited to this task.

The main settings channel for JSON is TCP port 9200, but other ports can be used. JSON commands are processed when received. Up to eight connections are allowed, and all connected ports are active, and the JSON commands will work while the printer is printing.

The port used for JSON can be changed or disabled using ip.port_json_config on page 1302.

NOTE: JSON is available on all communications ports, unless `line_print` is enabled, in which case you must use the main TCP JSON port, 9200. If you connect to port 9200, the printer ONLY accepts JSON commands. CPCL, SGD, ZPL, and other command languages are not supported.

Configuring JSON Usage for Communications

All JSON commands should follow the JSON specification for escaping, spacing, etc. All JSON commands are prefixed by `{}`.

Refer to http://www.json.org/ for full details on JSON formatting.

By enclosing a variable's value in curly braces, it indicates that the value is an object. Inside the object, you can declare any number of properties using a `"name": "value"` pairing, separated by colons. Multiple pairings are separated by commas.

Use the SGD variable name in the JSON command structure. To configure JSON usage for communication, refer to the following examples.

Getvar using JSON

To do a `getvar` in SGD you use the format:

```
! U1 getvar "sgd.name"
! U1 getvar "ip.port"
! U1 getvar "device.location"
```

To get a variable value using JSON:

```
{}{"sgd.name":null} returns {"sgd.name":"value"}
{}{"ip.port":null} returns {"ip.port":"9100"}
{}{"device.location":null} returns {"device.location":"my desk"}
```

You can get several values as follows:

```
{}{"device.friendly_name":null,  "device.company_name":null,
 "device.company_contact":null, "device.location":null}
```

The response is:

```
{"device.friendly_name":"XXQLJ120900310",
"device.company_name":"Zebra Technologies",
"device.company_contact":"123-555-1212",
"device.location":"My Desk"}
```

Setvar using JSON

To do a `setvar` in SGD you use the format:

```
! U1 getvar "sgd.name" "value"
! U1 getvar "ip.port" "9200"
! U1 setvar "device.location" "my desk"
```

To set a variable value using JSON:

```
{}{"sgd.name":"value"} sets the variable value to "value"
{}{"ip.port":"1234"} sets the variable value to "1234"
{}{"device.location":"my desk"} sets the variable value to "my desk"
```

NOTE: When you set an SGD value, it will return the value that was set, or the old value if the set failed. If:

```
{}{"sgd.name":"new_value"} fails, the variable value remains "old_value"
```

To set several values at once:

```
{}{"device.friendly_name":"XXQLJ120900310", "device.company_contact":"123#
555#1212", "device.location":"My Desk"}
```

The response is:

```
{"device.friendly_name":"XXQLJ120900310", "device.company_contact":"123#555#
1212", "device.location":"My Desk"}
```

Get an SGD Branch

You can retrieve all branch values by specifying the branch.

```
{}{"bluetooth":null} returns all SGDs in branch and their values.
```

Get an allvalues Report

You can request an `allvalues` report with just the values for all settings with characteristics. This will return all SGDs and their values.

```
{}{"allvalues":
{"ip.port":"6101”, "ip.port_alternate":"9100”, "ip.sgd_json_port":"9200”,
...
}}
```

Get an allconfig Report

You can request an allconfig report using JSON, and it will return all settings with characteristics.

To get all SGDs and their values along with various other information including defaults:

```
{}{"allconfig":null}
```

NOTE: For the "allconfig" response, it will start with {"allconfig":{ and end with }}

If you do an allconfig, you can get the setting attributes for all settings as follows:

```
{"allconfig": {"ip.port":{"value":"6101","type":"integer","range":"0-
 65535","clone":true,
"archive":true,"access":"RW"},
"ip.port_alternate":
{"value":"9100","type":"integer","range":"0-65535","clone":true,
"archive":true,"access":"RW"},
"ip.sgd_json_port":{"value":"9200","type":"integer","range":"0-
 65535","clone":true,
"archive":true,"access":"RW"}, another setting, ... the last setting}}
```

where:

- "value" indicates the current value stored in the setting.
- "type" indicates the type of value. Possible values are integer, enum, bool, string, double, ipv4-address, ipv6- address.
- "range" indicates the range of a setting. For strings this is the range of the string length. For enums it is the possible enum values.
- "clone" indicates if it is safe to store this setting and apply it to another link-os printer.
- "archive" indicates if is safe to store this setting and apply it to same link-os printer at a later time.
- "access" indicates if the setting is RW (read/write), R (read-only), or W (write-only).

If you do an allconfig, you can get the setting attributes for all settings as follows:

```
{}{"allconfig":null}
```

For the values used above it returns these entries:

```
"device.friendly_name":{"value":"XXQLJ120900310","type":"string","range":
"0-17","clone":false,"archive":true,"access":"RW"},
"device.company_contact":{"value":"123-555-1212","type":"string","range":
"0-128","clone":true,"archive":true,"access":"RW"},
"device.location":{"value":"my desk","type":"string","range":"
0-128","clone":true,"archive":true,"access":"RW"},
```

SGD Command Reference

The SGD commands are described in this section of the programming guide.

alerts.add

This command is used to configure the ZebraNet Alert System. It allows Zebra software to add new alerts without having to use the ZPL ^SX command. This allows the software to configure printers that do not have ZPL on them, and it provides the software with a single way to configure alerts. It also allows the software to configure alerts via local ports such as USB and serial.

The format is similar to the ^SX command. It can delete the alert when both the set and clear flags are set to FALSE.

Setvar

This command instructs the printer to add the new alert with the configuration specified in the comma-delimited list.

To configure the ZebraNet Alert system:

```
! U1 setvar "alerts.add" "[condition],[destination],[set],[clear],
 [destination_address],[port],[quelling],[SGD_name]"
```

Parameters

- The alert condition. This can be any of the values returned from `alerts.conditions`.
- The alert destination type. This can be any of the values returned from `alerts.destinations`.
- On Set - Set to Y if the alert should be sent when the event is set.
- On Clear - Set to Y if the alert should be sent when the event is cleared.
- Destination address - applies to TCP, UDP, EMAIL, SNMP, SDK, MQTT, and HTTP POST destination types. The maximum length of this address is 255 characters.
- Port - Applies to TCP and UDP types.
- Quelling - When set to Y it prevents the alert from being sent. N is the default.
- SGD Name - the name of the SGD command to be added. This is valid only when the alert condition is SGD_SET.

Values

Defined via `alerts.conditions`: PAPER OUT, RIBBON OUT, HEAD TOO HOT, HEAD COLD, HEAD OPEN, SUPPLY TOO HOT, RIBBON IN, REWIND, CUTTER JAM, MED, PRINTER PAUSED, PQ JOB COMPLETED, LABEL READY, HEAD ELEMENT BAD, BASIC RUNTIME, BASIC FORCED, POWER ON, CLEAN PRINTHEAD, MEDIA LOW, RIBBON LOW, REPLACE HEAD, BATTERY LOW, RFID ERROR, ALL MESSAGES, COLD START, SGD SET

Default

NA

Do

This command has the same functionality as the `setvar`.

To configure the ZebraNet Alert system:

```
! U1 do "alerts.add" "[condition],[destination],[set],[clear],
[destination_address],[port],[quelling],[SGD_name]"
```

Example

This example shows a `"Paper Out"` alert sent via the serial port, with no destination address specified.

```
! U1 setvar "alerts.add" "PAPER OUT,SERIAL,Y,N,,0,,"
```

alerts.conditions

This command lists the available conditions that can be specified in the first parameter of the `alerts.add` SGD. See `alerts.add` for information on the various parameters.

Getvar

To retrieve the list of available alert conditions for the printer:

```
! U1 getvar "alerts.conditions"
```

Values

PAPER OUT, RIBBON OUT, HEAD TOO HOT, HEAD COLD, HEAD OPEN, SUPPLY TOO HOT, RIBBON IN, REWIND, CUTTER JAM, MED, PRINTER PAUSED, PQ JOB COMPLETED, LABEL READY, HEAD ELEMENT BAD, BASIC RUNTIME, BASIC FORCED, POWER ON, CLEAN PRINTHEAD, MEDIA LOW, RIBBON LOW, REPLACE HEAD, BATTERY LOW, RFID ERROR, ALL MESSAGES, COLD START, SGD SET

Default

""

alerts.configured

This command creates a list of all the alerts that are configured on the printer. The alerts are delimited by the '|' character.

NOTE: Writing to this SGD will clear out the old alerts and set up the new ones.

Setvar

To create the list of alerts configured on the printer:

```
! U1 setvar "alerts.configured" "<a '|' delimited list of configured alerts>"
```

Values

A list of alerts to be set up on the printer. See `alerts.add` for the format of the individual alerts.

Default

`"COLD START,SNMP,Y,N,255.255.255.255,162,N"`

Getvar

To retrieve the currently configured alerts on the printer:

```
! U1 getvar "alerts.configured"
```

alerts.destinations

This command lists the available destinations that can be specified in the first parameter of the `alerts.add` SGD. See the `alerts.add` for information on the various parameters.

Getvar

To return a list of available alert destinations:

```
! U1 getvar "alerts.destinations"
```

Values

`SERIAL, PARALLEL, E-MAIL, TCP, UDP, SNMP, USB, HTTP-POST, BLUETOOTH, SDK, MQTT`

Default

NA

alerts.http.authentication.add

This command allows the user to add a single server/username/password triplet into the list of authentication entries.

When the printer attempts to connect to the URL in the HTTP POST alert, the server may require HTTP authentication (such as digest, basic, DNS, etc.). There may be multiple authentication requests along the route to the destination (for example, a local server first requires HTTP authentication as well as on the remote server). For each HTTP authentication request received while attempting to connect, the printer will enumerate the authentication entries and attempt to satisfy the request with the username/password pair provided for the respective server. The server name in the entry is what determines which username/password pair should be used for which authentication request. Both DNS names and IP addresses are acceptable.

The server, username, and password are separated by a single space (not a tab or other white space character). The server name is the only required field. If no username is supplied, but a password is, there must be two spaces between the server and the password fields. If there is a username but no password, or simply just the servername, no space is required at the end of the entry.

Setvar

To add server/username/password triplet into the list of authentication entries:

```
! U1 setvar "alerts.http.authentication.add" "servername[ username]
[ password]"
```

Values

Maximum string of 2048 characters.

Default

NA

Do

This command has the same settings as the `setvar`.

To add server/username/password triplet into the list of authentication entries:

```
! U1 do "alerts.http.authentication.add" "servername[ username][ password]"
```

Values

Maximum string of 2048 characters.

Default

NA

Example 1

A username and a password is supplied:

```
! U1 setvar "alerts.http.authentication.add" "my.server.lan johndoe password"
```

Example 2

No password is supplied:

```
! U1 setvar "alerts.http.authentication.add" "my.server.lan johndoe"
```

Example 3

No username is supplied (note the double space):

```
! U1 setvar "alerts.http.authentication.add" "my.server.lan password"
```

Example 4

No username or password is supplied:

```
! U1 setvar "alerts.http.authentication.add" "my.server.lan"
```

alerts.http.authentication.entries

This command lists the server names added to the authentication entries list via `alerts.http.authentication.add.`

Only the server names will be shown; the username and passwords will not be shown. The server names are separated by a `\r\n` so that each shows up on its own line and is easier to read.

Getvar

To return the server names added to the authentication entry list:

```
! U1 getvar "alerts.http.authentication.entries"
```

Values

A list of server names.

Default

NA

alerts.http.authentication.remove

This command allows the user to remove a single server/username/password triplet from the list of authentication entries. To remove an entry, only the server name is supplied, and the entire entry will be removed. If an invalid entry is supplied no action is taken.

Note that the list of authentication triplets will be updated (and saved over a reset) but this SGD is just a command and doesn't have state. Therefore the persistent and restore defaults do not apply. The internal list that this command removes from, however, is persistent and defaultable (defaults to an empty list).

Setvar

To remove a server/username/password triplet from the list of authentication entries:

```
! U1 setvar "alerts.http.authentication.remove" "servername"
```

Value

Maximum string of 2048 characters

Default

NA

Do

This command has the same settings as the `setvar`.

To remove a server/username/password triplet from the list of authentication entries:

```
! U1 do "alerts.http.authentication.remove" "servername"
```

Value

```
Maximum string of 2048 characters
```

Default

NA

Example

A username and a password is supplied

```
! U1 setvar "alerts.http.authentication.remove" "my.server.lan"
```

alerts.http.logging.clear

This command clears the weblink alerts log entries. It does not disable logging. Setting this command to any value, including an empty string, will clear the weblink log entries.

Setvar

To clear the weblink alerts log entries:

```
! U1 setvar "alerts.http.logging.clear" "value"
```

Value

Any string value, including an empty string.

Default

NA

Do

To clear the weblink alerts log entries:

```
! U1 do "alerts.http.logging.clear" "value"
```

Values

Any string value, including an empty string.

Default

NA

Example

This example clears the log entries with an empty string value.

```
! U1 setvar "alerts.http.logging.clear" ""
```

alerts.http.logging.entries

This command returns the N number of entries in the http log, where N has a maximum value that is set by `alerts.http.logging.max_entries`.

The alerts http log is a collection of events related to sending HTTP POST messages. The log entries range anywhere from general status to errors that prevented a successful connection. Each log entry contains a timestamp for when it was logged by the system. The newest events will appear at the bottom of the list.

Getvar

To return the number of entries in the HTTP log:

```
! U1 getvar "alerts.http.logging.entries"
```

Values

NA

Default

NA

Example

This example shows the result from `alerts.http.logging.entries`:

```
 [01-03-2013 12:48:59.964] [http] Connected to 10.3.4.58 (10.3.4.58) port 80
[01-03-2013 12:48:59.978] [http] HTTP/1.1 100 Continue
[01-03-2013 12:49:01.999] [http] Closing connection
```

alerts.http.logging.max_entries

This command specifies the maximum number of individual log entries that will be stored in the `alerts.http.logging.entries` command.

NOTE: Changes to this command are immediate and may result in some log entries being lost. If there are N log entries currently in the log, the user sets the max_entires to M, where M is less than N, the oldest (N-M) log entries will be removed.

Setvar

To set the maximum number of log entries that will be stored:

```
! U1 getvar "alerts.http.logging.max_entries" "value"
```

Values

`"0"` - `"10000"`

NOTE: Setting the value to 0 disables logging.

Default

`"0"`

Getvar

To return the setting for the maximum number of log entries that will be stored:

```
! U1 getvar "alerts.http.logging.max_entries"
```

Do

To set the maximum number of log entries that will be stored:

```
! U1 do "alerts.http.logging.max_entries" "value"
```

Values

`"0"` - `"10000"`

NOTE: Setting the value to 0 disables logging.

Default

`"0"`

Example

In this example, `alerts.http.logging.max_entries` is set to 2.

The original log file:

```
[01-03-2013 12:48:59.964] [http] Connected to 10.3.4.58 (10.3.4.58) port 80
[01-03-2013 12:48:59.978] [http] HTTP/1.1 100 Continue
```

```
[01-03-2013 12:49:01.999] [http] Closing connection
```

When `alerts.http.logging.max.entries` is set to `"2"`, the log file is:

```
[01#03#2013 12:48:59.978] [http] HTTP/1.1 100 Continue
[01#03#2013 12:49:01.999] [http] Closing connection
```

alerts.http.proxy

This command assigns the URL of the proxy for any HTTP POST alerts. The proxy server protocol, port, domain, username, and password are all encoded into the URL via the format outlined in RFC2396.

The username and password must avoid the invalid characters listed in RFC2396 (e.g. ':', '@', '/', etc). If an invalid character must be used it needs to be escaped using '%' as described in http://www.ietf.org/rfc/rfc2396.txt.

When the setting is changed, the next HTTP POST alert will use the new value.

Setvar

To assign the proxy URL for HTTP POST alerts:

```
! U1 setvar "alerts.http.proxy" "http://username:password@mydomain.com:3128/"
```

Values

Any valid URL up to 2048 characters

URL format expected: `http://[user:pass@]domain[:port]/[path]`

Default

- The user:pass, port, and path are all optional.
- The default port is 1080.
- The default is to omit the username and password.

Getvar

To retrieve the proxy URL for HTTP POST alerts:

```
! U1 getvar "alerts.http.proxy"
```

Do

To assign the proxy URL for HTTP POST alerts:

```
! U1 do "alerts.http.proxy" "http://username:password@mydomain.com:3128/"
```

Values

Any valid URL up to 2048 characters

URL format expected: `http://[user:pass@]domain[:port]/[path]`

Default

- The user:pass, port, and path are all optional.
- The default port is 1080.
- The default is to omit the username and password.

Example

Examples of how to connect to various proxy servers:

```
http://username:password@mydomain.com:3128/
http://mydomain.com/
```

alerts.send_current_status_alerts

Generates all of the alerts specified in `alerts.configured` for the current status conditions in the printer and sends the alerts to the specified destination. The destination can be one of the types specified from `alerts.destinations`.

Setvar

To send the current status of the printer to the specified destination:

```
! U1 setvar "alerts.send_current_status_alerts" "<value>"
```

where <value> is `[destination],[destination_address],[port].`.

- `[destination]` can be any of the values returned from `alerts.destination`.
- `[destination_address]` applies to TCP, UDP, EMAIL, SNMP, SDK, MQTT, and HTTP POST destination types.
- `[port]` applies to TCP and UDP destination types.

Example

In this example the value of `alerts.configured` is:

```
! U1 setvar "alerts.configured" "COLD START,SNMP,Y,N,255.255.255.255,162,N,|
ALL MESSAGES,MQTT,Y,Y,1,0,N,| HEAD OPEN,Y,N,255.255.255.255,162,N,"
```

Valid matches:

- `"MQTT,1"` matches and would send alerts for all conditions to mqtt connection 1.
- `"SNMP,255.255.255.255,162"` matches and would send a HEAD_OPEN alert if the head was open by SNMP to 255.255.255.255 port 162.

Invalid matches:

- `"MQTT,2"` does not match because the MQTT connection number is different.
- `"SNMP,10.3.1.27,162"` does not match because the destination address is different.

alerts.tracked_settings.clear_log

This command clears the `alerts.tracked_settings.log`. Setting this command to any value, including an empty string, will clear the `tracked_sgds` log entries.

Setvar

To clear the `tracked_sgds` log entries:

```
! U1 setvar "alerts.tracked_settings.clear_log" "value"
```

Values

Any string value, including an empty string.

Default

NA

Do

To clear the `tracked_sgds` log entries:

```
! U1 do "alerts.tracked_settings.clear_log" "value"
```

Values

Any string value, including an empty string.

Default

NA

Example

This example clears the log entries with an empty string value.

```
! U1 setvar "alerts.tracked_settings.clear_log" ""
```

alerts.tracked_settings.log_tracked

This command creates a comma-delimited list of settings for which sets should be logged.

Setvar

To set the list of settings for which sets should be logged:

```
! U1 setvar "alerts.tracked_settings.log_tracked"
 "settings.name1,settings.name2..."
```

Values

Settings with commas between names.

Default

""

Getvar

To return a comma-delimited lists of settings being logged:

```
! U1 getvar "alerts.tracked_settings.log_tracked"
```

Do

To set the list of settings for which sets should be logged:

```
! U1 do "alerts.tracked_settings.log_tracked"
 "settings.name1,settings.name2..."
```

Values

Settings with commas between names.

Default

""

alerts.tracked_settings.max_log_entries

Sets or retrieves the maximum number of entries to be shown in the `alerts.tracked_settings.log`.

Setvar

To set the maximum number of entries:

```
! U1 setvar "alerts.tracked_settings.max_log_entries" "value"
! U1 do "alerts.tracked_settings.max_log_entries" "value"
```

Values

`"0"` to `"10000"`

Default

`"100"`

Getvar

To retrieve the maximum number of entries :

```
! U1 getvar "alerts.tracked_settings.max_log_entries"
```

alerts.tracked_sgds.log

This command reports the log of the settings listed in `alerts.tracked_settings.log_tracked`. The log entries will be fully JSON complaint.

Getvar

To retrieve the current log:

```
! U1 getvar "alerts.tracked_sgds.log"
```

Example

Sending `! U1 getvar "alerts.tracked_settings.log"` returns:

```
:"[{"settingsName":"newValue","timestamp"
:"06-24-2012 19:51:28.641"}]" for 1 entry or
"[{"settingsName":"newValue","timestamp"
:"06-24-2012
19:51:28.641"},\r\n{"settingsName2":"newValue2","timestamp":"06-24-2012
19:51:30.641"}] for 2 entries.
```

When the log is empty, the result will be: `""`

alerts.tracked_sgds.max_log_entries

This command sets the maximum number of entries to be shown in `alerts.tracked_settings.log`.

Setvar

To set the maximum number of alert log entires that will be stored:

```
! U1 setvar "alerts.tracked_sgds.max_log_entries" "value"
```

Values

`"0"` to `"10000"`

Default

`"100"`

Setting the value to `"0"` disables logging.

Getvar

To return the setting for the maximum number of alert log entires that will be stored:

```
! U1 getvar "alerts.tracked_sgds.max_log_entries"
```

Do

To set the maximum number of alert log entires that will be stored:

```
! U1 setvar "alerts.tracked_sgds.max_log_entries" "value"
```

Values

`"0"` to `"10000"`

Default

`"100"`

Setting the value to `"0"` disables logging.

Example

This example sets the maximum log entries to `"50"`.

```
! U1 setvar "alerts.tracked_sgds.max_log_entries" "50"
```

alerts.tracked_sgds.zbi_notified

This command provides a comma-delimited list of settings for which ZBI should be notified when the value is set.

Setvar

To set the list of the settings for which ZBI will be notified when the setting is set:

```
! U1 setvar "alerts.tracked_settings.zbi_notified"
 "settings.name1,settings.name2,etc."
```

Values

A comma delimited list of settings names.

Default

""

Getvar

To retrieve the list of the settings for which ZBI will be notified when the value is set:

```
U1 getvar "alerts.tracked_settings.zbi_notified"
```

Do

To set the list of the settings for which ZBI will be notified when the setting is set:

```
! U1 setvar "alerts.tracked_settings.zbi_notified"
 "settings.name1,settings.name2,etc."
```

Values

A comma delimited list of settings names.

Default

""

apl.enable

The `setvar` enables or disables a Virtual Device. The `getvar` returns the currently enabled Virtual Device.

Setvar

To enable or disable a virtual device:

```
! U1 setvar "apl.enable" "value"
```

Values

`"none"`, `"apl-d"`, `"apl-i"`, `"apl-e"`, `"apl-l"`, `"apl-m"`, `"apl-mi"`, `"apl-o"`, `"apl-t"`, `"pdf"`

Default

NA

Example

```
! U1 setvar "apl.enable" "pdf"
```

Getvar

To return the currently enabled Virtual Device:

```
! U1 getvar "apl.enable"
```

NOTE: Only the Virtual Device apps loaded on the printer can be enabled.

apl.framework_version

Returns the revision number of the Virtual Device framework.

Getvar

To return the revision number of the Virtual Device framework:

```
! U1 getvar "apl.framework_version"
```

apl.settings

Link-0S v6.3 and later include PDF Direct, which enables a printer to directly process PDF files received for printing. When using PDF Direct, sending a new `apl.settings` command overrides all previous commands. It is possible to send one command to set the multiple values.

By default, PDF Direct is disabled. To enable PDF Direct, use the command:

```
! U1 setvar "apl.enable" "pdf"
```

Setvar

To configure PDF Direct:

```
! U1 setvar "apl.settings" "value"
```

Values

`"dither"`, `"scale"`, `"scale-to-fit"`, `"no-varlen"`, `"orient"`, `"zpl-attach"`

NOTE: Multiple values can be specified in an `apl.settings` command.

Dithering

The default is off. To turn on, set `apl.settings` to include `dither`. For example:

```
! U1 setvar "apl.settings" "dither"
```

Scaling

Auto-Scale/Rotate

Default is off. To turn on, set `apl.settings` to include `scale-to-fit`. This setting modifies the size of the text or image to fit the media. The `scale-to-fit` setting takes precedence over `scale=WxH`, regardless of their order in a command.

For example:

```
! U1 setvar "apl.settings" "scale-to-fit no-varlen orient=N"
```

Scale Factor

Default is off. To turn on, set `apl.settings` to include `scale=`WxH. The W and H values are percentages that range from 1 to 100.

For example:

```
! U1 setvar "apl.settings" "scale=50x50"
```

Variable Length

Default is on when `ezpl.media_type` is set to `continuous`. To turn off, set `apl.settings` to include `no-varlen`. Specify `no-varlen` to use `scale-to-fit` with continuous media of a fixed or pre-configured length. The `varlen` setting disables `scale-to-fit`, use scale factor instead.

For example:

```
! U1 setvar "apl.settings" "scale-to-fit no-varlen scale=50x50"
```

NOTE: Use the Scale Factor and Auto-Scale with caution. Scaling can result in unscannable barcodes.

Page Orientation

Page Orientation

Default is ZPL "Inverted" (top-first), to handle multi-page documents. To change, set `apl.settings` to include `orient=N`. Possible values are N (normal) or I (inverted).

Advanced Options

ZPL Attach

All `"^XA..^XZ"` strings are concatenated and appended to the next PDF document (all pages), after which the strings are cleared. Use this setting to prepend raw ZPL data as a header to PDF documents.

Note that `"^XA..^XZ"` strings, and other data including SGDs, will continue to be passed through when received.

The default value is off. To enable, set `apl.settings` to include `"zpl-attach"`.

Getvar

To return the current PDF Direct settings:

```
! U1 getvar "apl.settings"
```

apl.version

This command returns the revision number of the Virtual Device system.

Getvar

To display the revision number of the Virtual Device system:

```
! U1 getvar "apl.version"
```

appl.bootblock

This command refers to the bootblock version. On the configuration label, the bootblock number is identified as the hardware ID.

Getvar

To return the bootblock version number that appears on the configuration label:

```
! U1 getvar "appl.bootblock"
```

Example

In this example, the `getvar` returns the bootblock version number.

```
! U1 getvar "appl.bootblock"
```

appl.date

This command refers to the date the firmware was created.

Getvar

To respond with the date the firmware was created in the mm/dd/yy format:

```
! U1 getvar "appl.date"
```

Example

In this example, the `getvar` returns the date the firmware was created.

```
! U1 getvar "appl.date"
"01/29/10"
```

appl.link_os_version

This command lists the version of the Link-OS™ feature set that is supported by the printer.

Getvar

To retrieve the Link-OS™ version of the printer:

```
! U1 getvar "appl.link_os_version"
```

Example

In this example, the `getvar` command returns version 1.0 of Link-OS™.

```
! U1 getvar "appl.link_os_version"
```

returns

`"1.0"`

appl.link_os_version_full

This command returns the full Link-OS version, including any extension release information.

Getvar

To retrieve the Link-OS version:

```
! U1 getvar "appl.link_os_version_full"
```

appl.name

This command refers to the printer's firmware version.

Getvar

To return the printer's firmware version:

```
! U1 getvar "appl.name"
```

Example

In this example, the `getvar` returns the printer's firmware version.

```
! U1 getvar "appl.name"
```

appl.option_board_version

This command returns the version number of the firmware running on the wireless option board.

Getvar

To return the version number of the firmware running on the wireless option board:

```
! U1 getvar "appl.option_board_version"
```

Example

This command returns the version number of the firmware running on the wireless option board.

```
! U1 getvar "appl.option_board_version"
"0.0.0 *"
```

capture.channel1.count

This command indicates the number of times that `capture.channel1.delimiter` was seen on the port specified in `capture.channel1.port`. Additionally, it indicates how many times `capture.channel1.data.raw` has been updated with user data as well as the number of times we reached the `capture.channel1.max_length`.

This will be shown in the HZA response under the capture data section.

Getvar

To return the number of times that `capture.channel1.delimiter` was seen on the port specified in `capture.channel1.port`:

```
! U1 getvar "capture.count"
```

capture.channel1.data.mime

This command provides a view to the data captured on the port specified by `capture.channel1.port` in a mime/base64 encoded format.

Getvar

To retrieve the data captured on the port specified by `capture.channel1.port`:

```
! U1 getvar "capture.channel1.data.mime"
```

Result

Data in mime-encoded format.

capture.channel1.data.raw

This command retrieves the user data captured off of the port specified in `capture.channel1.port`.

Any binary zeros in the capture.data stream will be replaced with the escaped representation of NULL ("\000"). The delimiter data is not stored as part of the captured data.

This will be shown in the HZA output within capture data section.

Getvar

To retrieve the user data captured off of the port specified in `capture.channel1.port`:

```
! U1 getvar "capture.channel1.data.raw"
```

capture.channel1.delimiter

This command stores the delimiter used to partition data received on the port specified by `capture.channel1.port` and stored in `capture.channel1.data.raw` and `capture.channel1.data.mime`.

This will be reported in the data capture section of the HZA response.

Setvar

To set the delimiter used to partition data received on the `capture.channel1.port`:

```
! U1 setvar "capture.channel1.delimiter" "delimiter"
```

Values

Any character set up to a maximum of 64 characters in length.

NOTE: Binary data can be used in the delimiter. To do this enter a '\' and then the 3 digit octal value of the character. `"\\"` = `'\'` in some tools, so to get `\002` you may need to enter `“\\002”`. Escaped octal characters count as a single character and not 4 (e.g. a delimiter of `“\001\000\002”` is 3 characters, not 12)

Default

`"\012"`

Getvar

To retrieve the delimiter:

```
! U1 getvar "capture.channel1.delimiter"
```

Example

Binary data can be used in the delimiter. To do this enter a `'\'` and then the 3 digit octal value of the character. Note: `"\\"` = `'\'` in some tools, so to get `\002` you may need to enter `"\\002"`.

```
"\000" = NULL (single character)
"end\015\012\000" = 'e'+'n'+'d'+'\r'+'\n'+ NULL (total of 6 characters
```

capture.channel1.max_length

This command sets a length indicating when to copy captured data to the data SGD if the delimiter has not been seen yet.

If the delimiter and the max_length are reached at the same time, the delimiter will not be part of the captured data. Of only part of the delimiter has been received, then the part of the delimiter we have received, will be part of the capture data.

When the max_length is changed, any data currently in the buffer will be thrown away, and the new value of max_length will be used.

The Capture Port shall be defaulted to 1000 bytes by any mechanism (including ^JUF, ^JUN, ^JUA, and `device.restore_defaults`).

Setvar

To instruct the printer to set a default data capture length:

```
! U1 setvar "capture.channel1.max_length" "value"
```

Values

"1" to "3000"

Default

"1000"

Getvar

To retrieve the default data capture length:

```
! U1 getvar "capture.channel1.max_length"
```

capture.channel1.port

This command determines the port that should be monitored for user data. This allows the user to attach an external device, such as a keyboard or barcode scanner, and have input captured into the `capture.channel1.data.raw` command. Once the data is in the SGD they can use it as they would any other SGD (this includes functionality that allows users to be sent an alert when an SGD value changes).

The data received on the specified port will be read until the value in `capture.channel1.delimiter` is seen, at which point the data received until (but not including) the delimiter will be stored in `capture.channel1.data.raw`.

For the port specified in `capture.channel1.port`, no data will be sent to any of the parsers on that port. All data received is assumed to be user input that is to be placed in `capture.channel1.data.raw`. To disable the data capture functionality, set `capture.channel1.port` to `"off"`

The delimiter will not be stored in `capture.channel1.data.raw`. The port will be shown in the data capture portion of the HZA response. The capture port shall be defaulted to `“off”` by any mechanism (including `^JUF, ^JUA`, and `device.restore_defaults`).

Setvar

To set the port to be monitored for user data:

```
! U1 setvar "capture.channel1.port" "value"
```

Values

- `off` means no data is stored in `capture.channel1.data.raw` and all data is sent to the parsers - normal operation
- `serial` means data is read off the serial port. No data sent to the parsers on this port.
- `usb` means ata is read off the USB port. No data sent to the parsers on this port.
- `bt` means data is read off the Bluetooth® port. No data sent to the parsers on this port.
- `usb_host` is not yet supported. Reserved for when usb host is implemented.

Default

`"off"`

Getvar

To retrieve the printer’s current port being monitored for user data:

```
! U1 getvar "capture.channel1.port"
```

Example

This example sets the command value to `"off"`, preventing it from capturing data.

```
! U1 setvar "capture.channel1.port" "off"
```

CISDFCRC16 Download Files

The CISDFCRC16 command downloads supported files types to the printer.

NOTE: When using certificate files, your printer supports:

- Using Privacy Enhanced Mail (PEM) formatted certificate files.
- Using the client certificate and private key as two files, each downloaded separately.
- Using exportable PAC files for EAP-FAST.

NOTE: When using certificate files, the time on the printer must be set correctly for the websocket connection to succeed, as the time is used in the certificate validation. Each line should be terminated with a CR/LF.

Type

```
! CISDFCRC16
<crc>
<filename>
<size>
<checksum>
<data>
```

Parameters	Values
<crc>	A four digit CRC value in hexadecimal. If 0000 is entered, then the CRC validation is ignored. For examples, see below.
<filename>	File name that is stored on the file system of the printer. An extension must be specified. Files must be saved to the E: drive.
<size>	An eight digit file size specified in hexadecimal which indicates the number of bytes in the <data> section.
<checksum>	A four digit checksum value in hexadecimal. If 0000 is entered, the CRC validation is ignored. The checksum value is calculated using the sum of the bytes in the <data> section. For examples, see below.
<data>	Binary data saved on the printer's file system as <filename>. Number of bytes in this field must match the <size> parameter.

NOTE: This command can be used in place of the ~DG and ~DY command for more saving and loading options. ~DY is the preferred command to download TrueType fonts on printers with firmware later than X.13. The CISDFCRC16 command also supports downloading wireless certificate files

Example 1

This example shows the CISDFCRC16 command used to download a private key file (privkey.nrd) to the printer. The different sections of the command are on separate lines.

```
! CISDFCRC16
BA0B
privkey.nrd
```

```
0000037B
E3AF
-----BEGIN RSA PRIVATE KEY-----
MIICXgIBAAKBgQDQXu/E9YuGlScfWQepZa8Qe/1mJRpmk8oPhPVvam/4M5/WaWQp
3/p1f8J17/hDH8fFq5Dnx3/tHaU7A4SKO8GeghX5hnp/mt4tuQEvsXkCrcgS1puz
z5dbO7ThhuzxYClnr7uiXPvSRXawgwDTPas+0q/6gHeUSXtA0EofuIyv7wIDAQAB
AoGBAJPnf3wn6wT5pE59DJIyakRiLmkt1wKOzvObJfgS7i2Yv1EbeAy9PnPe3vKG
Bovm6A+oi2/qTSTLUTiFc7QHXJPVxLmRiHMbf1Q8j+VJkGTpWt8EY/Px+HSM2HAP
jqd+Im0IiE9RQPsxWQH9UaauF6nl5gIfMF74BIPsVzFXLFfxAkEA6zSrCKCycE/P
14cjZibnLiWxdL3U3I9eWuhmIS37RB6UJFBCWUPWr26HlHzOKqhOUMbFf5hOmvkZ
gciN9A8kxwJBAOLK7Gyorre8iK9IMMWc7OIJc7H8pH1y/N2OtyaC1XuPfqz0H4PH
w2W2m3BhZ7ggHJLLiiFVF+Hr5X7cibFDo5kCQQDFe5lHSzXHWxvViN/N+0gL1RYk
QOcisTW1+n8VyLe5wDr+Km0q6eytq44mvIuWAW6QH/TfZxBIynICKFQX4UctAkAm
P80iAkz9RfnTfhxjp7S35poxoYdodPU6tLAk+ZnhrfDSYJXUFuPYirSqfnMMtbW7
+EICnyRZAP0CqVU7pUm5AkEAnH2O6dKvUvwOEX+CsCVATRrejKLCeJ+6YZWqiD9X
0XGJgrHNXGpDtQiVSGM59p0XnHTZJYjvVNdNOMnhg333nQ==
-----END RSA PRIVATE KEY-----
```

Example 2

These are examples of CRC and checksum values:

The value of the <crc> field is calculated the CRC-16 for the contents of a specified file using the CRC16-CCITT polynomial which is x^16 + x^12 + x^5 + 1. It is calculated using an initial CRC of 0x0000.

Given 4 bytes of data : 0x25, 0x62, 0x3F, 0x52:

1. Adding all bytes together gives 0x118.
2. Drop the carry nibble to get 0x18.
3. Get the two's complement of the 0x18 to get 0xE8.
4. This is the checksum byte.

comm.baud

This command refers to the printer's comm (cable) baud rate.

NOTE: Once the printer's communication port parameters have been changed, the host terminal must also be configured to match the new printer settings before the host can communicate again.

Setvar

To instruct the printer to change the baud rate:

```
! U1 setvar "comm.baud" "value"
```

Values

- "9600"
- "19200"
- "38400"
- "57600"
- "115200"

Default

"19200"

Getvar

To instruct the printer to respond with the currently set printer baud rate:

```
! U1 getvar "comm.baud"
```

Examples

In this example, the `getvar` retrieves the current baud rate.

```
! U1 getvar "comm.baud"
```

This `setvar` example sets the communications baud rate to 19200 BPS.

```
! U1 setvar "comm.baud" "19200"
```

comm.halt

This command halts all communications to the serial port if an error condition occurs.

Setvar

To instruct the printer to halt communication to the printer:

```
! U1 setvar "comm.halt" "value"
```

Values

- `"yes"`
- `"no"`

Default

`"yes"`

Getvar

To return the current value:

```
! U1 getvar "comm.halt"
```

Values

- `"yes"`
- `"no"`

Example

This `setvar` example sets the value set to `"yes"`.

```
! U1 setvar "comm.halt" "yes"
```

comm.mode

This command selects which serial interface to use. Other than RS232, the other options are used only with an external dongle.

Getvar

To report the serial interface currently in use:

```
! U1 getvar “comm.mode”
```

Values

- `"rs232"`
- `"rs422_rs485"`
- `"rs485_multidrop"`

Default

`"rs232"`

Setvar

To specify the serial interface to be used:

```
! U1 setvar “comm.mode” "value"
```

Values

- For ZE5X1
  - `"rs232"`
  - `"rs422_rs485"`
  - `"rs485_multidrop"`
- For other printers
  - `"rs232"`

comm.pnp_option

The `comm.pnp_option` command configures the RS-232 Serial Port Plug and Play setting on the printer.

Setvar

To instruct the printer to change the `comm.pnp_option` setting:

```
! U1 setvar "comm.pnp_option" "value"
```

Values

- `"off"` sets the printer to not attempt Plug and Play via serial at startup
- `"on"` sets the printer to attempt Plug and Play via serial at startup

Default

`"off"`

Getvar

To return the current setting for the `comm.pnp_option` setting:

```
! U1 getvar "comm.pnp_option"
```

Example

In this example, the `setvar` sets the serial port communications state to `"on"`.

```
! U1 setvar "comm.pnp_option" "on"
```

When the setvar value is set to `"on"`,the `getvarresult` is `"on"`.

NOTE: Not all Operating Systems and computer hardware support Plug and Play over RS-232 Serial port connections.

comm.type

This printer setting determines the behavior of the serial port interface. It selects one of three serial communication states: DTE, DCE or Autodetect.

Setvar

To instruct the printer to change the serial port interface type:

```
! U1 setvar "comm.type" "value"
```

Values

- `"auto"` = Autodetect
- `"dte"` = Force DTE (Tx on pin 2)
- `"dce"` = Force DCE (Rx on pin 2)

Default

`"auto"`

Getvar

To instruct the printer to respond with the current serial port interface type:

```
! U1 getvar "comm.type"
```

Examples

In this example, the `getvar` retrieves the serial port communications state.

```
! U1 getvar "comm.type"
```

This `setvar` example sets the communications port state to auto-detect.

```
! U1 setvar "comm.type" "auto"
```

comm.parity

This command sets the printer's communication parity.

Once the printer's communication port parameters have been changed, the host terminal must also be configured to match the new printer settings before the host can communicate again.

Setvar

To instruct the printer to set the communication port parity:

```
! U1 setvar "comm.parity" "value"
```

Values

- "N" — None
- "E" — Even
- "O" — Odd

Getvar

To instruct the printer to respond with the currently set printer parity:

```
! U1 getvar "comm.parity"
```

Examples

In this example, the `getvar` retrieves the currently set printer parity.

```
! U1 getvar "comm.parity"
```

This `setvar` example sets the parity to None.

```
! U1 setvar "comm.parity" "N"
```

comm.stop_bits

This command refers to the communication port stop bits of the printer.

NOTE: Once the printer's communication port parameters have been changed, the host terminal must also be configured to match the new printer settings before the host can communicate again.

Setvar

To instruct the printer to configure the `comm.port` stop bit value:

```
! U1 setvar "comm.stop_bits" "value"
```

Values

- `"1"`
- `"2"`

Default

`"1"`

Getvar

To instruct the printer to respond with the currently set stop bit value:

```
! U1 getvar "comm.stop_bits"
```

Example

In this example, the `getvar` retrieves the currently set stop bit value.

```
! U1 getvar "comm.stop_bits"
```

This `setvar` example configures the `comm.port` for 1 stop bit.

```
! U1 setvar "comm.stop_bits" "1"
```

cradle.comm.baud

Sets or retrieves the cradle serial USB port baud rate.

Setvar

To set the cradle serial USB port baud rate:

```
! U1 setvar "cradle.comm.baud" "value"
```

Values

"300", "1200", "2400", "4800", "9600", "19200", "38400", "57600", "115200", "230400", "460800", "921600"

Default

"115200"

Getvar

To return the cradle serial USB port baud rate:

```
! U1 getvar "cradle.comm.baud"
```

cradle.comm.handshake

Sets or retrieves the cradle serial USB port handshake mode.

Setvar

To set the cradle serial USB port handshake mode:

```
! U1 setvar "cradle.comm.handshake" "value"
```

Values

- `"rts/cts"` use hardware handshake via the request-to-send/clear-to-send pins
- `"xon/xoff"` use software handshake
- `"none"` no flow control

Default

`"rts/cts"`

Getvar

To retrieve the cradle serial USB port handshake mode:

```
! U1 getvar "cradle.comm.handshake"
```

cutter.clean_cutter

This command determines if the clean cutter option is enabled or disabled.

Setvar

To instruct the printer to set the clean cutter option:

```
! U1 setvar "cutter.clean_cutter"
```

Values

- `"on"` turns on clean cutter
- `"off"` turns off clean cutter

Default

`"on"`

Getvar

To retrieve the status of the clean cutter option:

```
! U1 getvar "cutter.clean_cutter"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "cutter.clean_cutter" "on"
```

When the setvar value is set to `"on"`, the `getvar` result is `"on"`.

cutter.clean_reminder_enable

This command enables or disables the cutter cleaning reminder feature.

Setvar

To set the cutter cleaning reminder feature:

```
! U1 setvar "cutter.clean_reminder_enable" "value"
```

Values

- "1" = enables the clean cutter reminder
- "2" = disables the clean cutter reminder

Default

disabled

Getvar

To retrieve the status of the `cutter.clean_reminder_enable` command:

```
! U1 getvar "cutter.clean_reminder_enable"
```

Example

In this example, the `setvar` sets the value to `"1"` (enabled).

```
! U1 setvar "cutter.clean_reminder_enable" "1"
```

cutter.clean_reminder_threshold

This command sets the threshold to trigger a cutter cleaning reminder alert.

Setvar

To set the cutter cleaning reminder feature:

```
! U1 setvar "cutter.clean_reminder_threshold" "value"
```

Values

0–4294967295

Default

100000

Getvar

To retrieve the status of the `cutter.clean_reminder_threshold` command:

```
! U1 getvar "cutter.clean_reminder_threshold"
```

Example

In this example, the `setvar` sets the value to `"200000"` (enabled).

```
! U1 setvar "cutter.clean_reminder_threshold" "200000"
```

device.allow_firmware_downloads

This command sets if the firmware downloads are allowed.

Setvar

To set the command:

```
! U1 setvar "device.allow_firmware_downloads" "value"
```

Values

- `"yes"` allow firmware downloads
- `"no"` does not allow firmware downloads

Default

`"yes"`

Getvar

To view the current setting:

```
! U1 getvar "device.allow_firmware_downloads"
```

Example

This setvar example sets the firmware downloads feature to `"no"`.

```
! U1 setvar "device.allow_firmware_downloads" "no"
```

device.applicator.data_ready

This command will specify if a `"high"` or `"low"` value is required for the applicator to indicate it is ready to receive data.

Setvar

To set the value:

```
! U1 setvar "device.applicator.data_ready" "value"
```

Values

- `"high"`
- `"low"`

Default

- `"high"`
- `"low"`

Getvar

To instruct the printer to respond with the currently set value:

```
! U1 getvar "device.applicator.data_ready"
```

device.applicator.end_print

This command allows you to control an online verifier or applicator device.

Setvar

This command is similar to the b parameter for ^JJ on page 266.

To set the value for the applicator port mode:

```
! U1 setvar "device.applicator.end_print" "value"
```

Values

- "off"
- "1"End Print signal normally high, and low only when the printer is moving the label forward.
- "2" End Print signal normally low, and high only when the printer is moving the label forward.
- "3" End Print signal normally high, and low for 20 ms when a label has been printed and positioned.
- "4" End Print signal normally low, and high for 20 ms when a label has been printed and positioned.

Default

"off"

Getvar

To instruct the printer to respond with the currently set value:

```
! U1 getvar "device.applicator.end_print"
```

device.applicator.feed

This command will specify if a `"high"` or `"low"` value is required for an applicator to feed media.

Setvar

To set the value:

```
! U1 setvar "device.applicator.feed" "value"
```

Values

- `"high"`
- `"low"`

Default

`"low"`

Getvar

To instruct the printer to respond with the currently set value:

```
! U1 getvar "device.applicator.feed"
```

device.applicator.media_out

This command will specify if a `"high"` or `"low"` value is required for an applicator to indicate that the media has run out.

Setvar

To set the value:

```
! U1 setvar "device.applicator.media_out" "value"
```

Values

- `"high"`
- `"low"`

Default

`"low"`

Getvar

To instruct the printer to respond with the currently set value:

```
! U1 getvar "device.applicator.media_out"
```

device.applicator.pause

This command will specify if a `"high"` or `"low"` value is required for an applicator to pause printing.

Setvar

To set the value:

```
! U1 setvar "device.applicator.pause" "value"
```

Values

- `"high"`
- `"low"`

Default

`"low"`

Getvar

To instruct the printer to respond with the currently set value:

```
! U1 getvar "device.applicator.pause"
```

device.applicator.reprint

This command will specify if a `"high"` or `"low"` value is required for an applicator to reprint a label.

Setvar

This command is similar to ~PR on page 324.

To set the value:

```
! U1 setvar "device.applicator.reprint" "value"
```

Values

- `"high"`
- `"low"`

Default

`"low"`

Getvar

To instruct the printer to respond with the currently set value:

```
! U1 getvar "device.applicator.reprint"
```

device.applicator.rfid_void

This command will specify if a `"high"` or `"low"` value is used for the RFID void signal, which occurs when an RFID label is voided by the printer.

This command is supported only on the ZT411/ZT421 and ZT600 Series printers.

Setvar

To set the value:

```
! U1 setvar "device.applicator.rfid_void" "value"
```

Values

- `"high"`
- `"low"`

Default

`"low"`

Getvar

To instruct the printer to respond with the currently set value:

```
! U1 getvar "device.applicator.rfid_void"
```

device.applicator.ribbon_low

This command will specify if a `"high"` or `"low"` value is required for an applicator to indicate that the ribbon is running out.

Setvar

To set the value:

```
! U1 setvar "device.applicator.ribbon_low" "value"
```

Values

- `"high"`
- `"low"`

Default

`"high"`

Getvar

To instruct the printer to respond with the currently set value:

```
! U1 getvar "device.applicator.ribbon_low"
```

device.applicator.ribbon_out

This command will specify if a `"high"` or `"low"` value is required for an applicator to indate that the ribbon has run out.

Setvar

To set the value:

```
! U1 setvar "device.applicator.ribbon_out"
```

Values

- `"high"`
- `"low"`

Default

`"low"`

Getvar

To instruct the printer to respond with the currently set value:

```
! U1 getvar "device.applicator.ribbon_out"
```

device.applicator.service_required

This command will specify if a "high" or "low" value is required for an applicator to indicate that maintenance is required.

Setvar

To set the value:

```
! U1 setvar "device.applicator.service_required" "value"
```

Values

- "high"
- "low"

Default

"low"

Getvar

To instruct the printer to respond with the currently set value:

```
! U1 getvar "device.applicator.service_required"
```

device.applicator.start_print

This command will specify if a `"high"` or `"low"` value is required for an applicator to start printing.

Setvar

To set the value:

```
! U1 setvar "device.applicator.start_print" "value"
```

Values

- `"high"`
- `"low"`

Default

`"low"`

Getvar

To instruct the printer to respond with the currently set value:

```
! U1 getvar "device.applicator.start_print"
```

device.bluetooth_installed

Determines if there is a Bluetooth radio installed in the printer.

Getvar

To return if the Bluetooth radio is installed in the printer or not:

```
! U1 getvar "device.bluetooth_installed"
```

Result

- `"yes"` a Bluetooth radio is installed in the printer.
- `"no"` a Bluetooth radio is not installed in the printer.

device.command_override.active

This command enables or disables the `device.command_override` function. When enabled, the printer will ignore the list of commands previously specified using the `device.command_override.add`. Use of this command does not modify the list of commands to be overridden.

NOTE: This setting is not defaulted as part of a factory default (`^JUF` or `^default`). The setting is persistent across a power cycle or rest (`~JR` or `device.reset`).

Setvar

To turn on/off the `device.command_override` function:

```
! U1 setvar "device.command_override.active" "value"
```

Values

- `"yes"` = active
- `"no"` = inactive

Default

`"yes"`

Getvar

To return the active/inactive state of `device.command_override` command:

```
! U1 getvar "device.command_override.active"
```

Result

- `"yes"` the command is active
- `"no"` the command is inactive

Example

```
! U1 setvar "device.command_override.active" "yes"
```

```
! U1 setvar "device.command_override.active" "no"
```

device.command_override.add

This command adds a specified command to the list of commands that will be ignored by the printer. The list is saved when the printer is powered off and is not cleared when the printer is defaulted.

- If there are items in the list and the `device.command_override.active` is set to `"yes"`, then the config label will show `ACTIVE COMMAND OVERRIDE.`
- If there are no items in the list or `device.command_override.active` is set to `"no"`, then the config label will show `INACTIVE COMMAND OVERRIDE.`

Setvar

To instruct the printer to add a specified command to the list of override commands:

```
! U1 setvar "device.command_override.add" "command"
```

Values

Any ZPL or Set/Get/Do command.

Default

NA

Example

When specifying a ZPL command, the command must be preceded by the current format or control prefix character (e.g. ^ or ~). Multiple commands must be declared with their own `setvar` declaration.

```
! U1 setvar "device.command_override.add" "^MN"
! U1 setvar "device.command_override.add" "^PR"
! U1 setvar "device.command_override.add" "comm.baud"
! U1 setvar "device.command_override.add" "device.reset"
```

The following example is not valid.

```
! U1 setvar "device.command_override.add" "~HI,~HS,^MN"
```

Instead, send the command as:

```
! U1 setvar "device.command_override.add" "~HI"
! U1 setvar "device.command_override.add" "~HS"
! U1 setvar "device.command_override.add" "^MN"
```

You cannot add `"device.command_override.clear"` to the list of accepted override commands.

device.command_override.clear

This command clears all commands from the command override list.

Setvar

To clear the list of override commands specified in `device.command_override.list`:

```
! U1 setvar "device.command_override.clear" "value"
```

Values
NA

Default
NA

device.command_override.list

This command returns to the host a list of the current set of commands that the printer will ignore.

Getvar

To print a comma-delimited list of override commands:

```
! U1 getvar "device.command_override.list"
```

Values
NA

Default
NA

device.company_contact

This command sets the company contact information, which can be accessed from the `server/sysinfo.htm` webpage.

Setvar

To set the company contact information:

```
! U1 setvar "device.company_contact" "value"
```

Values

A string up to 128 characters in length.

Result

`""`

Getvar

To return the current company contact information:

```
! U1 getvar "device.company_contact"
```

Example

This `setvar` example shows the value set to `"Zebra"`.

```
! U1 setvar "device.company_contact" "Zebra"
```

device.configuration_number

Returns the current SKU number of the printer.

Getvar

To return the device configuration number:

```
! U1 getvar "device.configuration_number"
```

device.cpcl_synchronous_mode

This command enables/disables CPCL synchronous mode. When the printer is in sync mode, parsing will "lock" while printing is going on, allowing behavior similar to that of the legacy SH3 mobile printers.

Setvar

To turn on or off the `device.cpcl_synchronous_mode`:

```
! U1 setvar "device.cpcl_synchronous_mode" "value"
```

Values

- `"on"` puts the printer in CPCL synchronous mode
- `"off"` puts the printer in default mode

Default

`"off"`

Getvar

To return the current value of the setting:

```
! U1 getvar "device.cpcl_synchronous_mode"
```

Example

Consider issuing a label immediately followed by an SGD request. When sync mode is `"off"`, the SGD will be returned nearly immediately after submitting the label. When sync mode is `"on"`, the SGD will be returned after the label has printed.

```
! U1 setvar "device.cpcl_synchronous_mode" "on"
```

```
! U1 setvar "device.cpcl_synchronous_mode" "off"
```

device.cutter_installed

This command reports if a cutter unit is installed.

Getvar

To check if a cutter is installed:

```
! U1 getvar "device.cutter_installed"
```

Values

- "Yes"
- "No"

Default

"No"

device.download_connection_timeout

This command instructs the printer to abort a firmware download if the printer fails to receive any download data in the set amount of seconds. If the set amount of seconds is exceeded, the download will be aborted, and the printer automatically restarts. This command prevents the printer from being locked into the downloading state, if the communication to the host is interrupted.

Setvar

To instruct the printer to abort a firmware download if the printer fails to receive any download data in the set amount of seconds:

```
! U1 setvar "device.download_connection_timeout" "value"
```

Values

"0" through "65535"

Default

"0" ("0" disables this feature)

Getvar

To retrieves the connection time out value (in seconds):

```
! U1 getvar "device.download_connection_timeout"
```

Example

This setvar example shows the value set to "0".

```
! U1 setvar "device.download_connection_timeout" "0"
```

When the setvar value is set to "0", the getvar result is "0".

device.download_interactive_mode

This command enables an interactive firmware download. When enabled, the printer sends status messages to the host as the firmware file is sent and processed. The status messages are in JSON format and are sent back over the same channel to which the firmware is being sent.

Getvar

To determine whether interactive mode is enabled:

```
! U1 getvar "device.download_interactive_mode"
```

Values

- `"on"` indicates that the interactive mode is enabled
- `"off"` indicates that the interactive mode is disabled

Default
`"off"`

Setvar

To enable the interactive mode for firmware download:

```
! U1 setvar "device.download_interactive_mode" "on"
```

Values

- `"on"` indicates that the interactive mode is enabled
- `"off"` indicates that the interactive mode is disabled

Restore Default Setting

To set the printer to the default setting for firmware download interactive mode:

```
! U1 setvar "device.restore_defaults" "device.download_interactive_mode"
```

device.epl_legacy_mode

This command places the printer in a 2824/2844 compatibility mode for vertical registration.

Setvar

To instruct the printer to change the epl_legacy_mode setting:

```
! U1 setvar "device.epl_legacy_mode" "value"
```

Values

- `"off"` epl_legacy_mode not active
- `"registration"` EPL legacy registration mode on
- `"print orientation"` EPL legacy print orientation mode on
- `"all"` all EPL legacy modes on

Default

`"off"`

Getvar

To return the current setting value for the device.epl_legacy_mode setting:

```
! U1 getvar "device.epl_legacy_mode"
```

Example

This `setvar` example shows the value set to `"registration"`.

```
! U1 setvar "device.epl_legacy_mode" "registration"
```

This `setvar` example shows the value set to `"print_orientation"`.

```
! U1 setvar "device.epl_legacy_mode" "print_orientation"
```

This getvar example shows the response when the value was set to `"registration"` and `"print_orientation"`.

```
! U1 getvar "device.epl_legacy_mode"
"registration, print_orientation"
```

This getvar example shows the response when value was set to `"all"` .

```
! U1 getvar "device.epl_legacy_mode"
"all"
```

Example

This `setvar` example shows the value set to `"registration"`.

```
! U1 setvar "device.epl_legacy_mode" "registration"
```

This `setvar` example shows the value set to `"print_orientation"`.

```
! U1 setvar "device.epl_legacy_mode" "print_orientation"
```

This `getvar` example shows the response when the value was set to `"registration"` and `"print_orientation"`.

```
! U1 getvar "device.epl_legacy_mode"
"registration, print_orientation"
```

This `getvar` example shows the response when value was set to `"all"` .

```
! U1 getvar "device.epl_legacy_mode"
"all"
```

NOTE:

- This setting is not defaulted as part of a factory default (`^JUF` or `^default`). The setting is persistent across a power cycle or reset (`~JR` or `device.reset`).
- When setting the “registration” mode, the `"print_orienation"` mode is not changed. Likewise, when setting the `"print_orienation"` mode, the `"registration"` mode is not changed. Using `"off"` or `"all"` changes all modes.

Print Orientation Mode

NOTE: When the printer is powered on, the print orientation defaults to `^PON` (EPL ZB mode). The print orientation setting is not saved across power cycles. This is different than TLP2844, LP2844, TLP2824, LP2824, and TLP3842 printers. Those printers have a default of ZB (ZPL `^PON` mode) and the print orientation setting is saved across power cycles. To make the printer have the print orientation behavior of the TLP2844, LP2844, TLP2824, LP2824, and TLP3842 printers, set the epl_legacy_mode to `"print_orienation"`.

Registration Mode

NOTE: When printing labels using EPL commands, printing starts 1mm from the top edge of the label (from the gap). This is known as the “no print zone”. When printing in ZT mode, the “no print zone” starts at the gap on the leading edge of the label. When printing in ZB mode, the “no print zone” starts from the gap on the trailing edge of the label. In the TLP2844, LP2844, TLP2824, LP2824, and TLP3842 printers, the distance from gap to start of print (the “no print zone”) is not always 1mm.

The table below shows the nominal distance.

Distance from Edge of Label to First Print Line (No Print Zone)			
Legacy Printer Model	**New Printer Model**	**ZT Mode**	**ZB Mode**
LP2844	GX420, GK420 (direct thermal)	1.9 mm	0.0 mm
TLP2844	GX420, GK420 (thermal transfer)	0.4 mm	1.6 mm
TLP3842	GX430 (thermal transfer)	0.0 mm	1.2 mm
LP2824	LP 2824 Plus (direct thermal)	1.5 mm	0.4 mm
TLP2824	TLP 2824 Plus (thermal transfer)	0.1 mm	1.8 mm

NOTE: Setting epl_legacy_mode to `"registration"` selects the distance shown in the table. Setting epl_legacy_mode to `"off"` selects a no print zone distance of 1mm.

device.feature.bluetooth_le

Indicates whether or not the printer supports Bluetooth LE.

Getvar

To return if the printer supports Bluetooth LE:

```
! U1 getvar "device.feature.bluetooth_le"
```

Values

- `"present"` Bluetooth LE radio is installed
- `"not present"`Bluetooth LE radio is not installed
- `"not available"`Bluetooth LE radio is not available on this printer

device.feature.mcr

Indicates if the magnetic card reader is installed and available.

Getvar

To return if the magnetic card reader is installed and available:

```
! U1 getvar "device.feature.mcr"
```

Values

- `"not available"` the magnetic card reader is not available on the printer
- `"not present"` the magnetic card reader is available but not installed
- `"present"` the magnetic card reader is both available and installed on the printer

Default

NA

device.feature.nfc

Indicates if the printer supports the optional Active Near Field Communication (NFC) feature, and if it is currently installed.

Getvar

To see if the printer supports the optional Active Near Field Communication (NFC) feature, and if it is currently installed:

```
! U1 getvar "device.feature.nfc"
```

Values

- `"not available"` active NFC is not supported.
- `"not present"` active NFC is supported, but no reader is installed.
- `"present"` active NFC is supported with a reader is installed.

Default

NA

device.feature.ribbon_cartridge

Indicates if the printer can accept a ribbon cartridge, and if so, if one is installed.

Getvar

To return if a ribbon cartridge is installed or not:

```
! U1 getvar "device.feature.ribbon_cartridge"
```

Result

`"not available"` the ribbon cartridge is not available on the platform

`"not present"` the printer is capable of accepting a ribbon cartridge, but one is not currently installed

`"present"`a ribbon cartridge is installed

device.feature.802_11ac

This command returns information on the 802.11AC radio status.

Getvar

To return the current setting:

```
! U1 getvar "device.feature.802_11ac"
```

Result

- `"not present"` if the printer model supports an 802.11ac option but the printer does not have the feature installed.
- `"not available"` if the printer model does not support an 802.11ac feature option.
- `"present"` if the printer has an 802.11ac radio installed.

device.feature.802_11ax

This command returns the status of the 802.11ax radio.

Getvar

To return the status of the 802.11ax radio:

```
! U1 getvar "device.feature.802_11ax"
```

Values

- `not available` if the printer model does not support the 802.11ax feature.
- `not present` if the printer model supports the 802.11ax feature, but the feature is not installed on the printer.
- `present` if the feature is installed on the printer.

device.feature.head_element_test

This command retrieves the head element test status on the printer.

Getvar

To return the head element test feature availability:

```
! U1 getvar "device.feature.head_element_test"
```

Result

- `"present"` if head test is present on the printer
- `"not present"` if head test is present on the printer
- `"not available"` if head test is not available on the platform

device.friendly_name

This command shows the name assigned to the printer.

Setvar

To set the printer's name:

```
! U1 setvar "device.friendly_name" "value"
```

Default

"xxxxxxxxxx" ("xxxxxxxxxx"represents the main logic board serial number)

Getvar

To retrieve the name assigned to the printer:

```
! U1 getvar "device.friendly_name"
```

Example

This `setvar` example shows the value set to "xxxxxxxxxx".

```
! U1 setvar "device.friendly_name" "xxxxxxxxxx"
```

When the `setvar` value is set to "xxxxxxxxxx", the `getvar` result is "xxxxxxxxxx".

device.frontpanel.feedenabled

This command can be used to enable or disable the **FEED** key or any other key on the printer.

Setvar

To instruct the printer to change the `front_panel.feedenable`d setting:

```
! U1 setvar "device.frontpanel.feedenabled"
```

Values

- `"yes"` Front Panel keys are enabled
- `"no"` Front Panel keys are disabled

Default

- `"no"` for GX420s printers
- `"yes"` all supported printers except GX420s

Power On Default

- `"no"` for GX420s printers
- `"yes"` all supported printers except GX420s

Getvar

To retrieve the current setting for the `front_panel.feedenable` command:

```
! U1 getvar "device.frontpanel.feedenabled"
```

Example

In this example, the `setvar` sets the value to `"no"`.

```
! U1 setvar "device.frontpanel.feedenabled" "no"
```

NOTE:

- On GX420 printers with an LCD display, there is a **SCROLL** and **SELECT** key in addition to the **FEED** key. Both the **SCROLL** and **SELECT** keys are enabled or disabled when the **FEED** key is enabled or disabled using this command.
- On power up, for model GX420s printer, the command value is set to `"no"`. For all other printers, on power up, the command value is set to `"yes"`.

device.frontpanel.key_press

This command instructs the printer to press a button on the front panel.

Setvar

To instruct the printer to press a button on the front panel:

```
! U1 setvar "device.frontpanel.key_press"
```

Values

The values vary per printer, as follows:

ZM400, Z4M/Z6M, and RZ400/RZ600

- "A" Pause
- "B" Feed
- "C" Cancel
- "D" Setup/Exit
- "E" Minus
- "F" Select
- "G" Plus

XiIIIplus

- "A" Pause
- "B" Feed
- "C" Cancel
- "D" Setup/Exit
- "E" Previous
- "F" Next/Save
- "G" Minus
- "H" Plus
- "I" Calibrate

S4M

- "A" Pause
- "B" Feed
- "C" Up Arrow
- "D" Cancel
- "E" Menu
- "F" Enter

Xi4, RXi4

- "A" Pause
- "B" Feed
- "C" Cancel
- "D" Setup/Exit
- "E" Previous
- "F" Next/Save
- "G" Minus
- "H" Plus
- "I" Calibrate

Example

This `setvar` example shows the value set to "A".

```
! U1 setvar "device.frontpanel.key_press" "A"
```

device.frontpanel.line1

This command overrides the content that is shown on the first line of the front panel when the printer is showing the idle display. Use of the `getvar` function is dependent on first using the `setvar` function. For example, to have the first line of the idle display to show HELLO, you must first send a `setvar` command; then a `getvar` command can be sent to retrieve the value HELLO.

Setvar

For details on the supported character set, see ZBI Character Set on page 1600.

To instruct the printer to set the content that is shown on line one of the front panel:

```
! U1 setvar "device.frontpanel.line1" "value"
```

Values

The maximum amount of alphanumeric ASCII characters available for line 1 on the printer's front panel.

Default

`""`

Getvar

To retrieves the content that is shown on line one of the front panel:

```
! U1 getvar "device.frontpanel.line1"
```

Example

This `setvar` example shows the value set to `"sample line 1"`.

```
    ! U1 setvar "device.frontpanel.line1" "sample line 1"
```

When the setvar value is set to `"sample line 1"`, the `getvar` result is `"sample line 1"`.

device.frontpanel.line2

This command overrides the content that is shown on the second line of the front panel when the printer is showing the idle display. Use of the `getvar` function is dependent on using the `setvar` function.

For example, to have the second line of the idle display show HELLO, you must first send a `setvar` command; then a `getvar` command can be sent to retrieve the value HELLO.

For details on the supported character set, see ZBI Character Set on page 1600.

Setvar

To instruct the printer to set the content that shows on line two of the front panel:

```
! U1 setvar "device.frontpanel.line2" "value"
```

Values

The maximum amount of alphanumeric ASCII characters available for line two on the printer's front panel.

Default

""

Getvar

To retrieve the content that shows on line two of the front panel:

```
! U1 getvar "device.frontpanel.line2"
```

Example

This `setvar` example shows the value set to `"sample line 2"`.

```
! U1 setvar "device.frontpanel.line2" "sample line 2"
```

When the `setvar` value is set to `"sample line 2"`, the `getvar` result is `"sample line 2"`.

device.frontpanel.xml

This command retrieves the current content of the front panel in an XML format.

Getvar

To retrieve the file that determines the representation of the front panel:

```
! U1 getvar "device.frontpanel.xml"
```

Example

In this example, the `getvar` shows the status of the LEDs and the two lines of the front panel in XML formatted text. The text below is formatted for easy reading. When you use this command the response will not contain line feeds.

```
! U1 getvar "device.frontpanel.xml"
<FRONT-PANEL>
<LCD>
<LINE1>PRINTER READY</LINE1>
<LINE2>V53.16.0</LINE2>
</LCD>
<LEDS>
<PAUSE-LED>STEADY-OFF</PAUSE-LED>
<DATA-LED>STEADY-OFF</DATA-LED>
<ERROR-LED>STEADY-OFF</ERROR-LED>
</LEDS>
</FRONT-PANEL>
```

device.host_identification

This command is designed to be sent from the host to the Zebra printer to retrieve information. Upon receipt, the printer responds with information on the model, software version, dots-per-millimeter setting, memory size, and any detected options.

This command is equivalent to the ~HI ZPL command.

Getvar

To display information about the printer:

```
! U1 getvar "device.host_identification"
```

Result

"XXXXXX,V1.0.0,dpm,000KB,X"

Values

XXXXXX indicates the model of Zebra printer

V1.0.0 is the version of software

dpm is dots/mm printheads

- 6
- 8
- 12
- 24

000KB is the memory size

- 512KB (.5 MB)
- 1024KB (1 MB)
- 2048KB (2 MB)
- 4096KB (4MB)
- 8192KB (8MB)

x displays options specific to printer (for example, cutter)

device.host_status

When this command is sent to the printer, the printer sends three data strings back. To avoid confusion, the host prints each string on a separate line.

This command is similar to the ~HS ZPL command.

The response for this SGD command does not include the STX at the beginning of each data line and does not include the ETX at the end of each data line as found in the ~HS response. Additionally, the first and second response lines for the SGD command contain a CR/LF at the end of each line.

NOTE: When the command is sent, the printer will not send a response to the host if the printer is in one of these conditions:

- MEDIA OUT
- RIBBON OUT
- HEAD OPEN
- REWINDER FULL
- HEAD OVER-TEMPERATURE

Getvar

To return the current setting value:

```
! U1 getvar "device.host_status"
```

Result

Three strings, each on their own line.

```
"aaa,b,c,dddd,eee,f,g,h,iii,j,k,l
mmm,n,o,p,q,r,s,t,uuuuuuuu,v,www
xxxx,y"
```

See definitions for String 1, String 2, and String 3 below.

String 1

```
"aaa,b,c,dddd,eee,f,g,h,iii,j,k,l"
```

The nine-digit binary number is read according to this table:

aaa	communication (interface) settings[a]
b	paper out flag (1 = paper out)
c	pause flag (1 = pause active)
dddd	label length (value in number of dots)
eee	number of formats in receive buffer
f	buffer full flag (1 = receive buffer full)
g	communications diagnostic mode flag (1 = diagnostic mode active)
h	partial format flag (1 = partial format in progress)

[a] This string specifies the printer's baud rate, number of data bits, number of stop bits, parity setting, and type of handshaking. This value is a three-digit decimal representation of an eight-bit binary number. To evaluate this parameter, first convert the decimal number to a binary number.

aaa = a^8 a^7 a^6 a^5 a^4 a^3 a^2 a^1 a^0	
a^7 = Handshake 0 = Xon/Xoff 1 = DTR	a^8 a^2 a^1 a^0 = Baud 0000 = 110 0001 = 300 0110 = 600 0011 = 1200 0100 = 2400 0101 = 4800 0110 = 9600 0111 = 19200 1000 = 28800* 1001 = 38400* 1010 = 57600* 1011 = 14400 * Available only on certain printer models
a^6 = Parity 0 = Odd 1 = Even	
a^5 = Disable/Enable 0 = Disable 1 = Enable	
a^4 = Stop Bits 0 = 2 Bits 1 = 1 Bit	
a^3 = Data Bits 0 = 7 Bits 1 = 8 Bits	

String 2

```
"mmm,n,o,p,q,r,s,t,uuuuuuuu,v,www"
```

mmm	function settings [a]
n	unused
o	head up flag (1 = head in up position)
p	ribbon out flag (1 = ribbon out)
q	thermal transfer mode flag (1 = Thermal Transfer Mode selected)
r	Print Mode .14↑ Values 4 to 5 are only supported in firmware version V60.14.x, V50.14.x, V53. 15.x, or later. • 0 Rewind • 1 Peel-Off • 2 Tear-Off • 3 Cutter • 4 Applicator • 5 Delayed cut • 6 Reserved [b] • 7 Reserved [b] • 8 Reserved [a] • 9 RFID
s	print width mode
t	label waiting flag (1 = label waiting in Peel-off Mode)
uuuuuuuu	labels remaining in batch
v	format while printing flag (always 1)
www	number of graphic images stored in memory

[a] This string specifies the printer's media type, sensor profile status, and communication diagnostics status. As in String 1, this is a three-digit decimal representation of an eight-bit binary number. First, convert the decimal number to a binary number.

[b] These values are only supported on the Xi4, RXi4, ZM400/ZM600, RZ400/RZ600, and ZT200 Series printers.

The eight-digit binary number is read according to this table:

mmm = m7 m6m m5 m4 m3 m2 m1 m0	
m7 = Media Type 0 = Die-Cut 1 = Continuous	m4 m3 m2 m1 = Unused 0 = Off 1 = On
m6 = Sensor Profile 0 = Off	m0 = Print Mode 0 = Direct Thermal 1 = Thermal Transfer
m5 = Communications Diagnostics 0 = Off 1 = On	

String 3

`"xxxx,y"`

XXXX	password (printers running Link-OS v5.3 or earlier versions)
`0000`	password. (printers running Link-OS 6 or later versions)
y	• 0 (static RAM not installed) • 1 (static RAM installed)

device.idle_display_format

Retrieves and sets the front panel's idle display format.

NOTE: This command does not apply to printers with a color-touch display.

Setvar

To set the front panel's idle display format:

```
! U1 setvar "device.idle_display_format" "value"
```

Values

`fw-version,ip-address,mm/dd/yy-24-hr,mm/dd/yy-12-hr,dd/mm/yy-24-hr,dd/mm/yy-12-hr`

Default

`"fw-version"` (firmware version)

Getvar

To retrieve the front panel's idle display format:

```
! U1 getvar "device.idle_display_format"
```

Result

`fw-version,ip-address,mm/dd/yy-24-hr,mm/dd/yy-12-hr,dd/mm/yy-24-hr,dd/mm/yy-12-h`

device.idle_display_value

Returns the printer's current front panel idle display information.

NOTE: This command does not apply to printers with a color-touch display.

Getvar

To return the printer current front panel idle display information:

```
! U1 getvar "device.idle_display_value"
```

Result

A firmware version, the printer's IP address, or the date.

device.internal_wired_setting_location

This command identifies the location from where internal_wired network specific settings should be retrieved.

Setvar

To specify the location from where internal_wired network specific settings should be retrieved:

```
! U1 setvar "device.internal_wired_setting_location" "value"
```

Values

- `"network card"`
- `"printer"`

NOTE: `"printer"` is the only valid option for the QLn series and ZD500 series printers.

Default

`"network_card"`

Getvar

To display the location where internal_wired network specific settings are retrieved from:

```
! U1 getvar "device.internal_wired_setting_location"
```

NOTE: `"printer"` is the only valid getvar option for the QLn series and ZD500 series printers.

device.jobs_print

This command identifies the number of jobs to be printed.

Getvar

To retrieve the number of jobs to be printed:

```
! U1 getvar "device.jobs_print"
```

Example

In this example, the `getvar` retrieves the jobs currently being printed or last printed.

```
! U1 getvar "device.jobs_print"
"1"
```

device.job_log.total_jobs_logged

This command returns the total number of jobs logged, which is used on the "server/joblog.htm" webpage.

Getvar

To return the current setting value:

```
! U1 getvar "device.job_log.total_jobs"
```

NOTE: The value resets to 0 after a power cycle.

device.languages

This command identifies the programming language that the printer is currently using.

Setvar

To set the printer to the required programming language:

```
! U1 setvar "device.languages" "value"
```

Values

- `"epl"` Eltron Programming Language
- `"epl_zpl"` Eltron Programming Language and Zebra Programming Language
- `"zpl"` Zebra Programming Language
- `"hybrid_xml_zpl"` XML and ZPL Programming Languages
- `"apl-d"` Virtual Device-D (only Link-OS printers)
- `"apl-t"` Virtual Device-T (only desktop and table top printers with Link-OS)
- `"apl-e"` Virtual Device-E (only mobile printers with Link-OS)
- `"apl-o"` Virtual Device-O (only mobile printers with Link-OS)
- `"apl-i"` Virtual Device-I (only Link-OS printers)

NOTE: Not all values are accepted on all printers. Use the `! U1 getvar "allcv"` command to see the range of values that your printer supports. Values other than those listed may be available depending on the firmware version being used.

NOTE: `"zpl" and "hybrid_xml_zpl"` are equivalent. When the setvar is set to `"zpl"`, the getvar result will always be `"hybrid_xml_zpl".`

Default

`"epl_zpl"`

Getvar

To retrieve the programming language that the printer is currently using:

```
! U1 getvar "device.languages"
```

Example

This `setvar` example sets the programming language to `"hybrid_xml_zpl"` using the shorter value of `"zpl"`.

```
! U1 setvar "device.languages" "zpl"
```

device.light.cover_open_brightness

This command sets the brightness level for the Cover Open light.

Setvar

To set the brightness level for the cover open LEDs:

```
! U1 setvar "device.light.cover_open_brightness" "value"
```

Values

- `"high"` the LEDs display at maximum brightness when the cover is open
- `"medium"` the LEDs display at medium brightness when the cover is open
- `"low"` the LEDs display at lowest brightness when the cover is open
- `"off"` the LEDs remain off at all times

Default

`"high"`

Getvar

To retrieve the current brightness level setting for the cover open LEDs:

```
! U1 getvar "device.light.cover_open_brightness"
```

Example

This `setvar` example shows the value set to `"low"`.

```
! U1 setvar "device.light.cover_open_brightness" "low"
```

device.light.head_open_brightness

This command sets the brightness level for the Head Open light.

Setvar

To set the brightness level for the head open LEDs:

```
! U1 setvar "device.light.head_open_brightness" "value"
```

Values

- `"high"` the LEDs display at maximum brightness when the head is open
- `"medium"` the LEDs display at medium brightness when the head is open
- `"low"` the LEDs display at lowest brightness when the head is open
- `"off"` the LEDs remains off at all times

Default

`"high"`

Getvar

To retrieve the current brightness level setting for the head open LEDs:

```
! U1 getvar "device.light.head_open_brightness"
```

Example

This `setvar` example shows the value set to `"medium"`.

```
! U1 setvar "device.light.head_open_brightness" "medium"
```

device.location

Sets the system location, which is used on the "server/sysinfo.htm" webpage.

Setvar

To set the system location:

```
! U1 setvar "device.location" "value"
```

Values

Any ASCII string up to 128 characters.

Default

""(empty string)

Getvar

To retrieve the system location:

```
! U1 getvar "device.location"
```

device.loader_version

This command returns the device loader version.

Getvar

To have the printer return the loader version:

```
! U1 getvar "device.loader_version"
```

Example

In this `getvar` example, the printer returns with the loader version number.

```
! U1 getvar "device.loader_version"
```

device.ltu_installed

This command checks to see if a Liner Take-Up unit is installed.

Getvar

To check if the Liner Take-Up unit is installed or not:

```
! U1 getvar "device.ltu_installed"
```

Values

- "Yes"
- "No"

Default

"No"

device.mcu_communication.revision

This command retrieves the Communication MCU's revision number. The Communication MCU is responsible for performing communication functions.

Getvar

To retrieve the current revision of the Communication MCU application:

```
! U1 getvar "device.mcu_communication.revision"
```

Values

"1" to "254"

device.mcu_cutter.revision

This command retrieves the revision number of the cutter MCU application.

Getvar

To retrieve the revision of the cutter MCU application:

```
! U1 getvar "device.mcu_cutter.revision"
```

Values

A value of "1" to "254" that specifies the current revision.

device.mcu_cutter.desired_revision

This command reports the revision number contained in the download application file for the Cutter MCU. When the contents of this SGD can be used in conjunction with `device.mcu_cutter.revision` to determine if the download of the Cutter MCU was successful.

Getvar

To retrieve the revision number of the most recent application file downloaded to the printer:

```
! U1 getvar "device.mcu_cutter.desired_revision"
```

Values

"1" to "254"

device.mcu_io_expand_rev

This command reports the revision of the IO Expander MCU application.

Getvar

To report the revision of the MCU IO application:

```
! U1 getvar "device.mcu_io_expand.rev"
```

Values
"1" to "254"

device.mcu_io_expand.desired_rev

This command causes the printer to upgrade the IO Expander MCU to the desired revision if the revision and the desired revision do not match.

Getvar

To specify the desired revision of the IO Expander MCU application:

```
! U1 getvar "device.mcu_io_expand.desired_rev"
```

Values
"1" to "128"

device.orientation

This printer setting determines the installation orientation of the KR403 printer, either horizontal or vertical. It is intended for use only by the system integrator. Modification by an end user can result in unexpected printer behaviour.

Setvar

To instruct the printer to change the presenter loop length:

```
! U1 setvar "device.orientation" "value"
```

Values

- `"0"` printer is installed horizontally
- `"1"` is installed vertically

Default

`"0"` Printer is installed horizontally (original factory default only, value will not change when defaulting the printer with ^JUF).

Getvar

To instruct the printer to respond with the currently set presenter loop length:

```
! U1 getvar "device.orientation"
```

device.pause

This command stops printing after the current label is complete (if one is printing) and places the printer in Pause Mode.

This command is equivalent to ~PP.

Setvar

To stop printing and set the printer in Pause Mode:

```
! U1 setvar "device.pause" ""
```

Values
NA

Default
NA

Do

To stop printing and set the printer in Pause Mode:

```
! U1 do "device.pause" ""
```

Values
NA

Default

NA

device.pnp_option

This command defines the type of Plug and Play response that is sent by the printer after the printer is started. The printer must be restarted for a new PNP string to be reported.

Setvar

To instruct the printer to select the desired Plug and Play response option:

```
! U1 setvar "device.pnp_option" "value"
```

Values

- `"epl"` Eltron Programming Language
- `"zpl"` Zebra Programming Language

Default

`"zpl"`

Getvar

To retrieve the Plug and Play option setting:

```
! U1 getvar "device.pnp_option"
```

Example

This `setvar` example shows the value set to `"epl"`.

```
! U1 setvar "device.pnp_option" "epl"
```

When the `setvar` value is set to `"epl"`, the `getvar` result is `"epl"`.

NOTE: For GT800 printers only: when the printer's Plug and Play string is set to EPL, the KDU Plus displays `'ONNECTION - EPL Printer (DTE)` even when set to ZPL Forms mode. This behavior only affects the display, not the functionality.

device.pmcu.revision

Retrieves the Power Micro-Controller Unit's (PMCU) current revision number.

Getvar

To retrieve the power micro-controller unit's (PMCU) current revision number:

```
! U1 getvar "device.pmcu.revision"
```

device.position.accuracy

This printer setting retrieves/sets the accuracy of the geographic position values.

The units of the value depend upon the location provider that was used to determine the geographic coordinates. Usually, this is specified as a radius, in meters, of confidence around the location coordinates. Often, the radius represents a radius of 68% confidence that the true location lies within the circle, representing one standard deviation.

NOTE: These settings hold the value to which they are set, within the range restrictions. The printer does not perform any calculations, nor associate any meaning such as “meters” or “feet” to the values. The values can be determined by a number of methods, including an Android® or iOS® application communicating with the printer using the smart phone’s geolocation device.

Setvar

To set the accuracy of the geographic position values:

```
! U1 setvar "device.position.accuracy" "value"
```

Values

A decimal number with 6 decimal places, e.g. 25.370000. The value is saved as a double precision floating point number.

- Minimum: `"0"`
- Maximum: `"406700000"`

Getvar

To retrieve the accuracy of the geographic position values:

```
! U1 getvar "device.position.accuracy"
```

Example

```
! U1 setvar "device.position.accuracy" "25.37"
```

device.position.altitude

This printer setting retrieves the altitude above sea level.

The value is in meters above sea level. A positive number indicates a position above sea level. A negative number indicates a position below sea level. The position of sea level depends upon the system used to provide a nominal sea level reference position. This is often the World Geodetic System WGS 84 standard but depends upon the location provider.

Setvar

To set the altitude of the printer above sea level:

```
! U1 setvar "device.position.altitude" "value"
```

Values

A decimal number with 6 decimal places, e.g. 305.100000. The value is saved as a double precision floating point number.

- Minimum: `"-10000"`
- Maximum: `"406700000"`

Getvar

To retrieve the altitude above sea level:

```
! U1 getvar "device.position.altitude"
```

Example

```
! U1 setvar "device.position.altitude" "305.1"
```

device.position.latitude

This printer setting retrieves/sets the geographic latitudinal position.

Setvar

To set the latitude position of the printer:

```
! U1 setvar "device.position.latitude" "value"
```

Values

The value is in decimal degrees from 0.0 to +/-90.0.

Default

`"0.0"`

Getvar

To retrieve the latitude position of the printer:

```
! U1 getvar "device.position.latitude"
```

Values

The value is returned with 6 decimal places. A value of `0.000001` degree is on the order of 0.1 meter of distance on the earth's surface. (The correspondence between degrees and length on the earth's surface varies because the earth is an irregular ellipsoid.)

Example

```
! U1 setvar "device.position.latitude" "6.123456"
```

device.position.longitude

This printer setting retrieves/sets the geographic longitudinal position.

Setvar

To set the longitudinal position of the printer:

```
! U1 setvar "device.position.longitude" "value"
```

Values

- The value is in decimal degrees from 0.0 to +/-180.0.
- The value is saved as a double precision floating point number.

Default

`"0.0"`

Getvar

To retrieve the longitudinal position of the printer:

```
! U1 getvar "device.position.longitude"
```

Values

The value is returned with 6 decimal places. A value of 0.000001 degree is on the order of no more than 0.1 meter of distance on the earth's surface. (The correspondence between degrees and length on the earth's surface varies from approximately 0.1 meter at the equator to 0.0 at the poles.)

Example

```
! U1 setvar "device.position.longitude" "25.123456"
```

device.print_2key

Causes the printer to print the mobile configuration report (commonly known as a 2key report).

Setvar

To cause the printer to print the mobile configuration report:

```
! U1 setvar "device.print_2key" ""
```

The set value is ignored.

Do

To cause the printer to print the mobile configuration report:

```
! U1 do "device.print_2key"
```

device.print_reprogram_2key

This command determines whether the printer will print a configuration label or 2key report after the printer restarts following a firmware update.

When set to `"off"` the printer will not print the configuration label or 2key report after the printer is updated.

Setvar

To set whether a two-key report is printed or not:

```
! U1 setvar "device.print_reprogram_2key" "value"
```

Values

- `"on"`
- `"off"`

Default

`"off"`

Getvar

To retrieve the current setting for processing two-key report after printer firmware is reprogrammed:

```
! U1 getvar "device.print_reprogram_2key"
```

Example

This example disables printing of the two-key report after printer firmware is reprogrammed.

```
! U1 setvar "device.print_reprogram_2key" "off"
```

device.printhead.test.summary

This command retrieves a summary of the printer's printhead test results. This command mimics the results of the ~HQJT ZPL command output.

Getvar

To get the summary of the printer's printhead test results:

```
! U1 getvar "device.printhead.test.summary"
```

Result

A string in the format of A, B, C, D, E.

- A number Element Failure
- B Manual (M) or automatic (A) range
- C first test element
- D last test element
- E failure count

NOTE: The command will return a response for all LOS printers. However only printers that support the head test will display valid values. For all unsupported printers, C and D above will always be 0.

device.printhead.odometer

This command returns the current contents of the odometer. This value is the total number of dots printed over the life of the printhead.

Getvar

To return the current contents of the odometer:

```
! U1 getvar "device.printhead.odometer"
```

device.printhead.test.detail

This command returns the results of the last printhead test for the resistance values. This command is not reported in the ALLCV.

Getvar

To get the summary of the printer's printhead test details:

```
! U1 getvar "device.printhead.test.detail"
```

Result

A comma separated string as given below. Although the content below is shown on individual lines, it is displayed as one line of comma separated values.

- Current Date (as reported by rtc.date),
- Current Time (as reported by rtc.time),
- Odometer Value in cm (as reported by `"odometer.total_print_length"`),
- Part Number of the Printhead,
- Serial Number of the Printhead,
- Resistance profile of each Printhead element

NOTE: The command will return a response for all Link-OS printers. However only printers that support the head test will display valid values. For all unsupported printers, the result is "Test Not Run".

device.product_name_submodel

Retrieves the product name sub-model, which is derived from the Printer Configuration Code (PCC, also known as the SKU).

Getvar

To retrieve the product name sub-model:

```
! U1 getvar "device.product_name_submodel"
```

Values

- "hc" QLn Healthcare printers
- "none" QLn Standard printers and all other printers

device.prompted_network_reset

Reinitializes the wireless radio card and the print server (wired or wireless) when the Wireless or Wireless Plus print server is running. The command also causes any wireless radio card in the printer to re-associate to the wireless network.

This command is equivalent to the ^WR - Reset Wireless on page 400.

Setvar

To set the device prompted reset:

```
! U1 setvar "device.prompted_network_reset" "value"
```

Values

- `"yes"` causes the network to reset
- `"no"` no changes to the network

device.prompted_default_network

This command enables or disables the default device network settings.

Setvar

To enable or disable the device default network settings:

```
! U1 setvar "device.prompted_default_network" "value"
```

Values

- "Y" default the network settings
- "N" do not default the network settings

device.prompted_reset

This command enables the device prompted reset.

Setvar

To enable or disable the device prompted reset:

```
! U1 setvar "device.prompted_reset" "value"
```

Values

- "Y" reset the printer
- "N" do not reset the printer

device.protected_mode

This command retrieves information on the protected mode feature settings.

It returns `"off"` if protected mode is currently disabled or `"on"` if protected mode is currently enabled. Protected mode is enabled if a non-empty protected mode password for the administrator user has been set.

Getvar

To retrieve the status of protected mode:

```
! U1 getvar "device.protected_mode"
```

Values

- `"off"` protected mode is off.
- `"on"` protected mode is on.

Default

`"off"`

device.protected_mode_allowed

This command retrieves information on whether the protected mode feature is supported in the printer operating system.

Getvar

To determine whether the protected mode is allowed:

```
! U1 getvar "device.protected_mode_allowed"
```

Values

- `"yes"` protected mode is allowed.
- `"no"` protected mode is not allowed.

Default

`"yes"`

device.reset

This command instructs the printer to perform a soft reset.

Setvar

To instruct the printer to perform a soft reset:

```
! U1 setvar "device.reset" ""
```

Example

In this example, the `setvar` performs a soft reset.

```
! U1 setvar "device.reset" ""
```

device.reset_button_enable

This command disables the reset button on a printer for situations where the reset button might be accidentally triggered. The default value is "on" and the setting is not defaulted during a printer default.

Setvar

To disable the printer reset button:

```
! U1 setvar "device.reset_button_enable" "off"
```

Values

- "on" the printer reset button is enabled.
- "off" the printer reset button is disabled.

Default

"on"

Getvar

To determine the status of the printer reset button:

```
! U1 getvar "device.reset_button_enable"
```

NOTE: This setting is not defaulted as part of a factory default (^JUF or ^default). The setting is persistent across a power cycle or rest (~JR or device.reset).

device.restore_defaults

This command restores to the default of all settings within the specified SGD branch.

Setvar

To restore the default of all settings within the specified branch:

```
! U1 setvar "device.restore_defaults" "value"
```

Values

- `"ip"` default all parameters in the IP branch
- `"wlan"` default all parameters in the wlan branch
- `"internal_wired"` default all parameters in the internal wired branch

Do

To restore the default of all settings within the specified branch:

```
! U1 do "device.restore_defaults" "value"
```

Values

- `"ip"` default all parameters in the ip branch
- `"wlan"` default all parameters in the wlan branch
- `"internal_wired"` default all parameters in the internal wired branch

Example

These `do` and `setvar` examples restore the network card's WLAN parameters to their default values.

```
! U1 do "device.restore_defaults" "wlan"
! U1 setvar "device.restore_defaults" "wlan"
```

device.rewinder_installed

Determines if a rewind option is installed on the printer.

NOTE: The Rewind Option is not the same as the Liner Take-Up Option.

Getvar

To determine if a rewind option is installed:

```
! U1 getvar "device.rewinder_installed"
```

Values

- `"yes"` a rewind option is installed
- `"no"` no rewind option installed

device.save_2key

Sets or retrieves the current `device.save_2key` setting.

NOTE: The two-key report is a configuration listing used on legacy mobile printers.

Setvar

To set the current `device.save_2key` setting:

```
! U1 setvar "device.save_2key" "value"
```

Values

- `"on"` Two-key diagnostics reports will be saved to Flash memory whenever a two-key report is printed. The file will be named 2KEY.TXT.
- `"off"` Two-key reports will not be saved to Flash memory.
- `"now"` This choice can be used to generate a two-key diagnostics report on demand and save it to Flash memory (save only, does not print). This choice does not alter the "on"/"off" state of this SGD.

Default

`"on"`

Getvar

To retrieve the current device.save_2key setting:

```
! U1 getvar "device.save_2key"
```

Example

This example instructs the printer to generate a two-key diagnostics report and save it to Flash memory.

```
! U1 setvar "device.save_2key" "now"
```

device.sensor_select

Determines which media sensor will be used.

This command is similar to the ^JS ZPL command.

Setvar

To determine which media sensor will be used:

```
! U1 setvar "device.sensor_select" "value"
```

Values

- `"reflective"`
- `"transmissive"`

Getvar

To retrieve which media sensor is used:

```
! U1 getvar "device.sensor_select"
```

device.sensor_profile

This command sets the printer's sensor profile output destination.

Setvar

To set the sensor profile of the printer:

```
! U1 setvar "device.sensor_profile" "value"
```

Values

If a valid setting is not specified, then the sensor profile command is ignored.

`"print"`	forces all subsequent ~jg output to be printed on media.
`"store"`	forces all subsequent ~jg output to be stored on the E drive.
`"usb_host"`	forces all subsequent ~jg output to be stored on a USB Stick. If the sensor profile value of "usb_host" is selected and a USB Stick is not inserted at the time, then the ~jg is issued. An Acknowledged Alert is displayed on printers that support an LCD. If the Sensor Profile value of "usb_host" is selected and a USB Stick is not inserted at the time, then the ~jg is issued and the printer does not support an LCD, then the error is ignored.
`"reply"`	forces all subsequent ~jg output to be returned to the host on the same port that the command is issued.
`"display"`	forces all subsequent ~jg output to be stored on the LCD. If the Sensor Profile value of "display" is selected and the printer does not have at least a 240x128 pixel display, then the setting is ignored. The UI SRS defines the actual content to be displayed if the "display" option is selected.

Default

`"print"`

Getvar

To return the sensor profile values:

```
! U1 getvar "device.sensor_profile"
```

device.serial_number.option_board_date

Returns the date the option board was made.

NOTE: This command is functional only on printers that had their option board manufacturing date programmed when they were created. Older printers that do not have the option board creation date programmed will return a "?" or empty string.

Getvar

To return the date the option board was made:

```
! U1 getvar "device.serial_number.option_board_date"
```

Result

The date in mm/dd/yyyy format.

device.serial_numbers.control_panel_date

This command returns the date the control panel was made.

NOTE: This command is functional only on printers that had their control panel manufacturing date programmed when they were created. Older printers that do not have the control panel creation date programmed will return a "?" or empty string.

Getvar

To return the date the control panel was made:

```
! U1 getvar "device.serial_numbers.control_panel_date"
```

Result

The date in mm/dd/yyyy format.

device.serial_numbers.mlb_date

Returns the date the main logic board (MLB) was made.

NOTE: This command is functional only on printers that had their MLB manufacturing date programmed when they were created. Older printers that do not have the MLB creation date programmed will return a "?" or empty string.

Getvar

To return the date the main logic board (MLB) was made:

```
! U1 getvar "device.serial_numbers.mlb_date"
```

Result

The date in mm/dd/yyyy format.

device.serial_numbers.processor

Returns the unique main processor ID.

Getvar

To have the printer return the current setting value:

```
! U1 getvar "device.serial_numbers.processor"
```

device.serial_numbers.applicator_option_board_date

This command retrieves the applicator option board date.

For printers that do not store this value, the printer returns an empty string.

Getvar

To return the current setting:

```
! U1 getvar "device.serial_numbers.applicator_option_board_date"
```

Example

In this example, the printer reports the application option boards date.

```
! U1 getvar "device.serial_numbers.applicator_option_board_date"
"12/31/2014"
```

device.serial_numbers.wired_ethernet_option_board

This command retrieves the serial number of the wired Ethernet option board if it is installed in the printer. For printers that do not store this value, the printer returns an empty string.

Getvar

To return the current setting:

```
! U1 GETVAR "device.serial_numbers.wired_ethernet_option_board"
```

device.serial_numbers.wired_ethernet_option_board_date

This command retrieves the Ethernet option board date.

For printers that do not store this value, the printer returns an empty string.

Getvar

To return the current setting:

```
! U1 getvar "device.serial_numbers.ethernet_option_board_date"
```

Example

In this example, the `getvar` returns the current date of the Ethernet option board.

```
! U1 getvar "device.serial_numbers.ethernet_option_board_date"
"12/31/2014"
```

device.serial_numbers.applicator_option_board

This command retrieves the serial number of the applicator option board if it is installed in the printer. For printers that do not store this value, the printer returns an empty string.

Getvar

To return the current setting:

```
! U1 getvar "device.serial_numbers.applicator_option_board"
```

device.serial_numbers.cutter

This command returns the serial number of the cutter, if installed.

For printers that do not store this value, the printer returns an empty string.

Getvar

To return the serial number of the cutter board:

```
! U1 getvar "device.serial_numbers.cutter"
```

Value

A hexadecimal representation of the control panel serial number.

Example

In this example, the `getvar` returns the serial number of the cutter board.

```
! U1 getvar "device.serial_numbers.cutter"
```

```
"0123456789ABCDEF"
```

device.serial_numbers.cutter_date

This command returns the cutter date. For printers that do not store this value, the printer returns an empty string.

Getvar

To return the manufacturing date of the cutter board:

```
! U1 getvar "device.serial_numbers.cutter_date"
```

Example

In this example, the getvar returns the manufacturing date of the cutter board.

```
! U1 getvar "device.serial_numbers.cutter_date"
```

```
"12/31/2014"
```

device.serial_numbers.printhead

This command returns the serial number field of the printhead that is installed in the printer.

Getvar

To return the serial number field of the printhead that is installed in the printer:

```
! U1 getvar "device.serial_numbers.printhead"
```

device.serial_numbers.printhead_date

This command retrieves the printhead date. For printers that do not store this value, the printer returns an empty string.

Getvar

To return the printhead date:

```
! U1 getvar "device.serial_numbers.printhead_date"
```

Example

In this example, the `getvar` returns the manufacturing date of the printhead.

```
! U1 getvar "device.serial_numbers.printhead_date" "12/31/2014"
```

device.serial_numbers.usb_host_option_board_date

This command retrieves the USB Host option board date.

For printers that do not store this value, the printer returns an empty string.

Getvar

To return the USB host option board date:

```
! U1 getvar "device.serial_numbers.usb_host_option_board_date"
```

Example

In this example, the `getvar` returns the manufacturing date of the USB host option board.

```
! U1 getvar "device.serial_numbers.usb_host_option_board_date" "12/31/2014"
```

device.serial_numbers.usb_host_option_board

This command retrieves the serial number of the USB host option board. For printers that do not store this value, the printer returns an empty string.

Getvar

To return the USB host option board serial number:

```
! U1 getvar "device.serial_numbers.usb_host_option_board"
```

device.serial_numbers.parallel_option_board

This command retrieves the serial number of the parallel port option board. For printers that do not store this value, the printer returns an empty string.

Getvar

To return the parallel port option board serial number:

```
! U1 getvar "device.serial_numbers.parallel_option_board"
```

device.serial_numbers.parallel_option_board_date

This command retrieves the parallel port option board date.

For printers that do not store this value, the printer returns an empty string.

Getvar

To return the parallel option board date:

```
! U1 getvar "device.serial_numbers.parallel_option_board_date"
```

Example

In this example, the `getvar` returns the manufacturing date of the parallel port option board.

```
! U1 getvar "device.serial_numbers.parallel_option_board_date"
"12/31/2014"
```

device.set_clock_to_build_date

Enables or disables a lower bound of the firmware build date for the rtc.date SGD.

If enabled, when the printer powers up and it finds and RTC date earlier than the firmware build date, it will set the RTC date to the firmware build date.

Setvar

To enable or disable a lower bound of the firmware build date for the rtc.date SGD:

```
! U1 setvar "device.set_clock_to_build_date" "value"
```

Values

- `"enabled"`
- `"disabled"`

Default

`"enabled"`

Getvar

To retrieve the firmware build date for the `rtc.date` SGD:

```
! U1 getvar "device.set_clock_to_build_date"
```

device.slot_1

This command retrieves the type of board installed in the bottom slot of a ZT400 series printer, or in the single expansion slot of a ZT200 series printer.

Getvar

To retrieve the type of board installed in the bottom slot of a ZT400 series printer:

```
! U1 getvar "device.slot_1"
```

Values

- `"empty"` no board installed
- `"parallel"` a parallel communications board is installed
- `"wired"` a wired PrintServer board is installed
- `"wireless"` a wireless PrintServer board is installed

device.slot_2

This command retrieves the type of board installed in the bottom slot of a ZT400 series printer.

Getvar

To retrieve the type of board installed in the bottom slot:

```
! U1 getvar "device.slot_2"
```

Values

- `"empty"` no board installed
- `"parallel"` a parallel communications board is installed
- `"wired"` a wired PrintServer board is installed
- `"wireless"` a wireless PrintServer board is installed

device.super_host_status

This command returns printer description information in XML format. The printer returns information on format parameters, object directories, individual object data, and print status information.

This command is equivalent to the ^HZA ZPL command.

Getvar

To return printer description information in XML format:

```
! U1 getvar "device.super_host_status"
```

Result

Information on format parameters, object directories, individual object data, and print status information.

device.syslog.clear_log

This setting clears the local syslog.entries SGD. Any log messages previously sent to an ip address are not changed.

Setvar

To clear the local syslog file:

```
! U1 setvar "device.syslog.clear_log" ""
```

Values
NA

Default
NA

Do

To clear the local syslog file:

```
! U1 do "device.syslog.clear_log" ""
```

Values
NA

Default
NA

device.syslog.configuration

This setting specifies the location for the syslog messages to be recorded. The location may be either on the printer, or a syslog server IP address.

Setvar

To specify the location for the syslog messages to be recorded:

```
! U1 setvar "device.syslog.configuration" "value"
```

Values

A list of configuration entries, limited to 1000 characters. Entries must be in the form of `"severity,destination"` and delimited with a semi-colon.

SEVERITY - The severity levels, in decreasing severity order:

- emerg
- alert
- crit
- err
- warning
- notice
- info
- debug

When you specify the severity level, the lowest specified severity and all severity levels above it will be recorded. For example, if you specify `debug`, you will get all severity level reports. If you specify `crit`, you will get only `crit`, `alert`, and `emerg` severity reports.

DESTINATION - `"local"` or a syslog server IP address

When configuring the local syslog report, the first local entry is used and duplicate requests to local are ignored. To configure remote syslog messages you will first need a syslog server to accept them.

Default

`""`

Getvar

To retrieve the configuration string setting:

```
! U1 getvar "device.syslog.configuration"
```

Example 1

This example has emergency syslog messages being sent to an IP location, debug (and all higher severity) syslog messages to another IP address, and critical and higher syslog messages to local storage (either a file or SGD).

```
! U1 setvar "device.syslog.configuration"
 "emerg,128.168.0.1;debug,192.168.0.2;crit,local;"
```

Example 2

This example will only report emegency syslog messages to the local file, and ignore the duplicate location request for critical and higher reports.

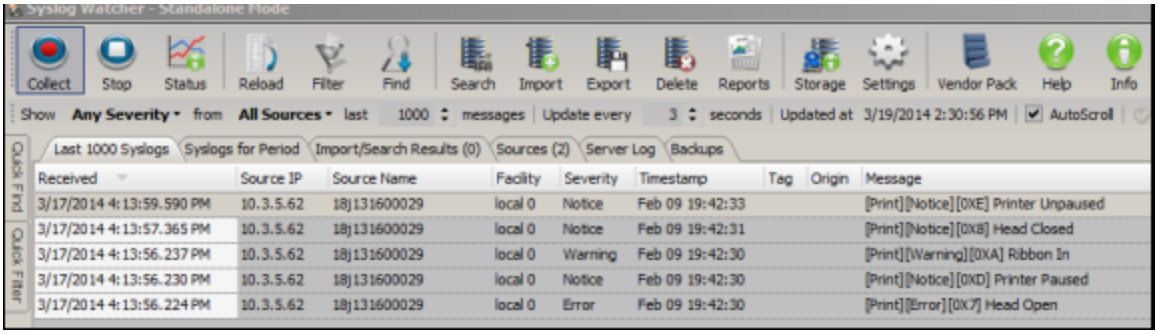
```
! U1 setvar "device.syslog.configuration" "emerg,local;crit,local;"
```

Example 3

```
This is an example of a syslog report stored at E:SYSLOG.TXT. Note that
 device.syslog.save_local_file must be enabled.
Feb 17 14:28:17: [Power][Informational][0X14] Power On
Feb 17 14:28:19: [Print][Informational][0XF] PQ Job Completed
Feb 17 14:28:20: [Print][Informational][0XF] PQ Job Completed
Feb 17 14:28:20: [Weblink][Informational][0X1005] Weblink disabled
Feb 17 14:28:34: [Network][Notice][0X1C] Cold Start
```

Example 4

This is an example of a syslog report from a syslog server application monitoring an IP address.



device.syslog.enable

This printer setting enables syslog messages. The destination of syslog messages is specified in `device.syslog.configuration`.

Setvar

To enable or disable syslog:

```
! U1 setvar "device.syslog.enable" "value"
```

Values

- `"on"`
- `"off"`

Default

`"off"`

Getvar

To retrieve if the syslog is enabled:

```
! U1 getvar "device.syslog.enable"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "device.syslog.enable" "on"
```

device.syslog.entries

This printer setting displays previously sent syslog messages. If there are no previously sent syslog messages, an empty string is returned.

Getvar

To display previously sent syslog messages:

```
! U1 getvar "device.syslog.entries"
```

The format of each syslog message includes the printer feature, the severity level, the unique message code, and the unique English message. This allows for more advanced systems administrators to filter particular messages of interest. Syslog currently supports unique messages for most printer alerts, WebLink, and some USB Host messages.

Example

This `getvar` example shows the value of the syslog file.

```
! U1 getvar "device.syslog.entries"
```

returns

```
Feb 17 14:28:17: [Power][Informational][0X14] Power On
Feb 17 14:28:19: [Print][Informational][0XF] PQ Job Completed
Feb 17 14:28:20: [Print][Informational][0XF] PQ Job Completed
Feb 17 14:28:20: [Weblink][Informational][0X1005] Weblink disabled
Feb 17 14:28:34: [Network][Notice][0X1C] Cold Start
```

device.syslog.log_max_file_size

This printer setting specifies the maximum size of the local syslog file.

Setvar

To set the maximum syslog file size to the specified value:

```
! U1 setvar "device.syslog.log_max_file_size" "value"
```

Values

A numerical value between 10000 and 400000

Default

`"10000"`

Getvar

To return the maximum allowed size of the syslog file:

```
! U1 getvar "device.syslog.log_max_file_size"
```

Example

This `setvar` example shows the value set to `"200000"`.

```
! U1 setvar "device.syslog.log_max_file_size" "200000"
```

device.syslog.save_local_file

This command saves the contents of the local syslog to E:SYSLOG.TXT. The local destination must be specified in `device.syslog.configuration`.

Setvar

To specify whether to save the contents of the local syslog file to E:SYSLOG.TXT:

```
! U1 setvar "device.syslog.save_local_file" "value"
```

Values

- `"yes"` the local syslog is saved to E:SYSLOG.TXT
- `"no"` the local syslog is not saved

Default

`"no"`

Getvar

To display the setting for saving the local syslog file to E:SYSLOG.TXT:

```
! U1 getvar "device.syslog.save_local_file"
```

Example

This `setvar` example shows the value set to `"yes"`.

```
! U1 setvar "device.syslog.save_local_file" "yes"
```

device.applicator.data_ready_activation

Sets whether the applicator port DATA READY signal is asserted for all formats, or only for printing formats.

Setvar

To set whether the applicator port DATA READY signal is asserted for all formats, or only for printing formats:

```
! U1 setvar "device.applicator.data_ready_activation" "value"
```

Values

- `"print"` indicates the data ready signal is activated on printing labels only.
- `"format"` indicates the data ready signal is activated on all formats.

Default

`"format"`

Getvar

To return the data ready activation value:

```
! U1 getvar "device.applicator.data_ready_activation"
```

Result

"print"

Example

The setvar example shows the data ready signal activated on printing labels only.

```
! U1 setvar "device.applicator.data_ready_activation" "print"
```

device.applicator.error_on_pause

Sets whether device applicator errors will be displayed.

Setvar

To set whether device applicator errors will be displayed:

```
! U1 setvar "device.applicator.error_on_pause" "value"
```

Values

`"enabled"` device applicator errors will be displayed, and SERVICE REQUIRED will be asserted.

`"disabled"` device applicator errors will not be displayed.

Default

`"enabled"`

Example

```
! U1 setvar "device.applicator.error_on_pause" "enabled"
```

Getvar

To return the current setting value:

```
! U1 setvar "device.applicator.error_on_pause" "enabled"
```

device.applicator.start_print_mode

Selects the applicator port START PRINT mode of operation.

Setvar

To select the start print mode of operation:

```
! U1 setvar "device.applicator.start_print_mode" "value"
```

Values

- `"level"` the Start Print signal does not need to be de-asserted to print the next label. As long as the Start Print signal is low and a label is formatted, a label prints.
- `"pulse"` the Start Print signal must be de-asserted before it can be asserted for the next label

Default

```
"pulse"
```

Getvar

To retrieve the current setting value:

```
! U1 getvar "device.applicator.start_print_mode"
```

Result

```
"level"
```

Example

In the setvar example below, the `"level"` start print mode of operation is set.

```
! U1 setvar "device.applicator.start_print_mode" "level"
```

device.applicator.voltage

Sets the output voltage of the applicator board. The value will not take effect until a reboot.

Setvar

To set the output voltage of the applicator board:

```
! U1 setvar "device.applicator.voltage" "value"
```

Values

- `"0"` indicates off
- `"5"` indicates 5V
- `"24"` indicates 24V

Default

```
"disabled"
```

Getvar

To retrieve the current setting value:

```
! U1 getvar "device.applicator.voltage"
```

Result

```
"5"
```

Example

In the setvar example below, the output voltage of the applicator board is set to `"5"`.

```
! U1 setvar "device.applicator.voltage" "5"
```

device.unique_id

This command retrieves the printer identifier.

Getvar

To retrieve the established printer identifier:

```
! U1 getvar "device.unique_id"
```

Example

In this example, assuming the printer's unique ID is 12345, the `getvar` shows `"12345"`.

```
! U1 getvar "device.unique_id"
```

device.unpause

This command causes a printer in Pause Mode to resume printing. The operation is identical to pressing **PAUSE** on the control panel of the printer when the printer is already in Pause Mode.

This command is equivalent to ~PS.

Setvar

To cause the printer in the pause mode to resume printing:

```
! U1 SETVAR "device.pause" ""
```

Values
NA

Default
NA

Do

To cause the printer in the pause mode to resume printing:

```
! U1 DO "device.pause" ""
```

Values
NA

Default
NA

device.uptime

This command identifies the amount of time the printer has been powered on. The string format is: xx days, xx hours, xx minutes, and xx seconds.

Getvar

To retrieve the amount of time the print has been powered on:

```
! U1 getvar "device.uptime"
```

Example

In this example, the `getvar` retrieves the amount of time the printer has been turned on.

```
! U1 getvar "device.uptime"
"00 days 02 hours 45 mins 30 secs"
```

device.user_p1

This command saves and retrieves user specified values.

Setvar

To instruct the printer to set user parameters:

```
! U1 setvar "device.user_p1" "value"
```

Values

An alphanumeric text string (1 - 20).

Default

""

Getvar

To retrieve user specified parameters:

```
! U1 getvar "device.user_p1"
```

Example

This `setvar` example shows the value set to `"test"`.

```
! U1 setvar "device.user_p1" "test"
```

When the `setvar` value is set to `"test"`, the `getvar` result is `"test"`.

device.user_p2

This command saves and retrieves user specified values.

Setvar

To instruct the printer to set user parameters:

```
! U1 setvar "device.user_p2" "value"
```

Values

An alphanumeric text string (1 - 20).

Default

""

Getvar

To save and retrieve user specified parameters:

```
! U1 getvar "device.user_p2"
```

Example

This `setvar` example shows the value set to `"test"`.

```
! U1 setvar "device.user_p2" "test"
```

When the `setvar` value is set to `"test"`, the `getvar` result is `"test"`.

device.user_vars.set_range

This command compliments the `device.user_vars.create` command, allowing a user to change the range of a user-created SGD variable. It has a similar syntax to `device.user_vars.create` with the exception that no default is specified.

Setvar

To change the range of a user-created variable:

```
! U1 setvar "device.user_vars.set_range" "name:type:range"
```

Values

- `name` identifies the name of the SGD to modify
- `type` must be the same type for `name` as when it was created
- `range` is x-y (for all but UPDOWNCHOICES and CHOICES) or a,b,c,d (for CHOICES and UPDOWNCHOICES)

  If no range is specified then it will delete whatever range is currently specified.

Default

NA

Example

This example modifies my_var to: `device.user_vars.my_var : b , Choices: a,b,c,d,e`

device.user_vars.create

This command creates a user variable with the specified name, type, range, and default value. The root SGD location for user variables is "device.user_vars".

Setvar

To instruct the printer to create a user variable with the specified parameters:

```
! U1 setvar "device.user_vars.create" "name:type:range:defaultValue"
```

Values

- name is the name of the SGD to appear in device.user_vars. The name can be anything from 1 to 64 printable ASCII characters. Any '.' within the name will be replaced with '_'. (e.g. "john.doe" will be changed to "john_doe". The name must be unique in the device.user_vars branch or it will not be created. The name will be converted to lower case.
- type is one of STRING, INTEGER, DOUBLE, CHOICES, UPDOWNCHOICES, UPDOWNINTEGER, UPDOWNDOUBLE. The type must be one of these types or the variable will not be created.
  - STRING - For strings, the range indicates the min/max length of the data that can be stored. If the range is left blank, the default range is a string length of 0-1024. There is no maximum string length, however, if large data is placed into the variables the user should be aware that system memory and performance will be affected. Strings larger than available system memory will not be stored. Values should attempt to stay around, or below, 5K.
  - INTEGER/UPDOWNINTEGER - For integers the range can be any number expressed by a 32-bit integer, signed or unsigned. If the range is left blank then a range of -32768 to 32767 will be used.
  - DOUBLE/UPDOWNDOUBLE - A double can be any value within the range of +/–1.7e308. If the range is left blank then a range of -32768.0 to 32767.0 will be used.
  - CHOICES/UPDOWNCHOICES - Choices must be specified in a comma delimited list. The range cannot be blank if the type is CHOICES or UPDOWNCHOICES.
- range is expressed as x-y. The range of a variable depends upon the type. Some types will create default ranges, while others will fail to be created if the range is invalid or not specified.
- default is the default value for the variable. The value must fall within the range specified or the variable will not be created. If the type is INTEGER, UPDOWNINTEGER, DOUBLE, UPDOWNDOUBLE the default value will be 0 if not specified. For STRING the default value will be an empty string if it is not specified. CHOICES and UPDOWNCHOICES must have a default value and it must be one of the choices within the specified range.

  All four parts of the creation string must be present (some can be empty) meaning that there must be three delimiter characters (':') present. There is no error shown or indicated otherwise when the variable is not created for some reason. If the variable is not created one of the rules listed above has been violated.

  Any user variables will be deleted from the device.user_vars branch on a power cycle (they won't be recreated on the next power up).

  Defaulting the user_vars branch will restore the consumers back to their defaulted values and will not remove them from the user_vars branch.

Example

To create a user variable named `userVar1` that is an integer with a minimum of 1, a maximum of 10, and a default/initial value of 5, issue this command:

```
! U1 setvar "device.user_vars.create" "userVar1:INTEGER:1-10:5"
```

After issuing the above “create” command the `device.user_vars.userVar1` SGD will be present in an ALLCV response.

After issuing the above “create” command the `device.user_vars.userVar1` SGD may be set via:

```
! U1 setvar "device.user_vars.userVar1" "2"
```

After issuing the above “create” command the `device.user_vars.userVar1` SGD may be retrieved via:

```
! U1 getvar "device.user_vars.userVar1"
```

device.xml.enable

This command enables and disables language parsing support for XML. When enabled (on), the printer will parse both ZPL and XML. When disabled (off), the printer will not parse XML data.

Setvar

To instruct the printer to disable or enable the language parsing support for XML:

```
! U1 getvar "device.xml.enable" "value"
```

Values

- `"on"` enables language parsing support for XML
- `"off"` disables language parsing support for XML

Default

`"on"`

Getvar

To enable and disable language parsing support for XML:

```
! U1 getvar "device.xml.enable"
```

Example

This `setvar` example shows the language parsing support for XML set to `"on"`.

```
! U1 setvar "device.xml.enable" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is language parsing support for XML set to `"on"`.

device.feature.lighted_arrows

Indicates if the Lighted Arrows hardware is installed.

Getvar

To retrieve if the Lighted Arrow hardware is installed or not:

```
! U1 getvar "device.feature.lighted_arrows"
```

Result

- `"not available"` indicates lights are not available for this printer
- `"present"` indicates lights are installed
- `"not present"` indicates lights are not installed

device.light.ribbon_path_brightness

Sets the brightness level for the ribbon path LED.

Setvar

To set the brightness level for the ribbon path LED:

```
! U1 setvar "device.light.ribbon_path_brightness" "value"
```

Values

- `"off"`
- `"low"`
- `"medium"`
- `"high"`

Default

`"high"`

Example

```
! U1 setvar "device.light.ribbon_path_brightness" "low"
```

Getvar

To retrieve the current setting value:

```
! U1 getvar "device.light.ribbon_path_brightness"
```

Result

```
"low"
```

device.light.media_path_brightness

Sets the brightness level for the media path LED.

Setvar

To set the brightness level for the media path LED:

```
! U1 setvar "device.light.media_path_brightness" "value"
```

Values

- `"off"`
- `"low"`
- `"medium"`
- `"high"`

Default

`"high"`

Example

```
! U1 setvar "device.light.media_path_brightness" "low"
```

Getvar

To retrieve the current setting value:

```
! U1 getvar "device.light.media_path_brightness"
```

Result

```
"low"
```

device.zuid

Reports a unique ID for the printer that is used for the MQTT topic and MQTT client ID. The zuid is created by the printer the first time it is needed and remains the same unless the printer is decommissioned, at which point a new unique ID is created.

Getvar

To retrieve the unique ID for the printer:

```
! U1 getvar "device.zuid"
```

Values

A unique hyphen-free string.

display.backlight

This parameter determines if the printer display backlight will be active. Valid only on printers with a display installed.

Setvar

To instruct the printer to turn or off the backlight display:

```
! U1 setvar "display.backlight" "value"
```

Values

- `"on"`
- `"off"`

Default

```
"on"
```

Getvar

To return if the display backlight is on or off:

```
! U1 getvar "display.backlight
```

Example

This example sets the backlight display to `"on"`.

```
! U1 setvar "display.backlight" "on"
```

display.backlight_on_time

This command sets the amount of time before the backlight turns off. Valid only on printers with a display installed.

Setvar

To set the display length in seconds:

```
! U1 setvar "display.backlight_on_time" "time"
```

Values

`0-8191`

Default

`0`

NOTE: If the value is set to 0, the backlight will remain on.

Getvar

To return the display length in seconds:

```
! U1 getvar "display.backlight_on_time"
```

Example

This `setvar` example shows the value set to one minute (60 seconds).

```
! U1 setvar "display.backlight_on_time" "60"
```

display.batch_counter

Sets whether batch counters will be displayed on the printer's control panel.

Setvar

To set whether batch counters will be displayed on the printer's control panel:

```
! U1 setvar "display.batch_counter" "value"
```

Values

- `"enabled"` indicates batch counters will be displayed
- `"disabled"` indicates batch counters will not be displayed

Default

`"disabled"`

Example

```
! U1 setvar "display.batch_counter" "enabled"
```

Getvar

To retrieve the current setting value:

```
! U1 getvar "display.batch_counter"
```

Result

`"enabled"`

display.bluetooth.mac

This command enables or disables the bluetooth MAC address display on printers.

Setvar

To set the command:

```
! U1 setvar "display.bluetooth.mac" "on"
```

Values

`"off"` indicates that the BT Mac address is not displayed

`"on"` indicates that the BT Mac Address is displayed

Default

`"off"`

Example

In this example, the setvar turns on the bluetooth MAC address display:

```
! U1 setvar "display.bluetooth.mac"  "on"
```

Getvar

To view the current bluetooth MAC address display setting:

```
! U1 getvar "display.bluetooth.mac"
```

display.calibrate

This command initiates the UI screen calibration if the user cannot enter the UI screen calibration from the menu.

Setvar

To set the command:

```
! U1 setvar "display.calibrate" ""
```

Values
NA

Do

To initiate the UI screen calibration:

```
! U1 do "display.calibrate" ""
```

Values
NA

NOTE: This command only works for printers with a touch display.

display.language

This command sets the display language for the control panel. This is equivalent to the ^KL ZPL command.

Setvar

To set the display language for the front panel:

```
! U1 setvar "display.language" "language"
```

Values

- "english"
- "spanish"
- "french"
- "german"
- "italian"
- "norwegian"
- "portuguese"
- "swedish"
- "danish"
- "spanish2" (same as Spanish)
- "dutch"
- "finnish"
- "japanese"
- "korean"
- "simplified chinese"
- "traditional chinese"
- "russian"
- "polish"
- "czech"
- "romanian"

Default

"english"

Getvar

To return the currently set display language:

```
! U1 getvar "display.language"
```

Example

This `setvar` example shows the value set to `"dutch"`.

```
! U1 setvar "dsplay.language" "dutch"
```

display.load_card

This command loads a specific card from a WML file.

Setvar

To load a specific card from a WML file:

```
! U1 setvar "display.load_card" "value"
```

Values

A valid WML filename and card within the WML filename.

NOTE: The card name is case sensitive.

Example

```
! U1 setvar "display.load_card" "BLUETOOTH.WML#bluetooth2"
```

display.password.level

Controls when to display the password WML card on the LCD display.

Setvar

To control when to display the password WML card on the LCD display:

```
! U1 setvar "display.password.level" "value"
```

Values

- `"all"` The user will always be prompted to enter a password when a field is to be modified.
- `"none"` The user will not be prompted to enter a password on the LCD display.
- `"selected"` The user will be prompted to enter a password only if the WML card contains a password ="on" attribute and the user attempts to change a setting.

Default

- `"selected"` QLn420, QLn320 Healthcare, and QLn220 Healthcare
- `"none"` all other platforms

Getvar

To retrieve the current setting value:

```
! U1 getvar "display.password.level"
```

display.root_wml

This command specifies which control file is first processed by the front panel of the printer.

Setvar

To specify which control file is first processed by the printer's front panel:

```
! U1 setvar "display.root_wml" "value"
```

Values

Any file name with a maximum of 128 characters in length.

Default

- `Z:INDEX420.WML` for the QLn420 printers
- `Z:INDEX320.WML` for the QLn220 and QLn320 printers
- `Z:INDEX.WML` for all other printers

If the value is `""` on power-up, then `Z:INDEX.WML` is used.

Getvar

To retrieve the current setting value:

```
! U1 getvar "display.root_wml"
```

display.text

This command retrieves the text data that is being used on the printer's LCD.

Getvar

To retrieve the text data that appears on the printer's LCD:

```
! U1 getvar "display.text"
```

Example

In this example, the getvar displays text content that appears on the printer's LCD.

```
! U1 getvar "display.text"
"PRINTER READY  V60.16.4Z"
```

ezpl.head_close_action

This command sets what happens to the media after the printhead is closed and the printer is taken out of pause.

This command is similar to the ^MF ZPL command.

Setvar

To instruct the printer on which action to perform when the printhead is closed:

```
! U1 setvar "ezpl.head_close_action" "value"
```

Values

- `"feed"` feed to the first web after sensor
- `"calibrate"` is used to force a label length measurement and adjust the media and ribbon sensor values.
- `"length"` is used to set the label length. Depending on the size of the label, the printer feeds one or more blank labels.
- `"no motion"` no media feed
- `"short cal"` short calibration

Default

`"calibrate"`

Getvar

To return the current set of action to be performed when the printhead is closed:

```
! U1 getvar "ezpl.head_close_action"
```

Example

This `setvar` example sets the calibration method to short calibration.

```
! U1 setvar "ezpl.head_close_action" "short cal"
```

ezpl.label_length_max

This command sets the maximum label length in inches. This command is equivalent to the ^ML ZPL command.

Setvar

To set the maximum label length in inches:

```
! U1 setvar "ezpl.label_length_max" "value"
```

Values

1.0 to 39.0

Default

"39"

Getvar

To retrieve the current maximum label length setting in inches:

```
! U1 getvar "ezpl.label_length_max"
```

Example

This example sets the label length to 6.2 inches.

```
! U1 setvar ezpl.label_length_max" "6.2"
! U1 setvar ezpl.label_length_max" "14"
```

Values can be expressed to one decimal place.

ezpl.label_sensor

This command sets the paper out threshold value.

Setvar

To set the paper out threshold value:

```
! U1 setvar "ezpl.label_sensor" "value"
```

Values

`"0" to "255",` integer values only

Default

`"70"`

Getvar

To retrieve the currently set paper out threshold value:

```
! U1 getvar "ezpl.label_sensor"
```

Example

This `setvar` example shows the value set to `50`.

```
! U1 setvar "ezpl.label_sensor" "50"
```

ezpl.manual_calibration

This command initiates a manual calibration sequence.

Setvar

To instruct the printer to initiate a manual calibration:

```
! U1 setvar "ezpl.manual_calibration" ""
```

Values
NA

Default
NA

Do

To instruct the printer to initiate a manual calibration:

```
! U1 do "ezpl.manual_calibration" ""
```

Values
NA

Default
NA

ezpl.media_type

This command specifies the media type being used. This command is similar to the ^MN ZPL command.

Setvar

To set the media type used in the printer:

```
! U1 setvar "ezpl.media_type" "value"
```

Values

- "continuous"
- "gap/notch"
- "mark"

Default

"gap/notch"

Getvar

To return the current media type setting:

```
! U1 getvar "ezpl.media_type"
```

Example

This setvar example sets the media type to "continuous".

```
! U1 setvar "ezpl.media_type" "continuous"
```

ezpl.power_up_action

This command sets what happens to the media when the printer is powered on. This command is similar to the ^MF ZPL command.

Setvar

To set the media motion and calibration setting at printer power up:

```
! U1 setvar "ezpl.power_up_action" "value"
```

Values

- "calibrate"
- "feed"
- "length"
- "no motion"
- "short cal"

Default

"calibrate"

Getvar

To return the current power up media motion and calibration settings:

```
! U1 getvar "ezpl.power_up_action"
```

Example

This setvar example sets the power up calibration setting to "length".

```
! U1 setvar "ezpl.power_up_action" "length"
```

ezpl.print_method

This command sets the print method. This command is similar to `^MT`.

Setvar

To set the print method:

```
! U1 setvar "ezpl.print_method" "value"
```

Values

- `"thermal trans"`
- `"direct thermal"`

Default

`"thermal trans"`

Getvar

To retrieve the current print method setting:

```
 ! U1 getvar "ezpl.print_method"
```

Example

This `setvar` example sets the print method to `"direct thermal"`.

```
! U1 setvar ezpl.print_method" "direct thermal"
```

ezpl.print_width

This command sets the print width of the label in dots.

Setvar

To set the print width:

```
! U1 setvar "ezpl.print_width" "value"
```

Values

Any printhead width.

Default

The width of the printhead.

Getvar

To retrieve the current print width setting:

```
! U1 getvar "ezpl.print_width"
```

Example

This `setvar` example sets the print width value to 3.

```
! U1 setvar "ezpl.print_width" "3"
```

ezpl.reprint_mode

This command turns on/off the reprint mode.

Setvar

To instruct the printer to turn on or off reprint mode:

```
! U1 setvar "ezpl.reprint_mode" "value"
```

Values

- "on"
- "off"

Default

"off"

Getvar

To retrieves the current setting for reprint mode:

```
! U1 getvar "ezpl.reprint_mode"
```

Example

This `setvar` example turns reprint mode on.

```
! U1 setvar "ezpl.reprint_mode" "on"
```

ezpl.take_label

This command sets the brightness of the take-label sensor.

Setvar

To set the take-label sensor brightness:

```
! U1 setvar "ezpl.take_label" "value"
```

Values

"0" to "255"

Default

"0"

Getvar

To retrieve the take-label sensor brightness:

```
U! getvar "ezpl.take_label"
```

Example

This example sets the take-label sensor brightness to 175.

```
! U1 setvar "ezpl.take_label" "175"
```

ezpl.take_label_calibration

This command initiates the calibration sequence of the take-label sensor. The printer returns the following statuses during the calibration process: `starting`, `inProgress`, and `complete`.

Setvar

To initiate the take-label sensor calibration sequence:

```
! U1 setvar "ezpl.take_label_calibration"
```

Values

NA

Default

NA

ezpl.tear_off

This command retrieves the tear-off position.

Setvar

To set the tear-off position:

```
! U1 setvar "ezpl.tear_off" "value"
```

Values

"-120" to "120"

Default

"-120" to "120"

Getvar

To retrieve the currently set tear-off position:

```
! U1 getvar "ezpl.tear_off"
```

Example

This `setvar` example sets the tear-off value to 110.

```
! U1 setvar "ezpl.tear_off" "110"
```

file.capture_response.begin

This command enables a file capture mode that will echo all return data to a file.

Do

Use this command in conjunction with `file.capture_response.end` and `file.capture_response.destination`. The begin and end commands must be issued from the same port.

To enable file capture mode:

```
! U1 do "file.capture_response.begin" ""
```

Values
NA

Default
NA

file.capture_response.destination

Sets where a log file is saved that logs data coming from the language parsers to a host (where data was originally sent).

Setvar

Use `file.capture_response.begin` and `file.capture_response.end` to start and stop logging.

To set where a log file is saved:

- `"printer_file"` causes captured files to be written to the printer's memory
- `"usb_file"` causes captured files to be written to a USB storage device

Default

`"printer_file"`

Example

```
! U1 setvar "file.capture_response.destination" "usb_file"
```

Getvar

To retrieve the current setting value:

```
! U1 getvar "file.capture_response.destination"
```

Result

```
"usb_file"
```

file.capture_response.end

This command disables file capture mode.

Do

Use this command in conjunction with `file.capture_response.begin` and `file.capture_response.destination`. The begin and end commands must be issued from the same port.

To disable file capture mode:

```
! U1 do "file.capture_response.end" ""
```

Values
NA

Default
NA

file.cert.expiration

This command retrieves the certificate expiration information.

Getvar

To return the file expiration certificate information:

```
! U1 getvar "file.cert.expiration"
```

Result

A file with the service name, file name and date of expiration for every certificate in use.

Example

In this example, the `getvar` command returns the certificate expiration information for each communication service (SHA2, WLAN, TLS, WebLink, etc).

```
! U1 getvar "file.cert.expiration"
[{"service":"SHA2","file":"SHA2_DEVICE","end_date":"2037-11-22"},
{"service":"WLAN","file":"CERTCLN.NRD","end_date":"2019-06-22"},
{"service":"WIRED","file":null,"end_date":null},
{"service":"WEBLINK1","file":"WEBLINK1_CERT.NRD","end_date":"2096-01-02"},
{"service":"WEBLINK2","file":null,"end_date":null},
{"service":"TLSRAW","file":null," end_date":null},
{"service":"HTTPS","file":"HTTPS_CERT.NRD","end_date":"2020-03-14"}]
```

In the example above, the command returns the service name, file name and date of expiration for every certificate in use. The expiration date is in the YYYY-MM-DD format. The certificates that are not provided by the user are listed as SHA_2 DEVICE as they are available in the Zebra certificate directory. The printer returns the certificate file information even for not enabled services. If a certificate is not in use for a particular service, the command returns a null value.

NOTE:

- This command is not displayed in an ALLCV or JSON allconfig as per the SW request as the JSON SGD is not compatible with the SDK.
- The command only works with certificate files in use by a service. This command does not work with CA, KEY, CSR, or any other files.

file.cert.supported_curves

This command retrieves a list of supported elliptical curves for certificates.

Getvar

To return the list of supported elliptical curves:

```
! U1 getvar "file.cert.supported_curves"
```

Result

A comma delimited list of curve names from OpenSSL that the printer supports for certificates and encryption in general.

file.delete

This command instructs the printer to delete specified files.

Do

To instruct the printer to delete specified files:

```
! U1 do "file.delete" "value"
```

Values

file name

NOTE: Be sure to always specify the memory location.

Example

This do example shows the specified file to delete.

```
! U1 do "file.delete" "e:abcd.zpl"
```

file.dir

This command displays a directory listing on the same port the command was received.

Setvar

To set the directory name from which to retrieve files:

```
! U1 setvar "file.dir" "value"
```

Values

directory letter

NOTE: Be sure to always specify the memory location.

Getvar

To retrieve a directory listing of the specified directory:

```
! U1 getvar "file.dir"
```

NOTE: Be sure to always specify the memory location.

Do

To set the directory name from which to retrieve files:

```
! U1 do "file.dir" "value"
```

Values

directory letter

NOTE: Be sure to always specify the memory location.

Example

This `do` example shows the directory listing of the specified directory.

```
! U1 do "file.dir" "R:"
- DIR R:*.*
-  11172192 bytes free R: RAM
```

file.dir_format

This command controls the output format of the `file.dir` Set/Get/Do command.

Setvar

To set the output format:

```
! U1 setvar "file.dir_format" "value"
```

Values

- `"cpcl"`
- `"zpl"`

Getvar

To retrieves the current setting for the output format:

```
! U1 getvar "file.dir_format"
```

Result

- `"cpcl"`
- `"zpl"`

Do

To set the output format:

```
! U1 do "file.dir_format" "value"
```

Values

- `"cpcl"`
- `"zpl"`

Example

This `do` example sets the directory format to `CPCL`.

```
! U1 do "file.dir_format" "cpcl"
```

file.type

This command displays the contents of the specified file.

Setvar

To instruct the printer to display the content of a file:

```
! U1 setvar "file.type" "value"
```

The contents are displayed on the same port as the command was received.

Values

the drive letter, file name, file extension, such as `R:TEST.ZPL`

NOTE: Be sure to always specify the memory location.

Do

To display the content of a specified file:

```
! U1 do "file.type" "value"
```

Values

The drive letter, file name, file extension, such as `R:TEST.ZPL`

NOTE: Be sure to always specify the memory location.

Example

This `setvar` example shows the value set to `"R:TEST.ZPL"`.

```
! U1 setvar "file.type" "R:TEST.ZPL"
```

When the `setvar` value is set to `"R:TEST.ZPL"`, the contents of the file `TEST.ZPL` located on the `R:` drive will be displayed.

file.run

This command instructs the printer to send a specified file to the parser.

Setvar

To instruct the printer to send a specified file to the parser:

```
! U1 setvar "file.run" "values"
```

Values

drive:filename.extension

NOTE: Be sure to always specify the memory location.

Do

To instruct the printer to send a specified file to the parser:

```
! U1 do "file.run" "value"
```

Values

drive:filename.extension

NOTE: Be sure to always specify the memory location.

Example

This `setvar` example will send the file "text.zpl" stored in RAM to the parser.

```
! U1 setvar "file.run" "R:text.zpl"
```

formats.cancel_all

The ~JA command cancels all format commands in the buffer. It also cancels any batches that are printing. This command is equivalent to the ~JA ZPL command.

Setvar

To cancel all format commands in the buffer:

```
! U1 setvar "formats.cancel_all" ""
```

Values
NA

Default
NA

Do

To cancel all format commands in the buffer:

```
! U1 do "formats.cancel_all" ""
```

head.authenticated

This command reports if the printhead is authenticated.

Getvar

To return the current state of the authenticated printhead:

```
! U1 getvar "head.authenticated"
```

Result

`"yes"` means the printhead has passed printhead authentication

`"no"` means the printhead has failed printhead authentication

Example

In the example below, the getvar returns the current state of the authenticated printhead.

```
! U1 getvar "head.authenticated" "yes"
```

head.darkness_switch_enable

Enables the darkness switch on desktop printers.

Setvar

To enable or disable the darkness switch:

```
! U1 setvar "head.darkness_switch_enable" "value"
```

Values

- `"on"` = enables the darkness switch
- `"off"` = disables the darkness switch

Default

`"on"`

Getvar

To retrieve the current setting value:

```
! U1 getvar "head.darkness_switch_enable"
```

head.darkness_switch

Indicates the value to which the darkness switch is set on the printer.

Getvar

To return which darkness switch is set on the printer:

```
! U1 getvar "head.darkness_switch"
```

Values

Looking at the switch from the rear of the printer:

- `"low"` indicates the darkness switch is on the left
- `"medium"` indicates the darkness switch is in the middle
- `"high"` indicates the darkness switch is on the right

head.element_test

This command will cause the printer to immediately run the head test on all printhead elements. The command can also display the result of the last head element test.

Do

To run the head test on all printhead elements:

```
! U1 do "head.element_test" ""
```

Getvar

To display the result of the last head element test:

```
! U1 getvar "head.element_test"
```

Values

The possible getvar responses include:

- `"Head Elements OK"` All head elements passed the test.
- `"n, n..."` A comma-separated list of elements that failed the test.
- `"Initialization Failed"` The test could not start.
- `"Failed to Attach"` The test could not start.
- `"Please Run Test"` Default response if there are no results to display.
- `"In Progress"` The element test has been started but not completed yet.

Default

`"Please Run Test"`

Example

This example shows a single element that failed the test.

```
"86"
```

This example shows a list of elements that failed the test.

```
"75,309,456,778,779"
```

head.latch

This command identifies if the printhead is open or closed.

Getvar

To retrieve the status of the printhead, open or closed:

```
! U1 getvar "head.latch"
```

Values

`"ok"` is closed

`"open"` is open

Example

In this example, the `getvar` retrieves the status of the print head.

```
! U1 getvar "head.latch"
"ok"
```

head.resolution.in_dpi

Returns the resolution of the print head in dots per inch as an integer.

Getvar

To return the resolution of the print head in dots per inch:

```
! U1 getvar "head.resolution.in_dpi"
```

Values

- "200"
- "300"
- "600"

input.capture

This parameter allows capturing input data in diagnostics mode. Input capture has three modes: , `"print"`, `"usb_host"`,`"run"`, and `"off"`. The `"print"`,`"usb_host"`, and `"run"` modes can be used to examine data received by the printer.

When in `"print"` mode the printer will save incoming data to files named `"in???.dmp",` where ??? is a number between 001 to 999. The printer will then print the text and hexadecimal representation of data bytes received instead of printing the formatted labels which that data might represent.

When in `"usb_host"` mode the printer will save incoming data to files named named `"in???.dmp",` where ??? is a number between 001 to 999. The printer will then save the files to a USB Host memory device if present.

When in `"run"` mode the printer will save captured incoming data to files as above, but will otherwise run the incoming data/commands normally.

The capture files should be deleted from printer memory after retrieving them. Leaving the printer in `"print"` or `"run"` mode and not deleting the capture files will reduce the printer's available flash memory.

The `"off"` mode is the printer's normal operating mode. Cycling power will also return the printer to `"off"` mode.

NOTE: This command does not capture a network packet trace.

Setvar

To set the directory name from which to retrieve files:

```
! U1 setvar "input.capture" "value"
```

Values

- `"print"`
- `"usb_host"`
- `"run"`
- `"off"`

Getvar

To retrieve the current `input.capture` setting value:

```
! U1 getvar "input.capture"
```

NOTE:

The `device.diagnostic_print` setting is deprecated.

Setting `device.diagnostic_print` to `"enabled"` is the equivalent of setting `input_capture` to `"print"`.

```
! U1 setvar "device.diagnostic_print" "enabled"
```

Setting `device.diagnostic_print` to `"disabled"` is the equivalent of setting `input_capture` to `"off"`.

```
! U1 setvar "device.diagnostic_print" "disabled"
```

log.reboot.code

Causes the printer to return a one-character value which indicates the reason for the last printer reboot.

Getvar

To return a one-character value which indicates the reason for the last printer reboot:

```
! U1 getvar "log.reboot.code"
```

Result

A one-character code indicating the reason for the reboot.

Values

- "0" Other
- "1" device.reset command
- "2" Mirror reset – new files
- "3" DTR off
- "4" Low-battery timeout
- "5" Low-battery shutdown
- "6" power.shutdown command
- "7" Idle timeout
- "8" Printer OS update
- "9" Reserved
- "a" Reserved
- "b" Off key
- "f" No data

Example

```
! U1 getvar "log.reboot.code"
"4"
```

The result indicates that the device rebooted because the battery timed out.

log.reboot.codes

Causes the printer to return a list of one-character values which indicates the reasons for the last 32 printer reboots.

Getvar

To return with the reboot codes:

```
! U1 getvar "log.reboot.codes"
```

Result

A string of one-character codes indicating the reason for the reboots. A total of 32 reboot events are stored; if less than 32 reboots have occurred, "f" is stored in any unpopulated event slot, indicating "no data" for that event.

Values

- "0" Other
- "1" device.reset command
- "2" Mirror reset – new files
- "3" DTR off
- "4" Low-battery timeout
- "5" Low-battery shutdown
- "6" power.shutdown command
- "7" Idle timeout
- "8" Printer OS update
- "9" Reserved
- "a" Reserved
- "b" Off key
- "f" No data

Example

```
! U1 getvar "log.reboot.codes"
"bb338bbbbbbbb3bbbbbbbbbbbbbbb1bb"
```

log.reboot.reason

Returns the reason for the last printer reboot, based on the `log.reboot.code`.

Getvar

To return the reason for the last printer reboot, based on the `log.reboot.code`:

```
! U1 getvar "log.reboot.reason"
```

Result

The reason for the last reboot.

Values

```
"Other"
"device.reset command"
"Mirror reset – new files"
"DTR off"
"Low-battery timeout"
"Low-battery shutdown"
"power.shutdown command"
"Idle timeout"
"New OS reprogramming"
"Unknown-1"
"Unknown-2"
"Off key"
"No data"
```

log.reboot.report

Causes the printer to return a list of values which indicate the reasons for the last 32 printer reboots.

Getvar

To return a list of values which indicate the reasons for the last 32 printer reboots:

```
! U1 getvar "log.reboot.report"
```

Result

The recorded reboot reasons as a list, starting with the most recent reboot reason first. A total of 32 reboot events are stored; if less than 32 reboots have occurred, "f" is stored in any unpopulated event slot, indicating "no data" for that event.

Values

```
"Other" "device.reset command" "Mirror reset – new files" "DTR off"
"Low-battery timeout" "Low-battery shutdown" "power.shutdown command"
"Idle timeout" "New OS reprogramming" "Unknown-1" "Unknown-2" "Off key"
"No data"
```

Example

```
! U1 getvar "log.reboot.report"
```

A list of 32 codes, in a carriage-return delimited list:

```
"Off key
Off key
Off key
.
.
.
No data
"
```

mcr.crypt.enabled

Retrieves the MCR encryption-enabled status.

Getvar

To retrieve the MCR encryption-enabled status:

```
! U1 getvar "mcr.crypt.enabled"
```

Values

`"off"` means mcr encryption is not enabled.

`"on"` means mcr encryption is enabled.

mcr.cancel

Cancels the magnetic card read operation currently in progress.

Setvar

To cancel the magnetic card read operation currently in progress:

```
! U1 setvar "mcr.cancel" ""
```

Values

No value needs to be specified.

Default

NA

mcr.crypt.key_mgmt

Retrieves the MCR encryption key management algorithm for a fixed key or DUKPT (Derived Unique Key Per Transaction). The return value applies only if "mcr.crypt.enabled" is "on".

Getvar

To retrieve the MCR encryption key management algorithm for a fixed key or DUKPT:

```
! U1 getvar "mcr.crypt.key_mgmt"
```

Values

"" fixed key algorithm

"dukpt" derived unique key per transaction

Default

NA

mcr.crypt.algorithm

Retrieves the MCR encryption algorithm. The return value applies only if `"mcr.crypt.enabled"` is `"on"`.

Getvar

To retrieve the MCR encryption algorithm:

```
! U1 getvar "mcr.crypt.algorithm"
```

Values

- `""`
- `"3des"`
- `"aes"`

Default
NA

mcr.out

Specifies the communication port which MCR (Mag Card Reader) output is sent to.

Setvar

To specify the communication port which MCR (Mag Card Reader) output is sent to:

```
! U1 setvar "mcr.out" "value"
```

Values

`"active"` means the data is sent out over the same port that the command was received on.

If “multiple” is specified in the option string of the `mcr.enable` command, data will continue to be be sent to the port defined by this command.

`"alert"` means the data will be forwarded as a weblink alert.

Default

`"active"`

mcr.revision

Returns the revision of the MCR (magnetic card reader).

Getvar

To return the revision of the MCR (magnetic card reader):

```
! U1 getvar "mcr.revision"
```

Result

```
"ID TECH TM3 SecureHead RS232 Reader V 5.14"
```

media.bar_location

Allows the user to configure the printer to look for a black mark bar on the front or back of the media.

NOTE: This command works only with printers that have a front media sensor.

Setvar

To configure the printer's black bar location:

```
! U1 setvar "media.bar_location" "value"
```

Values

- `"front"` uses media with bars on the front.
- `"back"` uses media with bars on the back

Options available by printer:

- iMZ220, iMZ320, QLn220, QLn320, QLn420, ZR318, ZR328 use `"front"`
- ZQ310, ZQ320, ZQ510, ZQ520 use`"front", "back"`
- All other printers use `"back"`

Default

- iMZ220, iMZ320, QLn220, QLn320, QLn420, ZR318, ZR328 use `"front"`
- ZQ310, ZQ320 with optional back bar sensor use `"back"`
- ZQ310, ZQ320 with no back bar sensor use `"front"`
- All other printers use`"back"`

Getvar

To return the current setting value:

```
! U1 getvar "media.bar_location"
```

media.cartridge.darkness

This command returns the recommended print darkness setting to be used with the media cartridge currently installed in the printer.

Getvar

To return the recommended print darkness setting for the printer:

```
! U1 getvar "media.cartridge.darkness"
```

Result

"0" to "300"

"0"= no cartridge installed

media.cartridge.labels_remaining

This command returns the number of labels which remain in the cartridge.

Getvar

To return the number of remaining labels:

```
! U1 getvar "media.cartridge.labels_remaining"
```

Result

An integer >= 0.

`"no"` indicates that the cartridge is not inserted or the printer does not support this command.

Example

In this example, the `getvar` returns the number of print labels that is remaining in the cartridge.

```
! U1 getvar "media.cartridge.labels_remaining" "10"
```

media.cartridge.width

This command returns the width of the media cartridge installed in the printer.

Getvar

To fetch the width of the media cartridge:

```
! U1 getvar "media.cartridge.width"
```

Result

A numeric value specified in dots.

`"0"` indicates that the cartridge is not installed

Example

In the example below, the getvar returns with the width of the media cartridge in dots.

```
! U1 getvar "media.cartridge.width" "300"
```

media.cartridge.total_label_cnt

This command returns the total number of labels that is initially available in the cartridge.

Getvar

To return the total number of labels initially available in the cartridge:

```
! U1 getvar "media.cartridge.total_label_cnt"
```

Result

The value depends on the length of the label and other factors. It typically ranges from 100-300 labels.

`"0"` inidciates that the cartridge is not installed

Example

In this example, the getvar returns with the total label count available in the cartridge.

```
! U1 getvar "media.cartridge.total_label_cnt" "100"
```

media.cartridge.speed

This command fetches the print speed for the media cartridge.

Getvar

To get the print speed for the cartridge:

```
! U1 getvar "media.cartridge.speed"
```

Result

`"0"` indicates that the cartridge is not installed

`"2"`

`"4"`

Currently, the only cartridge speeds supported are `"2"` and `"4"`.

media.cartridge.length

This command returns the initial label length of the media cartridge installed in the printer.

Getvar

To fetch the length of the media cartridge:

```
! U1 getvar "media.cartridge.length "
```

Result

A numeric value specified in dots.

`"0"` indicates that no cartridge is installed

Example

In the example below, the `getvar` returns with the width of the media cartridge in dots.

```
! U1 getvar "media.cartridge.length" "300"
```

media.cartridge.inserted

This command verifies if the media cartridge is inserted or not.

Getvar

To view if the media cartridge is inserted or not

```
! U1 getvar "media.cartridge.inserted"
```

Result

"yes"	cartridge is inserted
"no"	cartridge is not inserted

Example

In this example, the getvar returns with the information that the media cartridge is inserted.

```
! U1 getvar "media.cartridge.inserted" "yes"
```

media.cartridge.part_number

This command returns the part number of the media cartridge installed in the printer.

Getvar

To get the part number of the media cartridge:

```
! U1 getvar "media.cartridge.part_number"
```

Result

0 to 16 character string

"" indicates that no cartridge installed

Example

In this example, the getvar returns with the part number of the media cartridge.

```
! U1 getvar "media.cartridge.part_number" "100127132K"
```

media.cut_now

This command instructs the printer cycle the media cutter. If the printer is in Print Mode Kiosk (media.printmode “K”) then the cutter will execute a cut based on the value of media.present.cut_amount – either a normal cut or a partial cut. If the printer is not in Print Mode Kiosk (media.printmode “K”), this command does nothing. .

See media.present.cut_amount on page 864.

Setvar

To instruct the printer to cycle the media cutter:

```
! U1 setvar "media.cut_now" ""
```

NOTE: See media.present.cut_amount on page 864.

Do

To instruct the printer to cycle the media cutter:

```
! U1 do "media.cut_now" ""
```

media.darkness_mode

This command instructs the printer to set the darkness mode.

Setvar

To set the darkness mode:

```
! U1 setvar "media.darkness_mode" "value"
```

Values

- `"cartridge"` indicates cartridge mode (no changes allowed
- `"user"` indicates user mode (Darkness is set by the user, and the cartridge value is ignored. This value is used for all cartridges inserted in the printer).
- `"relative"` indicates relative mode (the specified darkness value is added to the cartridge default value)

Default

`"cartridge"`

Example

This `setvar` example shows the darkness mode set to `"cartridge"`.

```
! U1 setvar "media.darkness_mode" "cartridge"
```

media.draft_mode

This command puts the printer into draft mode setting.

NOTE: Setting the printer to draft mode may result in poorer print quality depending on print speed, label configurations, etc.

Setvar

To set the value:

```
! U1 setvar "media.draft_mode" "value"
```

Values

Accepted values are different for Link-OS and legacy printers.

Link-OS printers accept:

- `"on"` indicates faster ramp (acceleration) speed
- `"off"` indicates normal ramp (acceleration) speed

Legacy printers accept:

- `"enabled"` indicates faster ramp (acceleration) speed
- `"disabled"` indicates normal ramp (acceleration) speed

Default

Link-OS printers: `"off"`

Legacy printers: `"disabled"`

Getvar

To return the currently set value:

```
! U1 getvar "media.draft_mode"
```

media.dynamic_length_calibration

This command enables or disables the dynamic length calibration. This is identical to the first parameter of the ^XS command - Dynamic Length Calibration.

Setvar

To enable or disable the dynamic length calibration:

```
U1 setvar "media.dynamic_length_calibration" "value"
```

Values

"on" indicates dynamic length calibration is enabled.

"off" dynamic length calibration is disabled.

Default Value

- "on" for Desktop printers
- "off" for Industrial printers

Getvar

To return the current setting:

```
! U1 getvar "media.dynamic_length_calibration"
```

Example

In this example, the getvar returns the current setting of the dynamic length calibration.

```
! U1 setvar "media.dynamic_length_calibration" "on"
```

media.extended_presentation

This command increases the peel presentation position to match the extended peel plate option on some printer models.

Setvar

NOTE: This command is supported only by the ZT411 and ZT421 printers.

When this parameter is enabled, the peel-bar-to-printline, tear-off-to-printline, and applicator-to-printline distances are increased.

When this parameter is disabled, the distances return to their initial values.

To set the extended media presentation:

```
! U1 setvar "media.extended_presentation" "value"
```

Values

`"enabled"` enables extending the peel presentation position

`"disabled"` disables extending the peel presentation position

Default

`"disabled"`

The initial value is `"disabled"`; however, this setting will not reset to a particular default.

Getvar

To retrieve the extended media presentation:

```
! U1 getvar "media.extended_presentation"
```

Example

This `setvar` example enables the command by setting the value to `"enabled"`.

```
! U1 setvar "media.extended_presentation" "enabled"
```

media.feed_skip

This command only applies to labels created with CPCL commands. It controls the same setting as the second parameter of the CPCL SETFF command.

Setvar

To set the printer's feed skip length:

```
! U1 setvar "media.feed_skip" "value"
```

Values

A numeric value from "0" to "50".

Default

"5"

Getvar

To return the current setting value:

```
! U1 getvar "media.feed_skip"
```

Result

A numeric value from "0" to "50".

media.linerless_offset

This command sets or retrieves the value of the linerless media offset (or the size of the so-called "no print zone" or "dead zone"). This offset, which is at the beginning of the label, starts at the top-of-form and continues in the number of pixels defined by the last user-specified value.

Setvar

To set the printer to adjust the linerless media offset:

```
! U1 setvar "media.linerless_offset" "value"
```

Values and Defaults

For the ZT411 linerless printers, the no print zone or leader length data range and default value are DPI dependent:

DPI	Min	Default	Max
203	0	61	76
300	0	90	113
600	0	180	225

For other printers with a legacy cutter installed, the default value is `"0"`, and the range is `"0"` to `"225"`.

Getvar

To retrieve the existing value set for linerless media offset:

```
! U1 getvar "media.linerless_offset"
```

Example (for a 203 dpi linerless printer)

This `setvar` example shows the value set to `"65"`.

```
! U1 setvar "media.linerless_offset" "65"
```

When the `setvar` value is set to `"65"`, the `getvar` result is `"65"`.

media.media_low.external

This printer setting gets the status of the external `media.media_low` warning.

Getvar

To instruct the printer to respond with the currently set media print mode:

```
! U1 getvar "media.media_low.external"
```

Values

- `"0"` indicates paper present at sensor position
- `"1"` indicates no paper present

NOTE: The status of the sensor is sampled every time the printout is cut. If three succeeding samples show "no paper", the status reply changes to 1. This is to prevent a false alarm if the side of the paper roll is not clean. If the current status of the sensor is required, use ~HQES and extract the paper near-end sensor bit.

media.media_low.warning

This command retrievs the value of, or enables or disables the Supplies Warning system.

Setvar

To enable or disable the Supplies Warning System:

```
! U1 setvar "media.media_low.warning" "value"
```

Values

`"disabled"` indicates not active

`"enabled"` indicates active

Default

`"disabled"`

Getvar

To retrieve the setting for the Supplies Warning system:

```
! U1 getvar "media.media_low.warning"
```

Example

This `setvar` example disables the Supplies Warning system.

```
! U1 setvar "media.media_low.warning" "disabled"
```

media.part_number

Sets the media's part number.

Setvar

To set the media's part number:

```
! U1 setvar "media.part_number" "value"
```

Values

An alpha-numeric string between 0 and 64 characters

Default

NA

Example

```
! U1 setvar "media.part_number" "123AB987"
```

Getvar

To return the current setting value:

```
! U1 getvar "media.part_number"
```

Result

```
"123AB987"
```

media.present.cut_amount

This printer setting determines the type of cut made by the printer cutter (normal or partial) and, if partial, the length of the partial cut on each side, in mm.

Setvar

To instruct the printer to change the media cut amount:

```
! U1 setvar "media.present.cut_amount" "value"
```

Values

- `"0"` indicates a normal cut
- `"10"` to `"60"` indicates a partial cut where the value indicates the number of mm of media left uncut

Getvar

To respond with the currently set media cut amount:

```
! U1 getvar "media.present.cut_amount"
```

media.present.eject

This command instructs the printer to eject the document through the presenter module. The value is the amount to eject, in mm. The value of `media.present.length_addition` gets added to the value to determine the total length of media ejected.

Setvar

This command instructs the printer to eject the document through the presenter module.

```
! U1 setvar "media.present.eject" "value"
```

Values

`"0"` to `"255"` specifies the amount of media to eject in mm

NOTE: See media.present.length_addition on page 866.

Do

This command instructs the printer to eject the document through the presenter module.

```
! U1 do "media.present.eject" "value"
```

media.present.length_addition

This printer setting adds an additional amount to how far the paper is ejected during a present cycle. A standard amount of 50mm is always added to clear the kiosk wall. This amount is added to that 50mm. The total amount of media ejected this command is executed, then, is 50mm + media.present.length_addition + media.present.eject.

Setvar

This command instructs the printer to change the media present length addition.

```
! U1 setvar "media.present.length_addition" "value"
```

Values

"0" to "255" specifies the additional mm of media to eject

Getvar

This command instructs the printer to respond with the currently set media present length addition.

```
! U1 getvar "media.present.length_addition"
```

media.present.loop_length

This printer setting determines the length of the presenter loop. If loop_length is greater than loop_length_max (see `media.present.loop_length_max`) then it will be set equal to loop_length_max.

Setvar

This command instructs the printer to change the presenter loop length.

```
! U1 setvar "media.present.loop_length" "value"
```

Values

- `"0"` means paper is fed straight through the presenter
- 3-1023 specifies a loop length in mm

Default

`"400"`

Getvar

This command instructs the printer to respond with the currently set presenter loop length.

```
! U1 getvar "media.present.loop_length"
```

media.present.loop_length_max

This printer setting determines the maximum allowed length of the presenter loop.

Setvar

This command instructs the printer to change the presenter loop length.

```
! U1 setvar "media.present.loop_length_max" "value"
```

Values

- `"0"` feeds paper straight through to the presenter
- `"3"` to `"1023"` specifies the loop length in mm

Default

`"400"`

Getvar

This command instructs the printer to respond with the currently set presenter loop length.

```
! U1 getvar "media.present.loop_length_max"
```

media.present.cut_margin

This printer setting determines the margin between the cutter and the printhead.

Setvar

This command instructs the printer to change the media cut amount.

```
! U1 setvar "media.present.cut_margin" "value"
```

Values
"2" to "9" mm of distance

Default
"9" mm of distance

Getvar

This command instructs the printer to respond with the currently set media cut margin amount.

```
! U1 getvar "media.present.cut_margin"
```

media.present.present_timeout

This printer setting determines how long the printer will wait after a present event to clear the label. See ^KV ZPL command.

Setvar

This command instructs the printer to change the presenter function mode.

```
! U1 setvar "media.present.present_timeout" "value"
```

Values

"0" to "300"

If label is not taken, retract label when timeout expires. Timeout is in seconds. Zero (0) indicates that there is no timeout. The label will stay presented until removed manually or a new label is printed.

Getvar

This command instructs the printer to respond with the currently set presenter function mode.

```
! U1 getvar "media.present.present_timeout"
```

media.present.present_type

This printer setting determines the way that the printer performs a present command. See ^KV ZPL command.

Setvar

This command instructs the printer to change the presenter function mode.

```
! U1 setvar "media.present.present_type" "value"
```

Values

- "0" Ejects page when new page is printed
- "1" Retracts page when new page is printed
- "2" Does nothing when new page is printed

Getvar

This command instructs the printer to respond with the currently set presenter function mode.

```
! U1 getvar "media.present.present_type"
```

media.printmode

This printer setting determines the action the printer takes after a label or group of labels has printed.

Setvar

This command is equivalent to ^MM on page 305 .

This command instructs the printer to change the media print mode.

```
! U1 setvar "media.printmode" "value"
```

Values

- "T" Tear-off[1]
- "P" Peel-off (not available on S-300)[1, 2]
- "R" Rewind (depends on printer model)
- "A" Applicator (depends on printer model)[1]
- "C" (depends on printer model) Cutter[2]
- "D" Delayed cutter[1, 2]
- "F" RFID[1, 2]
- "L" Linerless Peel[2, 3]
- "U" Linerless Rewind[2, 3]
- "K" Kiosk[4]
- "V" Linerless Tear[2]
- "S" Stream[5]

1. Not supported on the KR403 printer.
2. Not supported on the ZE500 printer.
3. Not supported on the ZM400/ZM600 and RZ400/RZ600 printers.
4. Only supported on the KR403 printer.
5. Only supported on the ZE500 printer.

Getvar

This command instructs the printer to respond with the the currently set media print mode.

```
! U1 getvar "media.printmode"
```

Example

This `setvar` example shows the value set to "T".

```
! U1 setvar "media.printmode" "T"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"tear off"`.

Setvar / Getvar Relation

When setvar is	getvar response and control panel display
"T"	TEAR OFF
"P"	PEEL OFF
"R"	REWIND
"A"	APPLICATOR
"C"	CUTTER
"D"	DELAYED CUT
"L"	RESERVED
"U"	RESERVED
"K"	KIOSK

media.speed

This command specifies media print speed in inches per second (ips).

Setvar

This command instructs the printer to set the media print speed.

```
! U1 setvar "media.speed" "value"
```

Values

"2" to "12" ips

- "up" increments the printer speed by one unit
- "down" = decrements the speed by one unit

Default

"2"

Getvar

This command retrieves the currently set media print speed.

```
! U1 getvar "media.speed"
```

Example

This setvar example shows the value set to "2".

```
! U1 setvar "media.speed" "2"
```

When the setvar value is set to "2", the getvar result is "2".

This setvar example shows the value set to "up".

```
! U1 setvar "media.speed" "up"
```

If the current print speed is 2 and the setvar value is set to "up", the getvar result is "3".

This setvar example shows the value set to "down".

```
! U1 setvar "media.speed" "down"
```

If the current print speed is 2 and the setvar value is set to "down", the getvar result is "1".

media.serial_number

Sets the media's serial number.

Setvar

```
! U1 setvar "media.serial_number" "value"
```

Values
An alphanumeric string between 0 and 64 characters.

Default
NA

Example

```
! U1 setvar "media.serial_number" "A34567BC6789"
```

Getvar

```
! U1 getvar "media.serial_number"
```

The result is "A34567BC6789".

media.tof

Sets the offset of the black mark or label gap to the point of separation between documents.

Related ZPL Command

^MN

Setvar

To set the black mark offset value:

```
! U1 setvar "media.tof" "value"
```

Values

- 0 is the point of separation, such as the perforation or cut point
- -84 to 64 for the MZ printer
- -400 to 400 for all other Mobile printers
- -400 is the lower limit for all other printers

For all other printers, the upper limit depends on the print mode (Thermal Transfer or Direct Thermal), the printer model, the print head density, and the current value of Label Top.

Default
0

Example

```
! U1 setvar "media.tof" "-50"
```

Getvar

To return the current setting value:

```
! U1 getvar "media.tof"
```

Result
"-50"

IMPORTANT: The coordinate system for media.tof is reversed from ZPL; therefore media.tof is the negative of the b value of ^MNM,b. If ^MNM,b sets the offset value to "50", then the media.tofgetvar value returned will be "-50".

Example

```
^XA^MNM,50^XZ
! U1 getvar "media.tof"
```

Result
"-50"

Supported Devices

- iMZ220, iMZ320
- QLn220, QLn320, QLn420
- ZD220, ZD230
- ZD410, ZD420
- ZD500, ZD510
- ZD511
- ZD620
- ZD888
- ZQ120, ZQ220
- ZQ310, ZQ320
- ZQ510, ZQ520
- ZQ610, ZQ620

memory.flash_free

This parameter returns the amount of available Flash memory.

Getvar

```
! U1 getvar "memory.flash_free"
```

memory.flash_size

This parameter returns the total amount of Flash memory.

Getvar

To return the current setting value:

```
! U1 getvar "memory.flash_size"
```

memory.ram_free

This parameter returns the amount availabale Random Access Memory (RAM).

Getvar

To return the current setting value:

```
! U1 getvar "memory.ram_free"
```

memory.ram_size

This parameter returns the total amount of Random Access Memory (RAM).

Getvar

To return the RAM size:

```
! U1 getvar "memory.ram_size"
```

mqtt.enable

This command controls MQTT functionality. When disabled, open MQTT connections are closed. When enabled, a connection is established if all the configuration requirements are met.

Setvar

To enable or disable MQTT messaging:

```
! U1 setvar "mqtt.enable" "value"
```

Values

- `"on"` enables MQTT messaging.
- `"off"` disables MQTT messaging.

Default

`"off"`

Getvar

To view whether MQTT messaging is enabled:

```
! U1 getvar "mqtt.enable"
```

Values

- `"on"` indicates MQTT messaging is enabled.
- `"off"` indicates MQTT messaging is disabled.

mqtt.logging.clear

This command clears the MQTT log, it does not affect syslog.

Do

To clear the MQTT log:

```
! U1 do "mqtt.logging.clear"
```

mqtt.logging.entries

This command returns an MQTT list of log entries for both mqtt1 (conn1) and mqtt2 (conn2) that are created on printer power up and cleared after a power cycle or an explicit clearing. The number of entries is determined by `mqtt.logging.max_entries`.

Getvar

To return a list of MQTT log entries for connection 1 and connection 2 from the printer:

```
! U1 getvar "mqtt.logging.entries"
```

Value
An ASCII string of log data.

Example

This example shows an example of a log entry.

```
"[07-12-2021 21:31:29.733][Info][0000100F][mqtt1] Waiting 2 seconds "
```

mqtt.logging.max_entries

This command sets the maximum number of log entries recorded before the least-recently-logged entry is removed. A value of 0 disables recording MQTT logs to the `mqtt.logging.entries` SGD; log entries are still recorded to syslog. Changing this number to a lower value than the number of log entries currently recorded causes older entries to be removed until the total number of entries does not exceed the maximum number of entries.

Setvar

To set the maximum number of MQTT log entries:

```
! U1 setvar "mqtt.logging.max_entries" "value"
```

Value
A numeric value between `"1"` and `"10,000"`. A value of zero (`"0"`) disables logging to `mqtt.logging.entries`.

Default
`""`

Getvar

To return the setting for `mqtt.logging.max_entries`:

```
! U1 getvar "mqtt.logging.max_entries"
```

This command returns the current value for the maximum number of log entries listed in `mqtt.logging.entries`.

mqtt.restore_defaults

This command tells the printer to return to the default MQTT settings. No other settings are affected by this command.

Setvar

To restore the default MQTT settings on the printer:

```
! U1 setvar "mqtt.restore_defaults"
```

No values are required for this command.

mqtt.conn[1|2].clean_session_flag

This command specifies whether to clear all previous values on MQTT connection 1 or MQTT connection 2. When set to `"on"`, the default, the broker does not create a persistent session for new connections. If the client disconnects, any messages in the queue are lost. When set to `"off"`, the broker creates a persistent session and any messages in the queue persist if the client disconnects. The persisted messages are sent when the client reconnects.

Setvar

To persist MQTT messages on MQTT connections:

```
! U1 setvar "mqtt.conn1.clean_session_flag" "value"
```

```
! U1 setvar "mqtt.conn2.clean_session_flag" "value"
```

Values

- `"on"` (default) the broker does not persist MQTT message on connection 1.
- `"off"` causes the broker to persist messages

Default

`"on"`

Getvar

To have the printer return the value of the MQTT session flag for connection 1:

```
! U1 getvar "mqtt.conn1.clean_session_flag"
```

```
! U1 getvar "mqtt.conn2.clean_session_flag"
```

Values

- `"on"` (default) the broker does not persist MQTT message on connection 1.
- `"off"` causes the broker to persist messages

mqtt.conn[1|2].password

This command specifies the password to use for MQTT connection 1 or 2. If set to "", a password is not sent to the broker. MQTT 3.1.1 brokers require either username only, username and password, or neither. If only a password is set, a warning is issued in the syslog.

Setvar

To set the password for MQTT connection 1 or 2:

```
! U1 setvar "mqtt.conn1.password" "value"
```

```
! U1 setvar "mqtt.conn2.password" "value"
```

Value
A string with a maximum of 64 characters. If the password is empty, it is not sent to the broker.

Default
""

Getvar

To have the printer return the password for connection 1 or 2:

```
! U1 getvar "mqtt.conn1.password"
```

```
! U1 getvar "mqtt.conn2.password"
```

Result
The printer returns * and does not return the password

mqtt.conn[1|2].ping_interval

This command specifies how often connection 1 or 2 sends a ping to the MQTT broker to keep the connection open. This value is in seconds. If the MQTT broker does not receive a ping request within this interval, the broker will disconnect from the client.

Setvar

To set the ping interval for connection 1 or 2:

```
! U1 setvar "mqtt.conn1.ping_interval" "value"
```

```
! U1 setvar "mqtt.conn2.ping_interval" "value"
```

Value
A numeric value between `"1"` and `"300"` seconds.

Default
`"30"` seconds

Getvar

To have the printer return the current ping interval for connection 1 or 2:

```
! U1 getvar "mqtt.conn1.ping_interval"
```

```
! U1 getvar "mqtt.conn2.ping_interval"
```

The printer returns a numeric value between `"1"` and `"300"` seconds.

mqtt.conn[1|2].reset_now

This command tells the printer to reset MQTT connection 1 or 2 to apply the recently changed settings.

Setvar

To reset connection 1 or 2:

```
! U1 setvar "mqtt.conn1.reset_now" ""
```

```
! U1 setvar "mqtt.conn2.reset_now" ""
```

Values

This command does not require a value to reset the connection.

mqtt.conn[1|2].reset_required

This command shows if a reset is required in order for an MQTT setting to take effect. Any time a setting for MQTT is changed, a connection reset is required in order for the change to take effect.

Getvar

To have the printer return whether a reset of connection 1 or 2 is required:

```
! U1 getvar "mqtt.conn1.reset_required"
```

```
! U1 getvar "mqtt.conn2.reset_required"
```

Values

- `"no"` indicates a connection reset is not required.
- `"yes"` indicates a connection reset is required.

mqtt.conn[1|2].retry_interval_random_max

This command specifies the maximum random retry interval to attempt a connection to an MQTT broker when the connection is lost. This command prevents a fleet of printers from attempting to connect to a single broker after a network interruption that might cause denial of service (DOS) concerns. The interval is measured in seconds.

Setvar

To set the maximum random retry interval for connection 1 or 2:

```
! U1 setvar "mqtt.conn1.retry_intreval_random_max" "value"
```

```
! U1 setvar "mqtt.conn2.retry_intreval_random_max" "value"
```

Value
A numeric value between `"1"` and `"600"` seconds.

Default
`"120"`

Getvar

To have the printer return the current maximum random retry interval for connection 1 or 2:

```
! U1 getvar "mqtt.conn1.retry_intreval_random_max"
```

```
! U1 getvar "mqtt.conn2.retry_intreval_random_max"
```

The printer returns a numeric value between `"1"` and `"600"` seconds.

mqtt.conn[1|2].server_address

This command specifies the URI the printer client uses to connect to the MQTT broker for connection 1 or 2. The printer will not attempt to connect to the same server address on multiple connections or if the value is invalid. This address must be a TLS MQTT broker (mqtts) as non-TLS broker connections are not supported.

Setvar

To specify the URI for connection 1 or 2 to the MQTT broker:

```
U1 setvar "mqtt.conn1.server_address" "mqtts://<domain>[:port][/path]"
```

```
U1 setvar "mqtt.conn2.server_address" "mqtts://<domain>[:port][/path]"
```

Values

- `<domain>` is a required value. DNS or IP address is an accepted value.
- `[:port]` is an optional value. If the port is not specified, the default port of 8883 is used. A value of zero (0) is not valid.
- `[/path]` is an optional value.

The maximum length of the string is 2048 characters.

Getvar

To have the printer return the URI value for connection 1 or 2:

```
! U1 getvar "mqtt.conn1.server_address"
```

```
! U1 getvar "mqtt.conn2.server_address"
```

The command retrieves the URI.

Example

This example includes a DNS domain of `broker.zebra.com` using port 65412.

```
U1 setvar "mqtt.conn1.server_address" "mqtts://broker.zebra.com:65412"
```

mqtt.conn[1|2].tenant_id

This command specifies the top level of the MQTT topic the printer will subscribe to and publish to on connection 1 or 2.

Setvar

To set the top-level topic that the printer subscribes and publishes to on connection 1 or 2:

```
! U1 setvar "mqtt.conn1.tenant_id" "value"
```

```
! U1 setvar "mqtt.conn2.tenant_id" "value"
```

Values
1 to 64 non-whitespace ASCII characters and must not include a +, #, /, or $.

Default
`"zebra"`

Getvar

To have the printer return the current setting value for connection 1 or 2:

```
! U1 getvar "mqtt.conn1.tenant_id"
```

```
! U1 getvar "mqtt.conn2.tenant_id"
```

Result
The printer returns the tenant ID.

mqtt.conn[1|2].username

This command specifies the username to use for MQTT connection 1 or 2. If the username is set to "", the default, the username is not sent to the broker. MQTT 3.1.1 brokers require either username only, username and password, or neither. If only a password is sent, a warning is issued in syslog.

Setvar

To set the username for connection 1 or 2:

```
! U1 setvar "mqtt.conn1.username" "value"
```

```
! U1 setvar "mqtt.conn2.username" "value"
```

Value
String with a maximum of 64 characters.

Default
""

Getvar

To have the printer return the username for connection 1 or 2:

```
! U1 getvar "mqtt.conn1.username"
```

```
! U1 getvar "mqtt.conn2.username"
```

netmanage.avalanche.agent_addr

This parameter obtains or changes the Network Management agent IP address.

Setvar

To set the Network Management Agent IP address:

```
! U1 setvar "netmanage.avalanche.agent_addr" "value"
```

Values

Any valid IP address.

Default

`"0.0.0.0"`

Getvar

To retrieve the current Network Management IP address:

```
! U1 getvar "netmanage.avalanche.agent_addr"
```

Example

```
! U1 setvar "netmanage.avalanche.agent_addr" "10.14.2.200"
```

netmanage.avalanche.available_agent

This command returns the current IP address of the remote agent found during the Agent Discovery Phase.

Getvar

To obtain the IP address of the remote agent found during the Agent Discovery Phase:

```
! U1 getvar "netmanage.avalanche.available_agent"
```

Result

An IP address

Example

```
! U1 getvar "netmanage.avalanche.available_agent"
"10.3.4.128"
```

netmanage.avalanche.available_port

This command returns the available port of the remote agent found during the Agent Discovery Phase.

Setvar

To set the available port of the remote agent found during the Agent Discovery Phase:

```
! U1 setvar "netmanage.avalanche.available_port" "value"
```

Values

"0" to to "65535"

Default

"0"

Getvar

To retrieve the current port setting of the remote agent found during the Agent Discovery Phase:

```
! U1 getvar "netmanage.avalanche.available_port"
```

Example

```
! U1 setvar "netmanage.avalanche.available_port" "1800"
```

netmanage.avalanche.encryption_type

This parameter sets and gets the Network Management Encryption type to be used.

Setvar

To set the Network Management Encryption type to be used:

```
! U1 setvar "netmanage.avalanche.encryption_type" "value"
```

Values

- "0" None
- "1" Limburger
- "2" AES 128S

Default

"0"

Getvar

To retrieve the currently set Network Management Encryption type:

```
! U1 getvar "netmanage.avalanche.encryption_type"
```

Example

This example sets the value to Limburger (1) encryption type.

```
! U1 getvar "netmanage.avalanche.encryption_type" "1"
```

netmanage.avalanche.interval

This parameter obtains or sets the Network Management Update Interval time stored in the printer. Time is measured in milliseconds (e.g., a setting of "2000" equals 2 seconds).

Setvar

To set the Network Management Update Interval:

```
! U1 setvar "netmanage.avalanche.interval" "value"
```

Values

Any integer value from `"0"` to `"4294967295"` (4,294,967,295 milliseconds)

Default

`"0"`

Getvar

To retrieve the current Network Management Update Interval:

```
! U1 getvar "netmanage.avalanche.interval"
```

Example

This example sets the interval value to 3 seconds.

```
! U1 setvar "netmanage.avalanche.interval" "3000"
```

netmanage.avalanche.interval_update

This parameter turns on or off the Network Management Update Interval.

This command is related to netmanage.avalanche.interval on page 900.

Setvar

To turn on or off the network management interval update:

```
! U1 setvar "netmanage.avalanche.interval_update" "value"
```

Values

"off"

"on"

Default

"off"

Getvar

To retrieve the current network management interval update setting:

```
! U1 getvar "netmanage.avalanche.interval_update"
```

Example

This example sets the device's Network Management Interval Update setting to "on".

```
! U1 setvar "netmanage.avalanche.interval_update" "on"
```

netmanage.avalanche.model_name

This command obtains or sets the current Network Management Device Model Name stored in the printer.

Setvar

To set the current Network Management Device model name:

```
! U1 setvar "netmanage.avalanche.model_name" "value"
```

Values

A string up to 31 characters in length.

Default

NA

Getvar

To retrieve the current Network Management Device model name:

```
! U1 getvar "netmanage.avalanche.model_name"
```

Example

```
! U1 setvar "netmanage.avalanche.model_name" "ZT230"
```

netmanage.avalanche.set_property

This parameter sets Network Management Device Side Property (custom).

Setvar

To set the Network Management Device Side Property (custom):

```
! U1 setvar "netmanage.avalanche.set_property" "value"
```

Values

A string in the format of "AAAA=XXXXXXXX"

Getvar

To retrieve the current Network Management Device Side Property value:

```
! U1 getvar "netmanage.avalanche.set_property"
```

Example

This example will be viewed as a property under the general properety tree in avalanche console.

```
! U1 setvar netmanage.avalanche.set_property" "ZebraLocation=VH"
```

This example will be viewed as a property under the Zebra tree in avalanche console.

```
! U1 setvar netmanage.avalanche.set_property" "Zebra.Location=VH"
```

netmanage.avalanche.startup_update

This parameter sets and retrieves the Network Management Start Up Update setting.

Setvar

To set the device's Network Management Start Up Update setting:

```
! U1 setvar "netmanage.avalanche.startup_update" "value"
```

Values

"off"

"on"

Default

"off"

Getvar

To retrieve the device's current Network Management Start Up Update setting:

```
! U1 getvar "netmanage.avalanche.startup_update"
```

Example

```
    ! U1 setvar "netmanage.avalanche.startup_update" "on"
```

netmanage.avalanche.tcp_connection_timeout

This command sets the Network Management Timeout used for establishing a TCP connection to an Agent. Time is set in milliseconds.

Setvar

To set the Network Management Timeout used for establishing a TCP connection to an Agent:

```
! U1 setvar "netmanage.avalanche.tcp_connection_timeout" "value"
```

Values

Any integer value from `"0"` to `"4294967295"`

Default

`"0"`

Getvar

To retrieve the current Network Management Timeout used for establishing a TCP connection to an Agent:

```
! U1 getvar "netmanage.avalanche.tcp_connection_timeout"
```

Example

This examples sets the connection timeout to 2000 milliseconds (2 seconds).

```
! U1 setvar "netmanage.avalanche.tcp_connection_timeout" "2000"
```

netmanage.avalanche.terminal_id

Sets or retrieves the Terminal ID of the Avalanche server. This value is typically assigned by the Avalanche server.

Setvar

To set the Terminal ID of the Avalanche server:

```
! U1 setvar "netmanage.avalanche.terminal_id" "value"
```

Values

"0" to "402653183"

Default

"0"

Getvar

To return the current setting value:

```
! U1 getvar "netmanage.avalanche.terminal_id"
```

netmanage.avalanche.text_msg.beep

This parameter sets and gets Network Management Text Message Beep enable setting.

Setvar

To set the Network Management Text Message Beep enable setting:

```
! U1 setvar "netmanage.avalanche.text_msg.beep" "value"
```

Values

"off"

"on"

Default

"off"

Getvar

To retrieve the Network Management Text Message Beep enable setting:

```
! U1 getvar "netmanage.avalanche.text_msg.beep"
```

Example

```
! U1 setvar "netmanage.avalanche.text_msg.beep" "on"
```

netmanage.avalanche.text_msg.display

This command turns on and off the Network Management Text Message Display setting.

Setvar

To view the setting of the Network Management Text Message Display of the device:

```
! U1 setvar "netmanage.avalanche.text_msg.display" "value"
```

Values

"off"

"on"

Default

"off"

Getvar

To return the current Network Management Text Message Display enable setting:

```
! U1 getvar "netmanage.avalanche.text_msg.display"
```

Example

```
! U1 setvar "netmanage.avalanche.text_msg.display" "on"
```

netmanage.avalanche.text_msg.print

This command turns on and off the Network Management Text Message Print setting.

Setvar

To set the device's Network Management Text Message Print enable setting:

```
! U1 setvar "netmanage.avalanche.text_msg.print" "value"
```

Values

"off"

"on"

Default

"off"

Getvar

To return the current Network Management Text Message Print enable setting:

```
! U1 getvar "netmanage.avalanche.text_msg.print"
```

Example

```
! U1 setvar "netmanage.avalanche.text_msg.print" "on"
```

netmanage.avalanche.udp_timeout

This command sets the device's Network Management UDP timeout. Time is set in milliseconds.

Setvar

To set the device's Network Management UDP timeout:

```
! U1 setvar "netmanage.avalanche.udp_timeout" "value"
```

Values

Any integer value from `"0"` to `"4294967295"` (4,294,967,295 milliseconds)

Default

`"0"`

Getvar

To return the current Network Management UDP timeout setting:

```
! U1 getvar "netmanage.avalanche.udp_timeout"
```

Example

This example sets the timeout value to .4 seconds (400 milliseconds).

```
! U1 setvar "netmanage.avalanche.udp_timeout" "400"
```

netmanage.error_code

This parameter refers to Avalanche client error code.

Getvar

To return the current setting value:

```
! U1 getvar "netmanage.error_code"
```

Result

```
"0"
```

netmanage.state_code

This parameter refers to Avalanche client state code.

Getvar

To return the Avalanche client state code:

```
! U1 getvar "netmanage.state_code"
```

Result

```
"0"
```

netmanage.status_code

This parameter refers to Avalanche client status code.

Getvar

To return the Avalanche client status code:

```
! U1 getvar "netmanage.status_code"
```

Result

```
"0"
```

odometer.cut_marker_count

Returns the number of cuts incurred by the cutter or resets the counter to "0". This command tracks the same events as `odometer.total_cuts`, which cannot be reset.

Setvar

To set the number of cuts incurred by the cutter:

```
! U1 setvar "odometer.cut_marker_count" "value"
```

Values

`"0"` resets the counter to 0

Getvar

To return the current setting value:

```
! U1 getvar "odometer.cut_marker_count"
```

Result

An integer value of `"0"` or greater.

odometer.headclean

This printer setting refers to the head clean odometer count. This counter tracks how many inches and centimeters have passed through the printer since the head was last cleaned.

Setvar

To reset the head clean counter:

```
! U1 setvar "odometer.headclean" "value"
```

Values

`"0"` resets the head clean counter

Default

Must be an accepted value or it is ignored

Getvar

To retrieve the values for the head clean counter:

```
! U1 getvar "odometer.headclean"
```

Example

This example shows how to get the odometer head clean, how to reset it, and how to confirm the settings changed.

To see the current settings, type:

```
! U1 getvar "odometer.headclean"
```

Something similar to this is shown:

```
"1489 INCHES, 3784 CENTIMETERS"
```

To reset the these values to 0, type:

```
! U1 setvar "odometer.headclean" "0"
```

To confirm this settings were reset, type:

```
! U1 getvar "odometer.headclean"
```

If the resetting was successful, this is shown:

```
"0 INCHES, 0 CENTIMETERS"
```

odometer.headnew

This printer setting refers to the head replaced odometer count. This counter tracks how many inches and centimeter passed through the printer since the head was last replaced.

Setvar

To instruct the printer to reset the head new counter:

```
! U1 setvar "odometer.headnew" "value"
```

Values

"0" resets the head new counter

Default

Must be an accepted value or it is ignored

Getvar

To instruct the printer to retrieve the values for the head new counter:

```
! U1 getvar "odometer.headnew"
```

Example

This example shows how to get the odometer head new, how to reset it, and how to confirm the settings changed:

To see the current settings, type:

```
! U1 getvar "odometer.headnew"
```

Something similar to this is shown:

```
"1489 INCHES, 3784 CENTIMETERS"
```

To reset the these values to 0, type:

```
! U1 setvar "odometer.headnew" "0"
```

To confirm this settings were reset, type:

```
! U1 getvar "odometer.headnew"
```

If the resetting was successful, this is shown:

```
"0 INCHES, 0 CENTIMETERS"
```

odometer.label_dot_length

This command returns the length of the last label printed or fed (in dots).

Getvar

To return the length of the last label printed or fed (in dots):

```
! U1 getvar "odometer.label_dot_length"
```

Example

This is an example of how to reset the length using the ^LL command and how to use the `getvar` to confirm the change. For the ^LL command to work, the printer must be in continuous mode.

To change the odometer label dot length, type:

```
^XA
^LL500
^XZ
```

To get the current odometer label dot length, type:

```
! U1 getvar "odometer.label_dot_length"
```

Something similar to this is shown:

```
"500"
```

odometer.media_marker_count

This command refers to the non-resettable media marker count. The media marker counter keeps track of how many labels have passed through the printer by counting the bar sense marks on the back of the media or the gap in gap media. Labels are counted whether or not they have been printed.

Setvar

To set the non-resettable media marker count:

```
! U1 setvar "odometer.media_marker_count" "value"
```

Values

"0" to "4294967295"

Default

"0"

Getvar

To return the current setting value:

```
! U1 getvar "odometer.media_marker_count"
```

Result

```
"105"
```

odometer.media_marker_count1

This printer setting refers to the value of the first (count1) user resettable counter. The user resettable counters track how much media has passed through the printer in both inches or centimeters.

Setvar

To reset the first user resettable counter:

```
! U1 setvar "odometer.media_marker_count1" "value"
```

Values

"0" resets the counter

Default

Must be an accepted value or it is ignored.

Getvar

To return the current value of the first (count1) user resettable counter in both inches and centimeters:

```
! U1 getvar "odometer.media_marker_count1"
```

Example

This example shows how to get the first user resettable counter, how to reset it, and how to confirm the settings have changed:

To see the current settings, type:

```
! U1 getvar "odometer.media_marker_count1"
```

Something similar to this is shown:

```
"8516 INCHES, 21632 CENTIMETERS"
```

To reset the these values to 0, type:

```
! U1 setvar "odometer.media_marker_count1" "0"
```

To confirm these settings were reset, type:

```
! U1 getvar "odometer.media_marker_count1"
```

If the resetting was successful, this is shown:

```
"0 INCHES, 0 CENTIMETERS"
```

odometer.media_marker_count2

This printer setting refers to the value of the second (count2) user resettable counter. The user resettable counters track how much media has passed through the printer in both inches or centimeters.

Setvar

To reset the second user resettable counter:

```
! U1 setvar "odometer.media_marker_count2" "value"
```

Values

"0" resets the counter

Default

Must be an accepted value or it is ignored.

Getvar

To return the current value of the second (count2) user resettable counter in both inches and centimeters:

```
! U1 getvar "odometer.media_marker_count2"
```

Example

This example shows how to get the second user resettable counter, how to reset it, and how to confirm the settings have changed:

To see the current settings, type:

```
! U1 getvar "odometer.media_marker_count2"
```

Something similar to this is shown:

```
"8516 INCHES, 21632 CENTIMETERS"
```

To reset the these values to 0, type:

```
! U1 setvar "odometer.media_marker_count2" "0"
```

To confirm these settings were reset, type:

```
! U1 getvar "odometer.media_marker_count2"
```

If the resetting was successful, this is shown:

```
"0 INCHES, 0 CENTIMETERS"
```

odometer.net_media_length

This command displays the total amount of media used by the printer. The printer tracks all forward motion, subtracting all backward motion, so the odometer always indicates the total amount of media consumed by the printer. The media is measured in inches and centimeters.

Getvar

To return the current length, in centimeters, of media used by the printer:

```
! U1 getvar "odometer.net_media_length"
```

Result

This example shows that the printer used 9,492 inches, or 24,111 centimeters, of media:

```
"9492 INCHES, 24111 CENTIMETERS"
```

odometer.net_ribbon_length

This command displays the total amount of ribbon used by the printer. All forward and backward motion is tracked to determine the amount of ribbon used. The ribbon length is measured in inches and centimeters.

Getvar

To return the current length, in centimeters, of ribbon used by the printer:

```
! U1 getvar "odometer.net_ribbon_length"
```

Result

This example shows that the printer used 244 inches, or 621 centimeters, of ribbon:

```
"244 INCHES, 621 CENTIMETERS"
```

odometer.retracts_count

This printer value records the number of times a label has been retracted since the last time the counter has been reset.

Setvar

To reset the current count of retractions:

```
! U1 setvar "odometer.retracts_count" "value"
```

Values

`"0"` resets the counter

Default

NA

Getvar

To respond with the current number of retractions that have happened since the last time the counter was reset:

```
! U1 getvar "odometer.retracts_count"
```

odometer.rfid.valid_resettable

This command resets the RFID valid label counter to zero.

Setvar

To set the RFID valid counter to zero:

```
! U1 setvar "odometer.rfid.valid_resettable" "value"
```

Values

"0"

Getvar

To respond with the current RFID valid counter value:

```
! U1 getvar "odometer.rfid.valid_resettable"
```

Example

This `setvar` example shows how the counter portion of the printer configuration labels looks when the RFID valid counter is reset by sending:

```
! U1 setvar "odometer.rfid.valid_resettable" "0"
```

Figure 18 Before

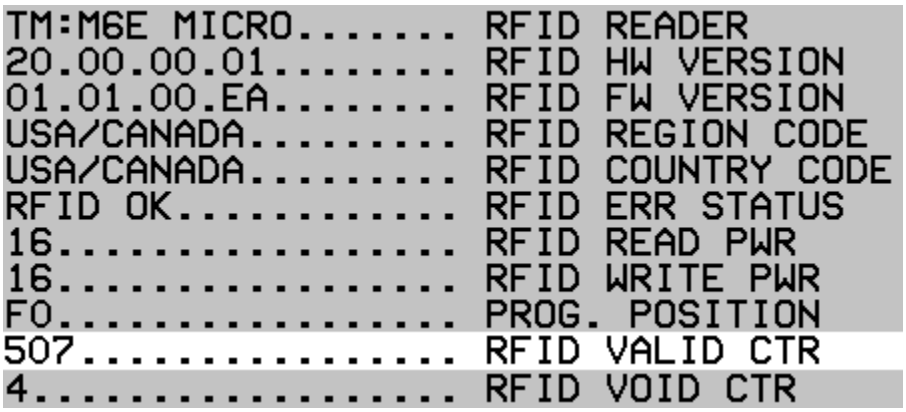
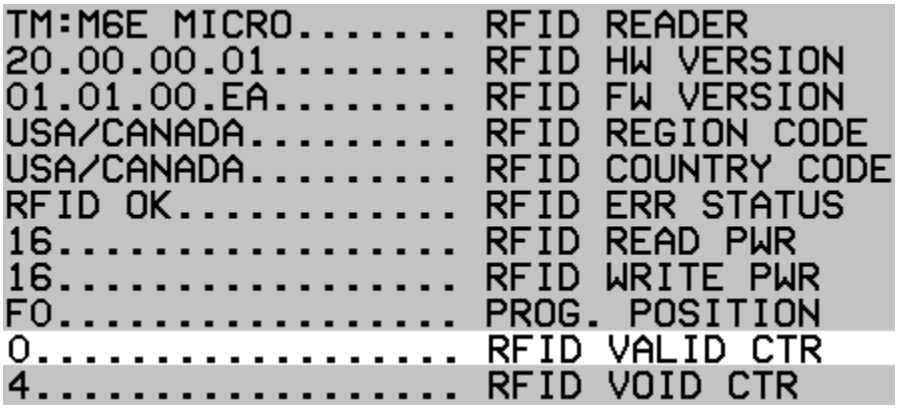


Figure 19 After



odometer.rfid.void_resettable

This command resets the RFID void label counter to zero.

Setvar

To set the RFID void counter to zero:

```
! U1 setvar "odometer.rfid.void_resettable" "value"
```

Values

"0"

Getvar

To respond with the current RFID void counter value:

```
! U1 getvar "odometer.rfid.void_resettable"
```

Example

This `setvar` example shows how the counter portion of the printer configuration labels looks when the RFID void counter is reset by sending:

```
! U1 setvar "odometer.rfid.valid_resettable" "0"
```

Figure 20 Before

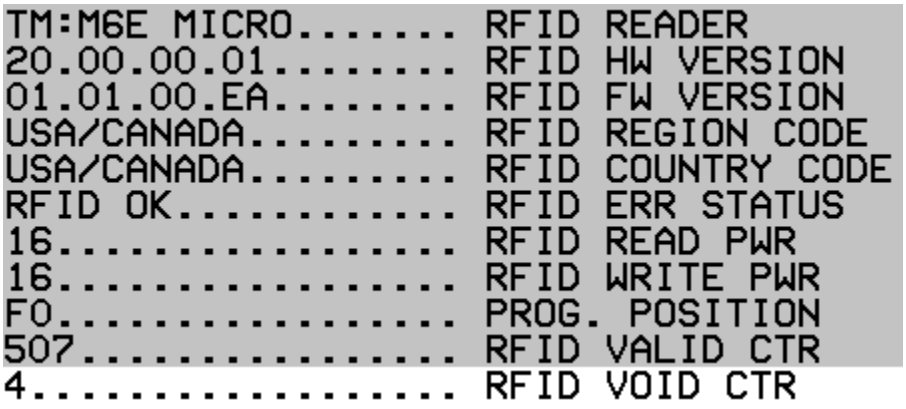
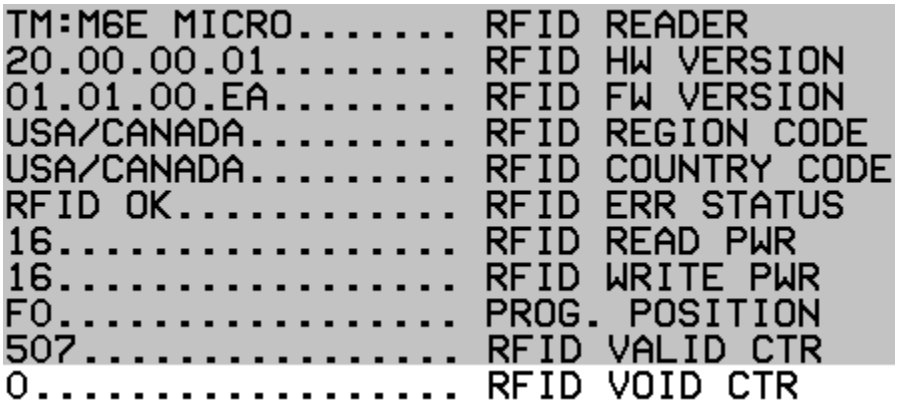


Figure 21 After



odometer.total_cuts

Displays the total number of cuts incurred by the cutter.

Getvar

To return the current setting value:

```
! U1 getvar "odometer.total_cuts"
```

Values
An integer

odometer.total_print_length

This command returns the total length of all media movement over the life of the printer.

NOTE: The number returned includes all media movement including backfeeds.

Getvar

To return the current setting value:

```
! U1 getvar "odometer.total_print_length"
```

Default

"0"

Example

To get the total length of media printed to date:

```
! U1 getvar "odometer.total_print_length"
(sample) "8560 INCHES, 21744 CENTIMETERS"
```

odometer.total_label_count

This command returns the total number of labels printed over the life of the printer.

NOTE: The number returned does not include form feeds or calibration labels.

Getvar

To return the current setting value:

```
! U1 getvar "odometer.total_label_count"
```

Example

To get the total number of labels printed to date:

```
! U1 getvar "odometer.total_label_count"
(sample) "31084"
```

odometer.user_label_count

Returns the number of labels printed since the last odometer set command.

Setvar

To set the user label count:

```
! U1 setvar "odometer.user_label_count" "value"
```

Values

"0" to "65000"

Related ZPL Commands

~RO 1

Getvar

To return the current setting value:

```
! U1 getvar "odometer.user_label_count"
```

Example

To get the total number of labels printed to date:

```
! U1 getvar "odometer.user_label_count"
(sample) "7544"
```

odometer.user_total_cuts

This command sets the number of cuts incurred by the cutter. This is the resettable version of the odometer.total_cuts SGD.

Setvar

To set the number of cuts incurred by the cutter:

```
! U1 setvar "odometer.user_total_cuts" "0"
```

Values

`"0"` resets the cut counter.

Getvar

To return the current number of cuts since the last time the cut counter was reset:

```
! U1 getvar "odometer.user_total_cuts"
```

Result

`"0"` to `"n"`

Here `"n"` is an integer

odometer.user_label_count[1|2]

Returns the number of labels printed since the last reset of each resettable odometer.

Setvar

To resets the counter value to 0:

```
! U1 setvar "odometer.user_label_count1" "value"
! U1 setvar "odometer.user_label_count2" "value"
```

Values

"0" to "4294967295"

Related ZPL Commands

~R0

Getvar

To return the current setting value:

```
! U1 getvar "odometer.user_label_count1"
! U1 getvar "odometer.user_label_count2"
```

Example

To get the total number of labels printed on to date:

```
! U1 getvar "odometer.user_label_count1"
"164"
```

odometer.latch_open_count

Returns the number of times the latch for the printhead has been opened.

Setvar

To return the number of times the latch for the printhead has been opened:

```
! U1 setvar "odometer.latch_open_count" "value"
```

Values

"0" to "4294967295"

Getvar

To return the number of times the latch for the printhead has been opened:

```
! U1 getvar "odometer.latch_open_count"
```

parallel_port.mode

This command sets the mode type for the parallel port.

Setvar

To set the mode type for the parallel port:

```
! U1 setvar "parallel_port.mode" "value"
```

Values

- `"bidirectional"`
- `"unidirectional"`

Default

`"bidirectional"`

Getvar

To retrieve the current mode type setting for the parallel port:

```
! U1 getvar "parallel_port.mode"
```

Example

```
! U1 setvar "parallel_port.mode" "bidirectional"
```

parallel_port.present

This command reports if there is a parallel port in the printer.

Getvar

To report if there is a parallel port in the printer:

```
! U1 getvar "parallel_port.present"
```

Result

- "present"
- "not installed"

power.average_current

Returns the battery pack average current value for mA for printers supporting Power Precision Plus batteries.

Getvar

To return the current value of the setting:

```
! U1 getvar "power.average_current"
```

Values

An integer

power.battery_led_blink_rate

Sets the Extended Smart Battery LED blink rate. The rate is set in multiples of 0.5 seconds.

Setvar

To set the Extended Smart Battery LED blink rate:

```
! U1 setvar "power.battery_led_blink_rate" "value"
```

Values

A number from `"0"` to `"127"`. The rate is set in multiples of 0.5 seconds.

Default

`"2"`

Getvar

To return the current setting value:

```
! U1 getvar "power.battery_led_blink_rate"
```

Example

The rate is set in multiples of 0.5 seconds. To achieve an On time of 1 second and an Off time of 4.5 seconds, one would use the following configuration:

On Duration = 2 (2 * 0.5 = 1 second), see power.battery_led_on_duration on page 941

Off Duration = 9 (9 * 0.5 = 4.5 seconds), see power.battery_led_off_duration on page 940

Blink Rate = 2 (2 * 0.5 = 1 second)

power.battery_led_enable

This command will enable or disable the illumination of the Extended Smart battery LED when one or more of the `power.battery` thresholds have been reached.

Setvar

To enable or disable the illumination of the Extended Smart battery LED whe-n one or more of the `power.battery` thresholds have been reached:

```
! U1 setvar "power.battery_led_enable" "value"
```

Values

`"on"` enables the Extended Smart battery LED

`"off"` disables the Extended Smart battery LED

Default

`"on"`

Getvar

To return the current setting value:

```
! U1 getvar "power.battery_led_enable"
```

power.battery_led_off_duration

Sets the Extended Smart Battery Led Off duration. The rate is set in multiples of 0.5 seconds.

Setvar

To set the Extended Smart Battery Led Off duration:

```
! U1 setvar "power.battery_led_off_duration" "value"
```

Values

A number between `"0"` and `"255"`. The rate is set in multiples of 0.5 seconds.

Default

`"9"`

Getvar

To return the current setting value:

```
! U1 getvar "power.battery_led_off_duration"
```

power.battery_led_on_duration

Sets the Extended Smart Battery Led On duration. The rate is set in multiples of 0.5 seconds.

Setvar

To set the Extended Smart Battery Led On duration:

```
! U1 setvar "power.battery_led_on_duration" "value"
```

Values

A number between `"0"` and `"255"`. The rate is set in multiples of 0.5 seconds.

Default

`"2"`

Getvar

To return the current setting value:

```
! U1 getvar "power.battery_led_on_duration"
```

power.battery_type

This command retrieves the battery type installed in the printer.

Getvar

To get the type of battery installed in the printer:

```
! U1 getvar "power.battery_type"
```

Default

`"unmanaged"`

Result

`"sb"`	smart battery
`"ppp"`	power precision plus
`"none"`	no battery
`"unmanaged"`	legacy unmanaged battery

Example

In the example below, the `getvar` retrieves the battery type installed in the printer.

```
! U1 getvar "power.battery_type" "sb"
```

NOTE: QLn and ZQ5 are not capable of authenticating a Power Precision Plus battery so these printers will report `"sb"`.

power.dtr_power_off

This command refers to the remote printer power control, and is used for power management. When Data Terminal Ready (DTR) is enabled the printer can be powered on and off via the Data Set Ready (DSR) signal. When DTR power off is enabled, a low to high transition will cause the printer to turn ON and a high to low transition will cause the printer to turn OFF.

NOTE: The inactivity time-out is disabled while DSR is active.

Setvar

To turn DTR power on or off:

```
! U1 setvar "power.dtr_power_off" "value"
```

Values

"off"

"on"

Default

"on"

Getvar

To retrieve the current DTR power-off setting:

```
! U1 getvar "power.dtr_power_off"
```

Example

```
! U1 setvar "power.dtr_power_off" "off"
```

power.energy_star.enable

Enables the Energy Star functionality.

For more information on Energy Star, see http://www.energystar.gov.

Setvar

To enable or disable the Energy Star functionality:

```
! U1 setvar "power.energy_star.enable" "value"
```

Values

- `"on"` enables the Energy Star functionality
- `"off"` disables the Energy Star functionality

Getvar

To return the current setting value:

```
! U1 getvar "power.energy_star.enable"
```

power.energy_star.timeout

Sets the amount of idle time before Energy Star mode in invoked. The time is specified is in seconds.

For more information on Energy Star, see http://www.energystar.gov.

Setvar

To set the amount of idle time before Energy Star mode in invoked:

```
! U1 setvar "power.energy_star.timeout" "value"
```

Values

`"180"` to `"65535"` seconds

Default

`"180"`

Getvar

To return the current setting value:

```
! U1 getvar "power.energy_star.timeout"
```

Example

This `setvar` example shows the value set to `"260"`.

```
! U1 setvar "power.energy_star.timeout" "260"
```

The `setvar` value is the `getvar` result. In this example, the `getvar` result is `"260"`.

power.label_queue.shutdown

Specifies if the printer should wait to shut down until all labels in its internal queue have been printed.

Setvar

To specify the label queue shutdown time:

```
! U1 setvar "power.label_queue.shutdown" "value"
```

Values

- `"yes"` specifies that the printer will wait to shut down until all labels in its internal queue have been printed
- `"no"` specifies that the printer will not wait to shut down until all labels in its internal queue have been printed

Default

`"no"`

Getvar

To return the current setting value:

```
! U1 getvar "power.label_queue.shutdown"
```

power.power_on_mode

Indicates if the printer will power on automatically when power is applied, i.e., when the power supply is plugged in.

Getvar

To return the current setting value:

```
! U1 getvar "power.power_on_mode"
```

Values

- `"auto"` means the jumper is present on option card, which makes the printer power on automatically when power is applied.
- `"manual"` means jumper is not present on option card or the option card doesn't support auto-power on, so the printer will power on only when the user presses the power button.
- `"not available"` means this is not an option on the printer.

Default

`"on"`

power.shutdown

Instructs the printer to shut down.

Do

To shut down the printer:

```
! U1 do "power.shutdown" ""
```

power.voltage

This command returns the current battery voltage.

Getvar

To return the current setting:

```
! U1 getvar "power.voltage"
```

Result

Current voltage reading in integers.

power.wake.radio

This command is used to enable or disable the power wake feature on printers that are radio (WLAN, BT Classic, and BTLE) enabled. The radio must be enabled to support waking on that interface. i.e. wlan.enable must be set to "yes" to support the wake feature on WLAN.

Setvar

To enable or disable the power wake setting:

```
! U1 setvar "power.wake.radio" "values"
```

Values

"on" "off"	ZQ6
"on" "off"	ZQ3
"on"	ZQ3 with BT Classic/BTLE radio installed (BT Only - UART)
"off"	ZQ5 with WLAN/BT Classic radio installed
"on"	ZQ5 with BT Classic/BTLE radio installed (BT Only - UART)
"on" "off"	ZD4xx, ZD6xx with WLAN/BT Classic/BTLE installed
"off"	ZD4xx, ZD6xx with BTLE only radio or no radio installed

Default

"on"

Getvar

To return the current setting:

```
! U1 getvar "power.wake.radio"
```

power.current

This command returns the battery pack instantaneous current value in mA for printers supporting Power Precision Plus batteries.

Positive values indicate charging current, whereas negative values indicate discharging current.

Getvar

To return the battery pack instantaneous current value:

```
! U1 getvar "power.current"
```

Result

"-32768" to "32767" mA

Example

In the example below, the `getvar` returns the battery pack instantaneous current value.

```
! U1 getvar "power.current" "5643 mA"
```

power.temperature

Returns the current battery temperature in degrees Celsius for printers that support a Power Precision Plus and Smart Batteries.

Getvar

To get the Power Precision battery temperature in Celsius:

```
! U1 getvar "power.temperature"
```

Example

In the example below, the `getvar` returns the current battery temperature in Celsius.

```
! U1 getvar "power.temperature""25.40 C"
```

power.percent_health

This command returns the percent health that is read from the fuel gauge for printers that support a Power Precision Plus battery. The battery health is expressed as a percentage of design capacity.

Getvar

To get the Power Precision battery health percentage:

```
! U1 getvar "power.percent_health"
```

Result

`"0"` to `"100"`

Example

In the example below, the `getvar` returns the Power Precision battery health percentage.

```
! U1 getvar "power.percent_health" "90"
```

power.part_number

Returns the battery part number for printers that support Power Precision Plus batteries.

Getvar

To get the Power Precision battery part number:

```
! U1 getvar "power.part_number"
```

Result

<=10 digit string

Example

In the example below, the `getvar` returns the battery part number for Power Precision battery printers.

```
! U1 getvar "power.part_number"  "0123456789"
```

power.sleep.cradle

This command enables or disables the sleep timeout feature while the printer is docked in the cradle. In the ZQ5 printer, the sleep timeout in the cradle is enabled by default. In the ZQ3 and ZQ6 printers, the sleep timeout in the cradle feature is disabled by default. This is done so as to manage the printer not having the Wake on BT, Wake on WLAN, and Wake feature on Ethernet support.

The command only affects the sleep timeout in cradle. If the user presses the power button, then the printer can still go to sleep, regardless of the setting.

Setvar

To set the command:

```
! U1 setvar "power.sleep.cradle" "values"
```

Values

- `"enabled"` indicates that `power.sleep.timeout` is honored while the printer is docked in a cradle.
- `"disabled"` indicates that `power.sleep.timeout` is disabled while the printer is docked in a cradle.

Default Value

`"disabled"` for ZQ3, ZQ6

`"enabled"` for ZQ5

Getvar

To return the current setting:

```
! U1 getvar "power.sleep.cradle"
```

Example

In the example below, the getvar returns the current setting of the sleep timeout in cradle feature.

```
! U1 getvar "power.sleep.cradle" "enabled"
```

power.remaining_capacity

This command returns the remaining capacity of the battery in milliamp hours (mAh).

Getvar

To return the remaining battery capacity:

```
! U1 getvar "power.remaining_capacity"
```

Result

`"0 mAh"` to `"65535 mAh"`

Example

In the example below, the `getvar` returns the remaining battery capacity of "1846 mAh".

```
! U1 getvar "power.remaining_capacity" "1846 mAh"
```

power.cycle_count

This command returns the number of charge cycles the battery has performed. A cycle is defined as a discharge of 80% of the pack's full charge capacity plus the concatenated partial charges that add to 80% of the pack's full charge capacity.

Getvar

To return the number of charge cycles:

```
! U1 getvar "power.cycle_count"
```

Example

In the example below, the `getvar` returns the number of charge cycles the battery has performed.

```
! U1 getvar "power.cycle_count" "77"
```

print.legacy_compatibility

This command turns off or on the legacy compatibility print quality.

Setvar

To enable or disable the legacy compatibility print quality:

```
! U1 setvar "print.legacy_compatibility" "value"
```

Values

- `"on"` uses legacy QLn print quality tables. Applicable to ZQ610 and ZQ620 printers only. Not supported on ZQ630.
- `"off"` uses ZQ6 print quality tables.

Default

`"off"`

Getvar

To return the current setting value:

```
! U1 getvar "print.legacy_compatibility"
```

print.tone

This command specifies the printer darkness.

Setvar

To set the darkness and relative darkness:

```
! U1 setvar "print.tone" "value"
```

Values

- "0.0" to "30.0" to adjust darkness
- "-0.1" to "-30.0" and "+0.1" to "+30.0" for incremental adjustments

Default

"4.0"

Getvar

To retrieve the printer's current darkness setting:

```
! U1 getvar "print.tone"
```

Example

This setvar example sets the value to "4.0".

```
! U1 setvar "print.tone" "4.0"
```

When the setvar value is set to "4.0", the getvar result is "4.0".

print.troubleshooting_label_print

Sets whether batch counters will be displayed on the printer's control panel.

Setvar

To set whether batch counters will be displayed on the printer's control panel:

```
! U1 setvar "print.troubleshooting_label_print" "value"
```

Values

- `"enabled"` specifies that batch counters will be displayed
- `"disabled"` specifies that batch counters will not be displayed

Default

`"disabled"`

Getvar

To return the current setting value:

```
! U1 getvar "print.troubleshooting_label_print"
```

Result

`"enabled"`

ribbon.serial_number

Sets the ribbon's part number.

Setvar

To set the ribbon's part number:

```
! U1 setvar "ribbon.serial_number" "value"
```

Values

An alpha-numeric string between 0 and 64 characters.

Default

NA

Example

```
! U1 setvar "ribbon.serial_number" "A34567BC6789"
```

Getvar

To return the current setting value:

```
! U1 getvar "ribbon.serial_number"
```

Result

"A34567BC6789"

ribbon.part_number

Sets the ribbon's part number.

Setvar

To set the ribbon's part number:

```
! U1 setvar "ribbon.part_number" "value"
```

Values

An alpha-numeric string between 0 and 64 characters.

Default

NA

Example

```
! U1 setvar "ribbon.part_number" "123AB987"
```

Getvar

To return the current setting value:

```
! U1 getvar "ribbon.part_number"
```

Result

```
"123AB987"
```

ribbon.cartridge.part_number

This command retrieves the part number of the ribbon cartridge installed in the printer. There is a 12 character max for the size of string returned since the cartridge allows for 10 character part numbers.

If a ribbon cartridge is not installed, or if the ribbon cartridge option is not present, then the command returns an empty string.

Getvar

To return the part number of the ribbon cartridge:

```
! U1 getvar "ribbon.cartridge.part_number"
```

Result

`"value"` <= 12 characters

`""` means ribbon cartridge is not installed or not available

Example

In this example, the `getvar` returns the part number of the ribbon cartridge.

```
! U1 getvar "ribbon.cartridge.part_number" "123456789A"
```

ribbon.cartridge.length_remaining

This command retrieves the length of ribbon remaining on the cartridge. This is specified in meters.

If a cartridge is not installed, the printer returns "0". If the cartridge option is not available in the printer, then the printer returns an empty string.

Getvar

To return the length of ribbon remaining on the cartridge:

```
! U1 getvar "ribbon.cartridge.length_remaining"
```

Result

- "0" to "74" meters
- "0" means cartridge not installed
- "" cartridge is not available

ribbon.cartridge.length

This command returns the original length of the ribbon cartridge installed in the printer. This is specified in meters. If a ribbon cartridge is not installed, then the printer returns "0". If the ribbon cartridge option is not present, then the command returns an empty string.

Getvar

To return the current setting:

```
! U1 getvar "ribbon.cartridge.length"
```

Result

- `"0"` indicates the cartridge is not installed
- `""` indicates the cartridge option is not present

Example

In this example, the `getvar` returns the original length of the ribbon cartridge.

```
! U1 getvar "ribbon.cartridge.length"  "100"
```

ribbon.cartridge.authenticated

This command returns the printer cartridge authentication status. The setting gets updated every time an authentication occurs (power up, head close, or any other time).

If a ribbon cartridge is not installed, then the printer returns "not installed". If the ribbon cartridge option is not present, then the command returns an empty string.

Getvar

To return the printer cartridge authentication status:

```
! U1 getvar "ribbon.cartridge.authenticated"
```

Result

`"yes"`	The cartridge installed is authenticated.
`"no"`	The cartridge installed is not authenticated.
`"not installed"`	The printer supports ribbon cartridge, but it is not installed (initial condition prior to authentication).
`"" (empty string)`	The printer does not support ribbon cartridge.

Example

In this example, the `getvar` returns that the cartridge is authenticated.

```
! U1 getvar "ribbon.cartridge.authenticated" "yes"
```

ribbon.cartridge.inserted

This command returns if the ribbon cartridge is inserted or not. The command is hidden in an ALLCV.

Getvar

To return if the cartridge is inserted o not:

```
! U1 getvar "ribbon.cartridge.inserted"
```

Result

`"yes"`	The cartridge is inserted.
`"no"`	The cartridge is not inserted or the cartridge mechanism does not exist, but the cartridge mechanism is an option on this printer platform.

ribbon.coating

This command sets the type of ribbon coating used.

Getvar

To determine the setting for the ribbon coating:

```
! U1 getvar “ribbon.coating”
```

Values

- `"ink side in"`
- `"ink side out"`

Setvar

To configure the printer for the printer ribbon:

```
! U1 setvar “ribbon.coating" "value"
```

Values

- `"ink side in"`
- `"ink side out"`

ribbon.tension

This command sets the tension of the ribbon.

Getvar

To report the current tension of the ribbon:

```
! U1 getvar “ribbon.tension”
```

Values

- `"low"`
- `"medium"`
- `"high"`

Default

`"high"`

Setvar

To specify the tension of the ribbon:

```
! U1 setvar “ribbon.tension" "value"
```

Values

- `"low"`
- `"medium"`
- `"high"`

rtc.exists

This command reports whether the system has a real-time clock.

Getvar

To determine if the system has a real-time clock:

```
! U1 getvar "rtc.exists"
```

Values

- `"no"` the system does not have a real-time clock.
- `"yes"` the system has a real-time clock.

rtc.date

The command sets the system date. The command accepts the month-day-year format, for example, "11-02-2021". If the format is not correct or the date is not valid, the command is ignored. The printer needs to be reset using `! U1 do "device.reset" ""` to accept this command.

Getvar

To report the current system date:

```
! U1 getvar "rtc.date"
```

Value

The command reports the system date in the month-day-year format.

Setvar

To set the system date:

```
! U1 setvar "rtc.date" "value"
```

Value

Accepts a date in the month-day-year format. If the format is not correct or the date is not valid, the command is ignored.

Example

This `setvar` example shows the value set to November 2, 2021 (11-02-2021).

```
! U1 setvar "rtc.date" "11-02-2021"
```

The `setvar` value is the `getvar` result. In this example, the `getvar` result is `"11-02-2021"`.

rtc.time

This command sets the system time. The time value must be in the hour:minute:second format, for example, 11:32:11. If the format is not correct or the time is not valid, the command is ignored. The printer needs to be reset using `! U1 do "device.reset" ""` to accept this command.

Getvar

To retrieve the current system time:

```
! U1 getvar "rtc.time"
```

Value
Returns the current system time in the hour:minute:second format.

Setvar

To set the current system time:

```
! U1 setvar "rtc.time" "value"
```

Value
Accepts a time in hour:minute:second format. If the format is not correct or the time is not valid, the command is ignored.

Example

This `setvar` example shows the value set to 11:32 and 11 seconds (11:32:11).

```
! U1 setvar "rtc.time" "11:32:11"
```

The `setvar` value is the `getvar` result. In this example, the `getvar` result is `"11:32:11"`.

rtc.timezone

This command specifies the POSIX-compliant time zone string.

This string includes the following:

- the time zone character specifier
- the offset from UTC
- daylight savings time adjustment
- when to go on and off of daylight savings time (if it pertains to the timezone).

Setvar

To set the POSIX-compliant time zone string:

```
! U1 setvar "rtc.timezone" "value"
```

Values

www.iana.org/time-zones This site is updated periodically to reflect changes made by political bodies to time zone boundaries, UTC offsets, and daylight-saving rules.

Getvar

To return the current setting value:

```
! U1 getvar "rtc.timezone"
```

Example

If you live in New York in the United States, in the Eastern time zone, your `setvar` string may look like:

```
! U1 setvar "rtc.timezone" "EST5EDT4,M3.2.0/02:00:00,M11.1.0/02:00:00"
```

The `"value"` string can be translated as follows: EST5 (Eastern Standard Time; 5 hours off UTC), EDT4 (Eastern Daylight Time; 4 hours off UTC), running from 2AM (`/02`; fully qualified: `/02:00:00`) from the second Sunday in March (`M3.2.0/02`) through 2AM (`/02`; fully qualified: `/02:00:00`) on the first Sunday in November (`M11.1.0/02`).

`M` indicates the Month follows, followed by the two-digit month, the week (`1` is the first week in which the specified weekday occurs, and `5` indicates the last week of the month with that weekday) and the weekday (`0` is Sunday). The time starts with a slash, and unspecified trailing fields default to zero.

Other examples for locations in the United States:

- US Central:

```
! U1 setvar "rtc.timezone" "CST6CDT5,M3.2.0/02,M11.1.0/02"
```

- US Mountain:

```
! U1 setvar "rtc.timezone" "MST7MDT6,M3.2.0/02,M11.1.0/02"
```

- US Pacific:

```
! U1 setvar "rtc.timezone" "PST8PDT7,M3.2.0/02,M11.1.0/02"
```

- US Alaska:

```
! U1 setvar "rtc.timezone" "AST9ADT8,M3.2.0/02,M11.1.0/02"
```

- US Hawaii:

```
! U1 setvar "rtc.timezone" "HST10"
```

rtc.unix_timestamp

This command sets or gets the printer time based on the Unix Epoch (UTC) number of seconds since January 1, 1970.

Setvar

To set the command:

```
! U1 setvar "rtc.unix_timestamp" "123123"
```

Values

"0" to "0xFFFFFFFF"

Getvar

To get the current printer time in seconds since 1970:

```
! U1 getvar "rtc.unix_timestamp"
```

sensor.air_pressure.current_reading

This command returns the atmospheric pressure in millibars.

Getvar

To determine the atmospheric pressure from the sensor:

```
! U1 getvar "sensor.air_pressure.current_reading"
```

Values

- `"0.0"` to `"4095.0"` millibars.
- `""` indicates that the sensor is not installed.

sensor.ambient_light.current_reading

This command returns the ambient light value. The ambient light reading is reported in lux.

Getvar

To return the amount of ambient light read by the sensor:

```
! U1 getvar "sensor.ambient_light.current_reading"
```

Values

- `"0"` to `"73000"` lux.
- `""` indicates that the sensor is not installed.

sensor.battery.in_volts

This command retrieves information on the battery current in volts.

Getvar

To return the current setting:

```
! U1 getvar "sensor.battery.in_volts"
```

Result

`"0.0"` to `"12.0"` volts

Example

In the `getvar` example below, the battery current volt reading of `"7.6"` is returned.

```
! U1 getvar "sensor.battery.in_volts" "7.6"
```

sensor.back_bar.brightness

This command retrieves the current back bar sensor brightness level.

Getvar

To return the back bar sensor brightness level:

```
! U1 getvar "sensor.back_bar.brightness"
```

Example

In the example below, the `getvar` retrieves the back bar sensor brightness level of `"10"`.

```
! U1 getvar "sensor.back_bar.brightness" "10"
```

sensor.back_bar.ppr_out_thold

This command retrieves the current back bar sensor paper out threshold value.

Getvar

To return the current sensor back bar paper out threshold value:

```
! U1 getvar "sensor.back_bar.ppr_out_thold"
```

Example

In the `getvar` example below, the current sensor back bar threshold value of `"10"` is returned.

```
! U1 getvar "sensor.back_bar.ppr_out_thold" "10"
```

sensor.back_bar.cur

This command retrieves the current back bar sensor value.

Getvar

To return the current back bar sensor value:

```
! U1 getvar "sensor.back_bar.cur"
```

Result

`"0"` to `"255"`

Example

In the `getvar` example below, the current sensor back bar value of `"10"` is returned.

```
! U1 getvar "sensor.back_bar.ppr_out_thold" "10"
```

sensor.cover_open

This printer setting determines if the printer media cover is open.

Getvar

To display if the printer cover is open or not:

```
! U1 getvar "sensor.cover_open"
```

Values

- `"yes"` means the printer cover is open
- `"no"` means the printer cover is closed

sensor.front_bar.ppr_out_thold

This command retrieves the current paper out threshold level of the front bar sensor of the printer.

Getvar

To return the current paper out threshold level:

```
! U1 getvar "sensor.front_bar.paper_out_threshold"
```

Example

In the `getvar` example below, the paper out threshold value of `"10"` is returned.

```
! U1 getvar "sensor.front_bar.paper_out_threshold" "10"
```

sensor.front_bar.cur

This command retrieves the current front bar sensor value.

Getvar

To return the current front bar sensor value:

```
! U1 getvar "sensor.front_bar.cur"
```

Result

`"0"` to `"255"`

Example

In the `getvar` example below, the front bar sensor value of `"10"` is returned.

```
! U1 getvar ! U1 getvar "sensor.front_bar.cur""10"
```

sensor.front_bar.thold

This command retrieves the current front bar sensor threshold value.

Getvar

To return the current front bar sensor threshold value:

```
! U1 getvar "sensor.front_bar.thold"
```

Example

In the `getvar` example below, the current front bar sensor threshold value of `"4"` is returned.

```
! U1 getvar "sensor.front_bar.thold" "4"
```

sensor.front_bar.gain

This command retrieves the current front bar sensor gain level.

Getvar

To return the current front bar sensor gain level:

```
! U1 getvar "sensor.front_bar.gain"
```

Example

In the `getvar` example below, the front bar sensor gain level of `"10"` is returned.

```
! U1 getvar "sensor.front_bar.gain" "10"
```

sensor.front_bar.brightness

This command retrieves the current front bar sensor brightness.

Getvar

To return the current front bar sensor brightness:

```
! U1 getvar "sensor.front_bar.brightness"
```

Example

In the `getvar` example below, the front bar sensor brightness level of `"10"` is returned.

```
! U1 getvar "sensor.front_bar.brightness" "10"
```

sensor.front_bar.offset

This command retrieves the current front bar sensor offset value.

Getvar

To return the current front bar sensor offset value:

```
! U1 getvar "sensor.front_bar.offset"
```

Example

In the `getvar` example below, the front bar sensor offset value of `"10"` is returned.

```
! U1 getvar "sensor.front_bar.offset" "10"
```

sensor.back_bar.offset

This command retrieves the current back bar sensor offset value.

Getvar

To return the current setting:

```
! U1 getvar "sensor.back_bar.offset"
```

Example

In the example below, the `getvar` retrieves the back bar sensor offset value of `"10"`.

```
! U1 getvar "sensor.back_bar.offset" "10"
```

sensor.gap.thold

This command retrieves the current gap sensor threshold level.

Getvar

To return the current gap sensor threshold level:

```
! U1 getvar "sensor.gap.thold"
```

Example

In the example below, the `getvar` retrieves the current gap sensor threshold value of `"10"`.

```
! U1 getvar "sensor.gap.thold" "10"
```

sensor.gap.offset

This command retrieves the current gap sensor offset value.

Getvar

To return the current gap sensor offset value:

```
! U1 getvar "sensor.gap.offset"
```

Example

In the example below, the `getvar` retrieves the current gap sensor offset value of `"10"`.

```
! U1 getvar "sensor.gap.offset" "10"
```

sensor.gap.gain

This command retrieves the current gap sensor gain level.

Getvar

To return the current gap sensor gain level:

```
! U1 getvar "sensor.gap.gain"
```

Example

In the example below, the `getvar` retrieves the current gap sensor gain level of `"10"`.

```
! U1 getvar "sensor.gap.gain" "10"
```

sensor.gap.brightness

This command retrieves the current gap sensor brightness level.

Getvar

To return the current gap sensor brightness level:

```
! U1 getvar "sensor.gap.brightness"
```

Example

In the example below, the `getvar` retrieves the current gap sensor brightness level of `"10"`.

```
! U1 getvar "sensor.gap.brightness" "10"
```

sensor.head.temp_avg

This command retrieves the current average head temperature in Celsius.

Getvar

To return the current average head temperature:

```
! U1 getvar "sensor.head.temp_avg"
```

Result

"-32768" to "32767" Celsius

Example

In the `getvar` example below, the head temperature average of "32" Celsius is returned.

```
! U1 getvar "sensor.head.temp_avg" "32"
```

sensor.head.temp_celsius

This command retrieves the current head temperature in Celsius.

Getvar

To return the current average head temperature:

```
! U1 getvar "sensor.head.temp_celsius"
```

Result

"-32768" to "32767" Celsius

Example

In the `getvar` example below, the head temperature average of `"0"` is returned.

```
! U1 getvar "sensor.head.temp_celsius" "0"
```

sensor.head.temp

This command retrieves the current head temperature of the printer.

Getvar

To return the current average head temperature:

```
! U1 getvar "sensor.head.temp"
```

Result

"-32768" to "32767" Celsius

Example

In the `getvar` example below, the head temperature average of "32" is returned.

```
! U1 getvar "sensor.head.temp" "32"
```

sensor.magnetometer.current_reading

This command returns a magnetic induction reading in gauss.

Getvar

To return the magnetic induction reading from the sensor:

```
sensor.magnetometer.current_reading
```

Values

The magnetic induction in gauss. If the sensor is not installed, the `getvar` returns `""`.

Example

In the following example, the `getvar` returns the magnetic induction reading from the sensor:

```
! U1 getvar "sensor.magnetometer.current_reading"
```

```
"{"x":-12.3455,"y":1.2345,"z":0.1234}"
```

The `getvar` returns the magnetic field reading as a vector with x, y, and z components, each in the range of +/-50 gauss.

sensor.object_temperature.current_reading

This command returns the temperature, in Celsius, from the sensor.

Getvar

To return the temperature from the object sensor:

```
! U1 getvar "sensor.object_temperature.current_reading"
```

Values

- Object temperature, in degrees Celsius, from `"-20.0"` to `"200.0"`.
- `""` indicates that the temperature sensor is not installed.

sensor.peel.thold

This command retrieves the current peel sensor threshold level.

Getvar

To return the current peel sensor threshold level:

```
! U1 getvar "sensor.peel.thold"
```

Example

In the example below, the `getvar` retrieves the current peel threshold level of `"5"`.

```
! U1 getvar "sensor.peel.thold" "5"
```

sensor.peel.gain

This command retrieves the current peel sensor gain level.

Getvar

To return the current peel sensor gain level:

```
! U1 getvar "sensor.peel.gain"
```

Example

In the example below, the `getvar` retrieves the current peel gain level of `"2"`.

```
! U1 getvar "sensor.peel.gain" "2"
```

sensor.paper_supply

Returns current paper supply status.

Getvar

To return the current paper supply status:

```
! U1 getvar "sensor.paper_supply"
```

Result

- `"ok"` indicates the printer has paper.
- `"out"` indicates that the printer is out of paper.

sensor.peeler

Obtains current peeler sensor status.

Getvar

To retrieve the current peeler sensor status:

```
! U1 getvar "sensor.peeler"
```

Result

- `"clear"` means the last printed item has been removed, or there are no items waiting to be removed.
- `"not clear"` means the last printed item has not yet been removed.

sensor.peel.brightness

This command returns the current peel sensor brightness level.

Getvar

To return the current peel sensor brightness level:

```
! U1 getvar "sensor.peel.brightness"
```

Example

In the example below, the `getvar` returns the current peel sensor brightness level of `"2"`.

```
! U1 getvar "sensor.peel.brightness" "2"
```

sensor.proximity.current_reading

This command returns proximity information. The higher the number, the closer the item is.

Getvar

To return the proximity information from the sensor:

```
! U1 getvar "sensor.proximity.current_reading"
```

Values

- `"0"` to `"4095"` A higher number indicates closer proximity.
- `""` indicates that the sensor is not installed.

sensor.width.in_dots

This command retrieves the current width sensor value in dots.

Getvar

To return the current width sensor value in dots:

```
! U1 getvar "sensor.width.in_dots"
```

Example

In the example below, the `getvar` retrieves the current width sensor value that is `"200"` dpi.

```
! U1 getvar "sensor.width.in_dots" "200"
```

sensor.width.cur

This command retrieves the current width sensor value.

Getvar

To return the current width sensor value:

```
! U1 getvar "sensor.width.cur"
```

Result

`"0"` to `"255"`

Example

In the example below, the `getvar` retrieves the current width sensor value.

```
! U1 getvar "sensor.width.cur" "5"
```

sensor.self_adjusting_enable

Enables the self-adjusting gap sensor.

Setvar

To enable the self-adjusting gap sensor:

```
! U1 setvar "sensor.self_adjusting_enable" "value"
```

Values

- `"yes"` enables the self-adjusting gap sensor and disables the traditional gap sensor
- `"no"` disables the self-adjusting gap sensor and enables the traditional gap sensor

Getvar

To retrieve the self-adjusting gap sensor:

```
! U1 getvar "sensor.self_adjusting_enable"
```

Example

This `setvar` example shows the value set to `"yes"`.

```
! U1 setvar "sensor.self_adjusting_enable" "yes"
```

The `setvar` value is the `getvar` result. In this example, the `getvar` result is `"yes"`.

usb.device.device_id_string

This command retrieves the manufacturer assigned IEEE1284 Device Identification string for USB devices.

Getvar

To retrieve the device ID string:

```
! U1 getvar "usb.device.device_id_string"
```

Example

```
! U1 getvar "usb.device.device_id_string"
```

Result

```
"MANUFACTURER:Zebra Technologies ;COMMAND SET:ZPL;MODEL:ZTC
 ZT220-200dpi ZPL;CLASS:PRINTER;OPTIONS:XML;"
```

usb.device.device_unique_id

This command sets the USB Unique Device ID setting. The identifier that makes any printer unique is set by the `"usb.device.serial_string"` command which is reported to the USB driver.

By default `"usb.device.serial_string"` reports the printer's serial number. If `"usb.device_unique_id"` is set to `"off"` the printer will report the `usb.device.serial_string` parameter as its product family (e.g. ZT230, etc).

When subsequent printers of the same model, with `"usb.device_unique_id"` parameter to `"off"`, are connected via USB, the host computer will not treat them as a new Plug and Play events, nor require new driver installations.

Setvar

To set the current USB Unique Device Id setting:

```
! U1 setvar "usb.device.device_unique_id" "value"
```

Values

- `"off"`
- `"on"`

Default

`"on"`

Getvar

To return the current USB Unique Device Id setting stored in the printer:

```
! U1 getvar "usb.device.device_unique_id"
```

Example

```
! U1 setvar "usb.device.device_unique_id" "off"
```

usb.device.device_version

This command returns the version of the USB device being queried.

Getvar

This command instructs the printer to respond with the printer's USB version.

```
! U1 getvar "usb.device.device_version"
```

Example

```
! U1 getvar "usb.device.device_version"
"1.1"
```

usb.device.manufacturer_string

This command retrieves the USB device manufacturer's name.

Getvar

To respond with the manufacturer name:

```
! U1 getvar "usb.device.manufacturer_string"
```

Example

```
! U1 getvar "usb.device.manufacturer_string"
"Zebra Technologies"
```

usb.device.product_id

This command retrieves the Product Identification number that a manufacturer assigned to a particular product. This number, along with the Vendor ID, allows a USB host to distinguish between devices.

Getvar

To retrieve the product ID:

```
! U1 getvar "usb.device.product_id"
```

Example

```
! U1 getvar "usb.device.product_id"
"003D"
```

usb.device.product_string

This command returns the manufacturer-assigned string describing a particular USB product.

Getvar

To instruct the printer to respond with the product string description:

```
! U1 getvar "usb.device.product_string"
```

Example

Issuing the command on a ZT210 printer:

```
! U1 getvar "usb.device.product_string"
"ZT210"
```

Issuing the command on a QLn320 printer:

```
! U1 getvar "usb.device.product_string"
"ZTC QLn320-203dpi CPCL"
```

NOTE: For firmware V68.19.0 and V72.19.0, the return value was changed to the longer name, which is the same as the USB PID. The QLnXXX is replaced by the printer model and number for each printer.

usb.device.serial_string

This command returns the manufacturer-assigned serial number for a particular USB product.

Getvar

To return the printer's serial number:

```
! U1 getvar "usb.device.serial_string"
```

Example

```
! U1 getvar "usb.device.serial_string"
"ABC1234567890"
```

usb.device.vendor_id

This command returns the Vendor Identification number that the USB-IF organization has assigned to a manufacturer. This number, along with the Product ID, allows a USB host to distinguish between devices.

Getvar

To retrieve the vendor ID of the device:

```
! U1 getvar "usb.device.vendor_id"
```

Example

```
! U1 getvar "usb.device.vendor_id"
"0a5f"
```

usb.halt

This command controls whether the printer allows communication over the USB port when the printer is in an error condition.

Setvar

To control whether the printer allows communication over the USB port when the printer is in an error condition:

```
! U1 setvar "usb.halt" "value"
```

Values

- "yes"
- "no"

Default

"no"

Getvar

To return the current "usb.halt" setting stored in the printer:

```
! U1 getvar "usb.halt"
```

Example

```
! U1 setvar "usb.halt" "yes"
```

usb.host.config_info_to_usb

This command is used by WML to save the ^HH output files to the USB thumb drive. The file names are written in the CONFIGxxx.TXT format, where xxx ranges from 1 to 999.

Setvar

To enable or disable saving the ^HH output to the USB thumb drive:

```
! U1 setvar "usb.host.config_info_to_usb" "values"
```

Values

"yes" means ^HH output is saved

"no" means no action is taken

Example

In the setvar example below, the ^HH output file is saved to the USB thumb drive.

```
! U1 setvar "usb.host.config_info_to_usb" "yes"
```

usb.host.fn_field_data

This command is used to collect a user's response to a presented ^FN prompt.

IMPORTANT: This command is used only in the context of the on-printer Print Station application. Altering the use of this command in the WML can make the Print Station application non-functional. It is recommended changes to portions of this portion of the menu system be done by Zebra Professional Services team.

Setvar

To collect a user's response to a presented ^FN prompt:

```
! U1 setvar "usb.host.fn_field_data" "value"
```

Values

The text entered by the user via a USB Human Interface Device (HID) keyboard.

Default

`"NULL"`

Getvar

To retrieve the current setting value:

```
! U1 getvar "usb.host.fn_field_data"
```

Result

Result

One of the following:

- `"NULL"`
- The user response to an ^FN prompt

usb.host.fn_last_field

This command is intended for use by WML code to control the presentation of ^FN fields contained within a template file that resides on the E:/ drive and has an extension of .ZPL.

Getvar

To return the current setting value:

```
! U1 getvar "usb.host.fn_last_field"
```

Result

- "YES"
- "NO"

usb.host.hid_count

This command displays the number of USB Human Interface Devices (HIDs) connected to the printer.

Refer to http://www.usb.org for further details on USB device types.

Getvar

To retrieve the number of USB HIDs connected to the printer:

```
! U1 getvar "usb.host.hid_count"
```

Result

When no devices are attached:

```
! U1 getvar "usb.host.hid_count"
"0"
```

usb.host.keyboard_input

This command enables/disables USB keyboard input to the printer, affecting the Print Station user menu item.

Setvar

To enable or disable USB keyboard input to the printer:

```
! U1 setvar "usb.host.keyboard_input"
```

Values

- `"on"` means keyboard input is supported.
- `"off"` means keyboard input is not supported

Default

`"on"`

Getvar

To retrieve the current setting value:

```
! U1 getvar "usb.host.keyboard_input"
```

usb.host.lock_out

This command enables/disables the USB host port.

Setvar

To enable or disable the USB port:

```
! U1 setvar "usb.host.lock_out" "value"
```

Values

- `"on"` disables the USB host port.
- `"off"` enables the USB host port.

Default

`"off"`

Getvar

To return the current setting value:

```
! U1 getvar "usb.host.lock_out"
```

usb.host.mass_storage_count

This command displays the number of USB mass storage devices connected to the printer. Only the first USB mass storage device found will be mounted and accessible.

Getvar

To retrieve the current setting value:

```
! U1 getvar "usb.host.mass_storage_count"
```

usb.host.read_list

This command scans the mounted usb mass storage device for non-user-restricted files and displays the files in a list.

Setvar

To scan the mounted usb mass storage device for non-user-restricted files and displays the files in a list:

```
! U1 setvar "usb.host.read_list" "value"
```

Values

- `"fill_store"` The printer creates a list of all of the top-level files that are readable on a connected mass storage device. (`.ZPL` and `.XML` files)
- `"store"` The printer copies the current file (the file listed on the display) or all eligible files if `SELECT ALL` is displayed from the USB mass storage device to the printer's `E:` drive.

NOTE: `"store"` does an exact byte-for-byte copy of the file being stored. ZPL files that are transferred to the printer using the "store" command are NOT prepared to be used with a recall format command (^XF). See XREF Character Substitution (in Mirror) for the modifications necessary to prepare a ZPL file for use with ^XF. If the ZPL file contains a ^DF, the appropriate way to place it on the printer's memory is to use the `"print"` command, below.

- `"fill_print"` The printer analyzes the content of the USB mass storage device and creates a list of files that may be printed. (`.ZPL` files only)
- `"print"` The printer prints the current file (the file listed on the display) or all eligible files if `SELECT ALL` is displayed
- `"clear"`

Getvar

To return the current setting value:

```
! U1 getvar "usbusb.host.read_list"
```

Result

The current file in the list or `"NONE"`.

Example

```
! U1 getvar "usbusb.host.read_list"
```

One of the following:

- The current file in the list.
- `"NONE"` if there are no files on the USB mass storage device.

usb.host.read_list_print_delay

This command specifies a number of milliseconds for the printer to wait before processing the next file when “SELECT ALL” is chosen on the USB PRINT FILES user menu.

Setvar

To specify the print delay time in milliseconds:

```
! U1 setvar "usb.host.read_list_print_delay"
```

Values

"0" to "65535"

Default

"0"

Getvar

To return the current setting value:

```
! U1 getvar "usb.host.read_list_print_delay"
```

usb.host.template_list

This command is used by WML to support selecting .ZPL template files resident on the E:/ drive.

Setvar

To list the .ZPL files from the E drive:

```
! U1 getvar "usb.host.template_list"
```

Values

- "fill" The printer analyzes the contents of its E:/ drive and creates a list of the template files with a .ZPL extension.
- "up" moves to the previous file in the list
- "down" moves to the next file in the list

Default

"on"

Getvar

To return the current setting value:

```
! U1 getvar "usb.host.template_list"
```

Result

One of the following:

- "UNINITIALIZED"
- "IN PROGRESS"
- "NONE" if no .ZPL files reside on the E:/ drive
- the current .ZPL file

usb.host.template_print_amount

This command is used by WML to control the number of template-type labels to print.

Setvar

To control the number of template-type labels to print:

```
! U1 setvar "usb.host.template_print_amount" "value"
```

Values

- "1" to "99999"
- "up" moves to the previous file in the list
- "down" moves to the next file in the list

Default

"1" which will change based on the contents of the selected template file

Getvar

To retrieve the current setting value:

```
! U1 getvar "usb.host.template_print_amount"
```

Result

The current value.

usb.host.write_list

This command scans the E: flash drive for non-user-restricted files and displays the files in a list.

Setvar

To scan the E: flash drive for non-user-restricted files and displays the files in a list:

```
! U1 setvar "usb.host.write_list" "value"
```

Values

- "fill_store" The printer analyzes the contents of its E: drive and creates a list of files that may be copied to a USB mass storage device connected to the printer.
- "store" The printer copies the current file (the file listed on the display) or all eligible files if "SELECT ALL" is displayed to the USB mass storage device.
- "up"
- "down"

Default

"on"

Getvar

To return the current setting value:

```
! U1 getvar "usb.host.write_list"
```

Result

One of the following:

The current file in the list.

- "NONE" if there are no files on the E: drive of the printer.
- "READONLY" if the USB mass storage device is read-only.

usb.mirror.appl_path

This command specifies the path to the location on a USB device from which Mirror files are retrieved. If no path is specified, then the path is `zebra/appl`.

Setvar

To specify the path to the location on a USB device from which Mirror files are retrieved:

```
! U1 setvar "usb.mirror.appl_path" "path"
```

Values

A valid path up to 255 characters.

Default

`"zebra/appl"`

Getvar

To retrieve the current setting value:

```
! U1 getvar "usb.mirror.appl_path"
```

usb.mirror.auto

This command determines if mirroring happens automatically when a USB device is inserted in the printer. To use this function, the setting for `usb.mirror.enable` must be `"on"`.

Setvar

To set the automatic mirroring of a USB device:

```
! U1 setvar "usb.mirror.auto" "value"
```

Values

- `"on"` Mirroring occurs automatically when the USB device is inserted and `usb.mirror.enable` is on.
- `"off"` Mirroring does not occur automatically when the USB device is inserted.
- `"prompt"` The printer gives you the option to start or abort a mirroring process.

Getvar

To return the current setting value:

```
! U1 getvar "usb.mirror.auto"
```

usb.mirror.enable

This command enables or disables the ability to perform mirroring of a USB device.

Setvar

To enable or disable the USB device mirroring ability:

```
! U1 setvar "usb.mirror.enable" "value"
```

Values

- `"on"` Mirroring is enabled
- `"off"` Mirroring is disabled

Default

`"on"`

Getvar

To retrieve the current setting value:

```
! U1 getvar "usb.mirror.enable"
```

usb.mirror.enabled

Enables or disables the ability to perform mirroring using a USB device memory stick.

Setvar

To set the command:

```
! U1 setvar "usb.mirror.enabled" "value"
```

Values

- `"on"` Mirroring is enabled
- `"off"` Mirroring is disabled

Default

`"on"`

Getvar

To have the printer return the current setting value:

```
! U1 getvar "usb.mirror.enabled"
```

usb.mirror.error_retry

This command specifies the number of times that the USB mirror operation will be repeated if the process fails.

Setvar

To specify the number of times that the USB mirror operation will be repeated:

```
! U1 setvar "usb.mirror.error_retry" "value"
```

Values

"0" to "65535"

Default

"0"

Getvar

To retrieve the current setting value:

```
! U1 getvar "usb.mirror.error_retry"
```

usb.mirror.feedback.auto

This command determines if a feedback file is written to the USB device connected to the printer after mirroring.

Setvar

To specify if the feedback file is written to the USB device or not:

```
! U1 setvar "usb.mirror.feedback.auto" "value"
```

Values

- `"on"` A feedback file is written to the USB device connected to the printer.
- `"off"` A feeback file is not written to the USB device connected to the printer.

Getvar

To return the current setting value:

```
! U1 getvar "usb.mirror.feedback.auto"
```

usb.mirror.feedback.odometer

This command instructs the printer to set or retrieve the `usb.mirror.feedback.odometer` value.

Setvar

To set the `usb.mirror.feedback.odometer` value:

```
! U1 setvar "usb.mirror.feedback.odometer" "value"
```

Values

`"0"` to `"65535"`

Default

`"0"`

Example

This example sets the counter to 0, which resets the counter.

```
! U1 setvar "usb.mirror.feedback.odometer" "0"
```

Getvar

To retrieve the `usb.mirror.feedback.odometer` value:

```
! U1 getvar "usb.mirror.feedback.odometer"
```

usb.mirror.feedback.path

This command specifies where the feedback file is stored on the USB drive connected to the printer.

Setvar

To specify the path of the feedback file stored on the USB drive:

```
! U1 setvar "usb.mirror.feedback.path" "value"
```

Values

A valid path up to 255 characters

NOTE: The path must exist on the USB drive before the printer can write files to it.

Default

`"zebra/feedback"`

Getvar

To return the current setting value:

```
! U1 getvar "usb.mirror.feedback.path"
```

usb.mirror.fetch

This command immediately initiates a USB mirroring operation. To use this function, the setting for `usb.mirror.enable` must be `"on"`.

Do

To initiate a USB mirroring operation:

```
! U1 do "usb.mirror.fetch"
```

usb.mirror.last_error

This command returns the last USB mirroring error string. If there are no errors, the result is `"No Error"`.

Getvar

To return the last USB mirroring error string:

```
! U1 getvar "usb.mirror.last_error"
```

usb.mirror.last_time

This command returns the date and time of the last USB mirroring operation. The date and time reported are from the printer's clock.

Getvar

To return the date and time of the last USB mirroring operation:

```
! U1 getvar "usb.mirror.last_time"
```

Result

```
02 18 2015 11:10:09
```

In the format mm dd yyyy hh:mm:ss

usb.mirror.path

This command specifies the root directory on the connected USB drive. The root directory contains the subdirectories from which files are retrieved during the mirror operation. If no path is specified, then the path is `/zebra`.

The standard subdirectores to use are:

- `"<update-root>"/appl`
- `"<update-root>"/files`
- `"<update-root>"`
- `"<feedback-root>"`

See How Mirror Works on page 1680 for a comprehensive overview of mirroring.

Setvar

To specify the root directory on the connected USB drive:

```
! U1 setvar "usb.mirror.appl_path" "path"
```

Values

A valid path up to 255 characters.

Default

`"zebra"`

Getvar

To retrieve the current setting:

```
! U1 getvar "usb.mirror.path"
```

usb.mirror.reset_delay

This command specifies a number of seconds for the printer to wait after it loads the last of the files in the `/commands` directory during mirroring.

Setvar

To specify the USB mirror reset delay time:

```
! U1 setvar "usb.mirror.reset_delay"
```

Values

`"0"` to `"900"`

Default

`"5"`

Getvar

To return the current setting value:

```
! U1 getvar "usb.mirror.reset_delay"
```

usb.mirror.success

This command returns the last USB mirroring event status string.

Getvar

To return the last USB mirroring event status string:

```
! U1 getvar "usb.mirror.success"
```

Values

- `"yes"` Indicates that the mirroring operation was successful.
- `"no"` Indicates that the mirroring operation was not successful.

usb.mirror.success_time

This command returns the date and time of the last successful USB mirroring operation. The date and time reported are from the printer's clock.

Getvar

To return the date and time of the last successful USB mirroring operation:

```
! U1 getvar "usb.mirror.success_time.fm"
```

Result

```
02 18 2015 11:10:09
```

In mm dd yyyy hh:mm:ss format.

zbi.control.add_breakpoint

This command instructs the printer to set a ZBI program break point.

Setvar

To instruct the printer to set a ZBI program break point:

```
! U1 setvar "zbi.control.add_breakpoint" "value"
```

Values

Any line number of the program currently being debugged.

Example

This `setvar` example shows setting the breakpoint at line `"30"`.

```
! U1 setvar "zbi.control.add_breakpoint" "30"
```

zbi.control.break

This command breaks the execution of the ZBI 2.0 program that is currently running.

Setvar

To break the execution of the ZBI program that is currently running:

```
! U1 setvar "zbi.control.break" ""
```

Values

""

Example

This `setvar` example shows the value set to "".

```
! U1 setvar "zbi.control.break" ""
```

zbi.control.clear_breakpoints

This command deletes all breakpoints in the current ZBI 2.0 program.

Setvar

To instruct the printer to delete all breakpoints:

```
! U1 setvar "zbi.control.clear_breakpoints" ""
```

Values

""

Example

This `setvar` example shows the value set to "".

```
! U1 setvar "zbi.control.clear_breakpoints" ""
```

zbi.control.delete_breakpoint

This command deletes a breakpoint in the current ZBI 2.0 program.

Setvar

To instruct the printer to delete the breakpoint at the line indicated by the value parameter:

```
! U1 setvar "zbi.control.delete_breakpoint" "value"
```

Values

You can use the same value as `add_breakpoint`.

Example

This `setvar` example shows the breakpoint set to `"30"`.

```
! U1 setvar "zbi.control.delete_breakpoint" "30"
```

zbi.control.line_number

This command gives you control and information about which line of a stopped ZBI 2.0 program is being executed.

Setvar

To sets which line of the current ZBI 2.0 program should be executed:

```
! U1 setvar "zbi.control.line_number" "value"
```

Values

Any line number of the currently stopped ZBI program.

Default

`"0"`

Getvar

To return the line number that is currently being executed in the ZBI 2.0 program:

```
! U1 getvar "zbi.control.line_number"
```

Example

This `setvar` example shows the value parameter set to `"30"`.

```
! U1 setvar "zbi.control.line_number" "30"
```

When the `setvar` value is set to `"30"`, the `getvar` result is `"30"`.

zbi.control.restart

This command restarts a ZBI 2.0 program that is currently stopped.

Setvar

To restart a ZBI 2.0 program that is currently stopped:

```
! U1 setvar "zbi.control.restart" "value"
```

Values

""

Example

This `setvar` example shows the value set to "".

```
! U1 setvar "zbi.control.restart" ""
```

zbi.control.run

This command runs the current ZBI 2.0 program that is loaded in the interpreter.

Setvar

To run the ZBI 2.0 program that is loaded in the interpreter:

```
! U1 setvar "zbi.control.run" ""
```

Values

""

Example

This `setvar` example shows the value set to "".

```
! U1 setvar "zbi.control.run" ""
```

zbi.control.step

This command restarts the execution of the currently stopped ZBI 2.0 program for one line.

Setvar

To instruct the printer to restart the execution of the currently stopped ZBI 2.0 program for one line:

```
! U1 setvar "zbi.control.step" ""
```

Values

""

Default

""

Example

This `setvar` example shows the value set to "".

```
! U1 setvar "zbi.control.step" ""
```

When the `setvar` value is set to "".

zbi.control.terminate

This command instructs the ZBI 2.0 program to terminate and shuts down the interpreter.

Setvar

To instruct the ZBI 2.0 program to terminate and shuts down the interpreter:

```
! U1 setvar "zbi.control.terminate" "value"
```

Values

""

Example

This `setvar` example shows the value set to "".

```
! U1 setvar "zbi.control.terminate" ""
```

zbi.control.variable_name

This command sets the name of the variable that is to be read or modified through `variable_value`.

Setvar

To set the variable that is to show on the front panel:

```
! U1 setvar "zbi.control.variable_name" "value"
```

Values

Any ZBI variable in the program that is currently being debugged.

Default

""

Getvar

To retrieve the variable value that is to show on the front panel:

```
! U1 getvar "zbi.control.variable_name"
```

Example

This `setvar` example shows the value set to `"MYVAR$"`.

```
! U1 setvar "zbi.control.variable_name" "MYVAR$"
```

When the `setvar` value is set to `"MYVAR$"`, the `getvar` result is `"MYVAR$"`.

zbi.control.variable_value

This command identifies the variable name.

Setvar

To set a value to the variable referenced by a variable_name:

```
! U1 setvar "zbi.control.variable_value" "value"
```

Values

A string or integer that is dependent on the variable type in `variable_name`.

Default

The current value of the variable referenced by variable_name.

Getvar

To retrieve the variable name that is loaded into the variable_name:

```
! U1 getvar "zbi.control.variable_value"
```

Example

This `setvar` example shows the value set to `"Hello World"`.

```
! U1 setvar "zbi.control.variable_value" "Hello World"
```

When the `setvar` value is set to `"Hello World"`, the `getvar` result is `"Hello World"`.

zbi.enable

This command enables ZBI on the printer.

Setvar

To set the command:

```
! U1 setvar "zbi.enable" "value"
```

Values

- `"on"`
- `"off"`

Default

`"on"`

Getvar

To have the printer return the current setting value:

```
! U1 getvar "zbi.enable"
```

zbi.key

This command identifies if the ZBI 2.0 option is enabled or disabled on the printer.

Getvar

To retrieve the status of the ZBI 2.0 option on the printer:

```
! U1 getvar "zbi.key"
```

Example

In this example, the `getvar` shows the status of ZBI on the printer.

```
! U1 getvar "zbi.key"
"ENABLED"
```

zbi.last_error

This command identifies the last error that the ZBI 2.0 interpreter encountered.

Getvar

To show the last error that the ZBI 2.0 interpreter encountered:

```
! U1 getvar "zbi.last_error"
```

Example

This example demonstrates how to make the ZBI 2.0 interpreter return the last error it encountered.

```
! U1 getvar "zbi.last_error"
```

zbi.program_list

This SGD is intended to be used on WML pages. It purpose is to support displaying the name of each ZBI program file contained on the E: drive.

IMPORTANT: A specific sequence required to get an accurate listing of files. See the example below.

Setvar

To display the progam list:

```
! U1 setvar "zbi.program_list" "value"
```

Values

- `"fill"` initializes the device (first step)
- `"up"` gets the previous file in the list
- `"down"` gets the next file in the list
- `"execute"` executes the currently specified zbi file (as determined by the `getvar` command)

Default

- `"none"`

Example

A specific sequence of commands is required to get an accurate listing of files.

Initialize the device by issuing:

```
! U1 setvar "zbi.program_list" "fill"
```

To get the first file, issue:

```
! U1 getvar "zbi.program_list"
```

This will return the current filename in the list of zbi files present on the E: drive. If the response returned is `"none"` you have reached the end of the list.

To get the next filename in the list, issue:

```
! U1 setvar "zbi.program_list" "up"
```

To get the previous filename in the list, issue:

```
! U1 setvar "zbi.program_list" "down"
```

To execute the current zbi filename, i.e. the one returned by a getvar, issue:

```
! U1 setvar "zbi.program_list" "execute"
```

Getvar

To return the current setting value:

```
! U1 getvar "zbi.program_list"
```

zbi.reseller_key

This command allows programs that are encrypted with this key in ZBI- Developer to run.

Setvar

To allow programs that are encrypted with this key in ZBI developer to run:

```
! U1 setvar "zbi.reseller_key" "value"
```

Values

Any valid encryption key provided by ZBI Developer.

Example

This `setvar` example shows the value set to `"abc123".`

```
! U1 setvar "zbi.reseller_key" "abc123".
```

zbi.revision

This command identifies the current ZBI version.

Getvar

To retrieve the current ZBI version:

```
! U1 getvar "zbi.revision"
```

Example

In this example, the `getvar` shows the current ZBI version.

```
! U1 getvar "zbi.revision"
```

zbi.running_program_name

This command identifies the name of the ZBI 2.0 program that is currently running.

Getvar

To retrieve the name of the currently running ZBI 2.0 program:

```
! U1 getvar "zbi.running_program_name"
```

Example

In this example, the `getvar` command causes the printer to respond that the program `choices.bas` is currently running.

```
! U1 getvar "zbi.running_program_name"
"CHOICES.BAS"
```

zbi.start_info.execute

This command instructs the ZBI 2.0 environment to execute the program listed in the file_name.

Setvar

To instruct the ZBI 2.0 environment to execute the program listed in the file_name:

```
! U1 setvar "zbi.start_info.execute"
```

Example

This `setvar` example executes the `"choices.bas"` program.

```
! U1 setvar "zbi.start_info.execute" "choices.bas"
```

zbi.start_info.file_name

This command prepares a program to run when the `zbi.start_info.execute` command is used. This command does not run the program.

Setvar

To prepare a ZBI 2.0 program to be executed using the `zbi.start_info.execute` command:

```
! U1 setvar "zbi.start_info.file_name" "value"
```

Values

A file name or path of a basic program

Getvar

To return the file path and file name of a ZBI 2.0 program to run using the `zbi.start_info.execute` command:

```
! U1 getvar "zbi.start_info.file_name"
```

Default

The last program run. If nothing has been run, `"*:\.BAZ"`.

Example

This `setvar` example shows the value set to `"E:PROGRAM1.BAS"`.

```
! U1 setvar "zbi.start_info.file_name" "E:PROGRAM1.BAS"
```

When the `setvar` value is set to `"E:PROGRAM1.BAS"`, the `getvar` result is `"E:PROGRAM1.BAS"`.

zbi.start_info.memory_alloc

This command identifies the amount of memory currently in use in a ZBI 2.0 program.

Getvar

To show the amount of memory currently in use in bytes:

```
! U1 getvar "zbi.start_info.memory_alloc"
```

Example

In this example, the `getvar` shows the amount of memory currently in use.

```
! U1 getvar "zbi.start_info.memory_alloc"
"17203"
```

zbi.state

This command shows the current state of the ZBI 2.0 program.

Getvar

To retrieve the current state of ZBI:

```
! U1 getvar "zbi.state"
```

Values

- `"running"` ZBI Interpreter is active and running a program
- `"off"` ZBI Interpreter is inactive
- `"stopped"` ZBI Interpreter is active but not executing a program

Example

In this example, the `getvar` shows that state of ZBI.

```
! U1 getvar "zbi.state"
"running"
```

zpl.calibrate

This command measures the media label length that is installed in the printer. It then sets the "zpl.label_length" parameter to the detected media label length.

Do

To calibrate to the media installed in the printer:

```
! U1 do "zpl.calibrate" ""
```

Values
NA

zpl.format_prefix

This command sets or returns the format character to the character corresponding to the ASCII Hex value xx. This command is similar to the ^CC and ~CC commands.

Setvar

To set the format character to the character corresponding to the ASCII Hex value xx:

```
! U1 setvar "zpl.format_prefix" "xx"
```

Values

"xx" = "00-FF", "00-ff", "up","down"

- Specifying the up or down will increment or decrement the current value.
- Note that the increment/decrement will skip over conflicting values.
- Conflicting values for zpl.format_prefix are zpl.command_prefix, and zpl.delimiter.

Getvar

To return the current value of the format character:

```
! U1 getvar "zpl.format_prefix"
```

Result

^ (5E)

zpl.caret

This command is used to change the format command prefix for ZPL commands. The default prefix is the caret (^)

This command is equivalent to the ~CC and ^CC ZPL commands.

Setvar

To set the command prefix:

```
! U1 setvar "zpl.caret" "value"
```

Values
Any ASCII character

Default

"^"

Getvar

To retrieve the current format command prefix:

```
! U1 getvar "zpl.caret"
```

Example

This `setvar` example changes the format prefix to a forward slash `"/"`

```
! U1 setvar "zpl.caret" "/"
```

zpl.control_character

This command is used to change the control command prefix. The default prefix is the tilde (~).

This command is equivalent to the ^CT and ~CT commands.

Setvar

To change the control character:

```
! U1 setvar "zpl.control_character" "value"
```

Values

ASCII values for the desired character "00-FF,00-ff,up,down"

See ASCII on page 1555 for more information.

Default

"7E" (tilde)

Getvar

To return the currently set control character:

```
! U1 getvar "zpl.control_character"
```

Example

This example sets the value set to "+".

```
! U1 setvar "zpl.control_characater" "2b"
```

zpl.delimiter

This command is used to change the delimiter character for ZPL commands. The default delimiter character is the comma (,).

Related Commands: ^CD and ~CD commands.

Setvar

To change the delimiter character for ZPL commands:

```
! U1 setvar "zpl.delimiter" "value"
```

Values

HEX values for the desired character "00-FF,00-ff,up,down"

See ASCII on page 1555 for more information.

Default

"2C" (comma)

Getvar

To return the currently set delimiter character:

```
! U1 getvar "zpl.delimiter"
```

Example

This example changes the delimiter to a semi-colon (;).

```
! U1 setvar "zpl.delimiter" "3B"
```

zpl.label_length

This command defines the length of the label. This command is necessary when using continuous media (media that is not divided into separate labels by gaps, spaces, notches, slots, or holes).

This command is equivalent to the ^LL command.

Setvar

To set the label length:

```
! U1 setvar "zpl.label_length" "value"
```

Values

"1" to "32000", (in dots) not to exceed the maximum label length.

While the printer accepts any value for this parameter, the amount of memory installed determines the maximum length of the label.

Getvar

To return the current label length setting:

```
! U1 getvar "zpl.label_length"
```

Comments

These formulas can be used to determine the value of y:

For 6 dot/mm printheads...	Label length in inches x 152.4 (dots/inch) = y
For 8 dot/mm printheads...	Label length in inches x 203.2 (dots/inch) = y
For 12 dot/mm printheads...	Label length in inches x 304.8 (dots/inch) = y
For 24 dot/mm printheads...	Label length in inches x 609.6 (dots/inch) = y

Values for y depend on the memory size. If the entered value for y exceeds the acceptable limits, the bottom of the label is cut off. The label also shifts down from top to bottom.

Example

```
! U1 setvar zpl.label_length" "1281"
```

zpl.label_length_always

This command allows the label length defined by ^LL to apply when the media is Gap or Black Mark. This command is persistent across a power cycle.

See also ^LL on page 294.

Setvar

To enable or disable the label length parameter:

```
! U1 setvar "zpl.label_length_always" "value"
```

Values

- "yes" applies the label length to all media.
- "no" causes the label length to apply only to continuous media.

Getvar

To determine the setting for zpl.label_length_always:

```
! U1 getvar "zpl.label_length_always"
```

Values

- "yes" indicates that the label length applies to all media.
- "no" indicates that the label length applies only to continuous media.

Default

"no"

zpl.label_orientation

This command instructs the printer to rotate the entire label output by 0, 90, 180, or 270 degrees.

Setvar

NOTE: If this command is set to 180, `print.invert_label` is enabled, and `^POI` is used. The outcome results in an inverted label orientation (because it is triple inverting).

To assign the value of the label orientation:

```
! U1 setvar "zpl.label_orientation" "value"
```

Values

`"0"`, `"90"`, `"180"`, `"270"`

Default

`"0"`

Getvar

To retrieve the value of the label orientation:

```
! U1 getvar "zpl.label_orientation"
```

Example

This `setvar` example shows the value set to `"90"`.

```
! U1 setvar "zpl.label_orientation" "90"
```

zpl.left_position

This command sets the label's left margin offset in dots.

Related Command: ^LS

Setvar

To set the label left margin offset in dots:

```
! U1 setvar "zpl.left_position" "value"
```

Values

"-9999" to "9999"

Default

"0"

Getvar

To retrieve the currently set left margin offset for the label:

```
! U1 getvar "zpl.left_position"
```

Example

```
! U1 setvar "zpl.left_position" "100"
```

zpl.system_error

This command returns the system error flags.

This command is equivalent to the error recording of ~HQES.

Getvar

To return the state of the system error flags:

```
! U1 getvar "zpl.system_error"
```

Result

```
"0,0,00000000,00000000"
```

(flag, error flag, Group 2, Group 1)

Example

This example shows how to request the printer's status.

To request the printer's status, type `! U1 getvar "zpl.system_error"`

The printer responds with data similar to this:

```
"1,1,00000000,00000004"
```

In this example, the Printer Status resolves to these conditions:

- The printer is in Pause (value = 1)
- The Error Flag is 0 if there are no errors (i.e. Group 2 and Group 1 are all 0s), and 1 if there are any errors.
- The cover/printhead is open (value = 4).

Table 24 Error Flags (~HQES)

Error Flags	Flag	Group 2	Group 1 (X = Value can be any hexadecimal number [0-9, A-F])							
		Nibbles16-9	Nibble8	Nibble7	Nibble6	Nibble5	Nibble4	Nibble3	Nibble2	Nibble1
No Error	0	00000000	0	0	0	0	0	0	0	0
Error Present	1	00000000	X	X	X	X	X	X	X	X
Printhead Thermistor Open	1	00000000	X	X	X	X	X	2	X	X
Invalid Firmware Config.	1	00000000	X	X	X	X	X	1	X	X
Printhead Detection Error	1	00000000	X	X	X	X	X	X	8	X
Bad Printhead Element	1	00000000	X	X	X	X	X	X	4	X

Table 24 Error Flags (~HQES) (Continued)

Error Flags	Flag	Group 2	Group 1 (X = Value can be any hexadecimal number [0-9, A-F])							
		Nibbles16-9	Nibble8	Nibble7	Nibble6	Nibble5	Nibble4	Nibble3	Nibble2	Nibble1
Motor Over Temperature	1	00000000	X	X	X	X	X	X	2	X
Printhead Over Temperature	1	00000000	X	X	X	X	X	X	1	X
Cutter Fault	1	00000000	X	X	X	X	X	X	X	8
Head Open	1	00000000	X	X	X	X	X	X	X	4
Ribbon Out	1	00000000	X	X	X	X	X	X	X	2
Media Out	1	00000000	X	X	X	X	X	X	X	1
Clear Paper Path Failed[1]	1	00000000	X	X	X	X	8	X	X	X
Paper Feed Error[1]	1	00000000	X	X	X	X	4	X	X	X
Presenter Not Running[1]	1	00000000	X	X	X	X	2	X	X	X
Paper Jam during Retract[1]	1	00000000	X	X	X	X	1	X	X	X
Black Mark not Found[1]	1	00000000	X	X	X	8	X	X	X	X
Black Mark Calibrate Error[1]		00000000	X	X	X	4	X	X	X	X
Retract Function timed out[1]	1	00000000	X	X	X	2	X	X	X	X
Paused[1]	1	00000000	X	X	X	1	X	X	X	X

1. This error flag is supported only on KR403 printers.

zpl.system_status

This command returns the errors and warnings of the system.

This command is equivalent to all data reported by the ~HQES ZPL command.

Getvar

To return the system error and warning flags:

```
! U1 getvar "zpl.system_status"
```

Result

```
"0","0","00000000","00000000","0","00000000","00000000"
```

(flag, error flag, group 2, group 1, warning flag, group 2, group 1)

Example

This example shows how to request the printer's status.

To request the printer's status, type `! U1 getvar "zpl.system_status"`

The printer responds with data similar to this:

```
"1,1,00000000,00000004,0,00000000,00000000"
```

In this example, the Printer Status resolves to these conditions:

- The printer is in Pause (value = 1)
- The Error Flag is 0 if there are no errors (i.e. Group 2 and Group 1 are all 0s), and 1 if there are any errors (non-zero).
- The cover/printhead is open (value = 4).
- The Warning Flag is 0 if there are no warnings (i.e. Group 2 and Group 1 are all 0s), and 1 if there are any errors (non-zero).

Table 25 Error Flags (~HQES)

Error Flags	Flag	Group 2	Group 1 (X = Value can be any hexadecimal number [0-9, A-F])							
		Nibbles16-9	Nibble8	Nibble7	Nibble6	Nibble5	Nibble4	Nibble3	Nibble2	Nibble1
No Error	0	00000000	0	0	0	0	0	0	0	0
Error Present	1	00000000	X	X	X	X	X	X	X	X
Printhead Thermistor Open	1	00000000	X	X	X	X	X	2	X	X
Invalid Firmware Config.	1	00000000	X	X	X	X	X	1	X	X
Printhead Detection Error	1	00000000	X	X	X	X	X	X	8	X
Bad Printhead Element	1	00000000	X	X	X	X	X	X	4	X

Table 25 Error Flags (~HQES) (Continued)

Error Flags	Flag	Group 2	Group 1 (X = Value can be any hexadecimal number [0-9, A-F])							
		Nibbles16-9	Nibble8	Nibble7	Nibble6	Nibble5	Nibble4	Nibble3	Nibble2	Nibble1
Motor Over Temperature	1	00000000	X	X	X	X	X	X	2	X
Printhead Over Temperature	1	00000000	X	X	X	X	X	X	1	X
Cutter Fault	1	00000000	X	X	X	X	X	X	X	8
Head Open	1	00000000	X	X	X	X	X	X	X	4
Ribbon Out	1	00000000	X	X	X	X	X	X	X	2
Media Out	1	00000000	X	X	X	X	X	X	X	1
Clear Paper Path Failed[1]	1	00000000	X	X	X	X	8	X	X	X
Paper Feed Error[1]	1	00000000	X	X	X	X	4	X	X	X
Presenter Not Running[1]	1	00000000	X	X	X	X	2	X	X	X
Paper Jam during Retract[1]	1	00000000	X	X	X	X	1	X	X	X
Black Mark not Found[1]	1	00000000	X	X	X	8	X	X	X	X
Black Mark Calibrate Error[1]	1	00000000	X	X	X	4	X	X	X	X
Retract Function timed out[1]	1	00000000	X	X	X	2	X	X	X	X
Paused[1]	1	00000000	X	X	X	1	X	X	X	X

1. This error flag is only supported on KR403 printers.

Table 26 Warning Flags (~HQES)

Warning Flags	Flag	Group 2	Group 1 (X = Value can be any hexadecimal number [0-9, A-F])							
		Nibbles16-9	Nibble8	Nibble7	Nibble6	Nibble5	Nibble4	Nibble3	Nibble2	Nibble1
No Warning	0	00000000	0	0	0	0	0	0	0	0
Warning Present	1	00000000	X	X	X	X	X	X	X	X
Paper-near-end Sensor[1]	1	00000000	X	X	X	X	X	X	X	8
Replace Printhead	1	00000000	X	X	X	X	X	X	X	4
Clean Printhead	1	00000000	X	X	X	X	X	X	X	2
Need to Calibrate Media	1	00000000	X	X	X	X	X	X	X	1

Table 26 Warning Flags (~HQES) (Continued)

Warning Flags	Flag	Group 2	Group 1 (X = Value can be any hexadecimal number [0-9, A-F])							
		Nibbles16-9	Nibble8	Nibble7	Nibble6	Nibble5	Nibble4	Nibble3	Nibble2	Nibble1
Sensor 1 (Paper before head)[1]	1	00000000	X	X	X	X	X	X	1	X
Sensor 2 (Black mark)[1]	1	00000000	X	X	X	X	X	X	2	X
Sensor 3 (Paper after head)[1]	1	00000000	X	X	X	X	X	X	4	X
Sensor 4 (loop ready)[1]	1	00000000	X	X	X	X	X	X	8	X
Sensor 5 (presenter)[1]	1	00000000	X	X	X	X	X	1	X	X
Sensor 6 (retract ready)[1]	1	00000000	X	X	X	X	X	2	X	X
Sensor 7 (in retract)[1]	1	00000000	X	X	X	X	X	4	X	X
Sensor 8 (at bin)[1]	1	00000000	X	X	X	X	X	8	X	X

1. This warning flag is only supported on KR403 printers.

zpl.zpl_mode

This command sets the ZPL mode to ZPL II or ZPL.

Setvar

To set the printer ZPL mode:

```
! U1 setvar "zpl.zpl_mode" "value"
```

Values

- `"zpl"`
- `"zpl II"`

Default

`"zpl II"`

Getvar

To return the current ZPL mode setting:

```
! U1 getvar "zpl.zpl_mode"
```

Example

This `setvar` example sets the ZPL mode to ZPL.

```
! U1 setvar "zpl.system_status" "zpl"
```

zpl.zpl_override

Enable this menu item to prevent the following ZPL commands from changing the printer's current settings:

- ^MM (print mode)
- ^MT (Direct Thermal or Thermal Transfer print method)
- ^MN (media type - non-continuous or continuous)

When this menu item is disabled, these commands override the printer's settings.

Setvar

To set the override status to the specified value:

```
! U1 setvar "zpl.zpl_override" "value"
```

Values

- "disabled" allows override
- "enabled" prevents ZPL commands from overriding printer settings.

Default

"disabled"

Example

This example enables zpl.zpl_override, which prevents ^MM, ^MT, and ^MN from making changes to the current printer settings.

```
! U1 setvar "zpl.zpl_override" "enabled"
```

zpl.relative_darkness

Changes the relative darkness for ZPL labels. This command is similar to the ZPL ^MD command.

IMPORTANT: This value is saved permanently on Desktop printers, but it is not saved permanently on Industrial or Mobile.

Setvar

To set the relative darkness for ZPL labels:

```
! U1 setvar "zpl.relative_darkness" "value"
```

Values

"" to "300"

Default

"0"

Getvar

To return the current setting value:

```
! U1 getvar "relative_darkness"
```

Example

These examples show setting the printer to different darkness levels:

- If the current value shown on the configuration label is "16", using a zpl.relative_darkness command of "-90" decreases the value to "7.0".
- If the current value shown on the configuration label is "1", using a zpl.relative_darkness command of "153" increases the value to "16.3".
- If the current value shown on the configuration label is "25", using a zpl.relative_darkness command of "105" increases the value to "30.0", which is the maximum value allowed.

Each zpl.relative_darkness command is treated separately in relation to the current value as printed on the configuration label.

SGD Network Commands

This section provides a high-level overview of the network Set / Get / Do (SGD) commands.

.15†

Wireless SGD commands are available in printers with the following firmware versions or later:

- V60.16.2Z or later
- V60.15.xZ or later
- V50.15.xZ or later
- V61.15.xZ or later
- V56.15.xZ or later
- V53.16.x or later
- V53.15.2Z or later
- R53.16.3Z or later
- R60.15.8Z or later
- R62.15.8Z or later
- R63.15.8Z or later
- R65.15.8Z or later

NOTE: The wireless commands listed in this chapter are for use with the Wireless Print Server and Wireless Plus Print Server, when used with firmware version V60.15.x, V50.15.x, or later.

Wired SGD commands are available in printers with the following firmware versions or later:

- V54
- V66
- V68
- V72
- V73
- V74
- V75
- V76
- V78
- V79
- V60.16.2Z or later
- V60.15.xZ or later
- V50.15.xZ or later
- V61.15.xZ or later
- V56.15.xZ or later
- V53.16.x or later
- V53.15.2Z or later
- R53.16.3Z or later
- R60.15.8Z or later
- R62.15.8Z or later
- R63.15.8Z or later
- R65.15.8Z or later

bluetooth.address

This command returns the printer's Bluetooth device address.

Getvar

To retrieve the printer's Bluetooth address:

```
! U1 getvar "bluetooth.address"
```

Example

In this example, the `getvar` command causes the printer to return the printer's Bluetooth address.

```
! U1 getvar "bluetooth.address"
```

bluetooth.afh_map

Sets or retrieves the default AFH (adaptive frequency hopping) channel map (Bluetooth radios 1.2 and later); 20 bytes.

Setvar

To selectively enables or disables individual Bluetooth channels for use when AFH mode is set to `"on"`:

```
! U1 setvar "bluetooth.afh_map" "value"
```

Values

20-byte string of hexadecimal characters

Default

`"7FFFFFFFFFFFFFFFFFFF"`

Getvar

To retrieve default AFH channel map:

```
! U1 getvar "bluetooth.afh_map"
```

Example

This `setvar` example shows the value set to `"7FFFFFFFFFFFFFFFFFFF"`.

```
! U1 setvar "bluetooth.afh_map" "7FFFFFFFFFFFFFFFFFFF"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"7FFFFFFFFFFFFFFFFFFF"`.

bluetooth.afh_map_curr

This command retrieves the current AFH (adaptive frequency hopping) channel map (Bluetooth radios 1.2 and later).

Getvar

To retrieve the current AFH channel map:

```
! U1 getvar "bluetooth.afh_map_curr"
```

Example

In this example, the `getvar` command causes the printer to retrieve the current AFH channel map.

```
! U1 getvar "bluetooth.afh_map_curr"
```

bluetooth.afh_mode

This command sets or retrieves AFH (adaptive frequency hopping) mode setting (Bluetooth radios 1.2 and later).

Setvar

To enable or disable AFH mode:

```
! U1 setvar "bluetooth.afh_mode" "value"
```

Values

- `"on"`enables AFH mode
- `"off"`disables AFH mode

Default

`"off"`

Getvar

To retrieve the current setting of the AFH mode:

```
! U1 getvar "bluetooth.afh_mode"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "bluetooth.afh_mode" "on"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"on"`.

bluetooth.allow_non_display_numeric_comparison

This command allows the user to enable or disable a numeric comparison connection for printers without a display and also enables or disables displaying the pass key.

NOTE: This is an unsupported command for printers with display.

Setvar

To set the numeric comparison connection:

```
! U1 setvar "device.allow_non_display_numeric_comparison" "value"
```

Values

`"off"`	Does not allow pass key connection for non display printers.
`"print"`	Prints and accepts the pass key on non display printers (Man in the Middle protection).
`"no print"`	Accepts but does not print the pass key for non display printer (no Man in the Middle protection).

Default

`"print"`

Getvar

To return the current setting value:

```
! U1 getvar "device.allow_non_display_numeric_comparison"
```

bluetooth.authentication

This command sets or retrieves Bluetooth authentication mode and works in combination with the `bluetooth.bluetooth_pin`.

Setvar

To enable or disable Bluetooth authentication:

```
! U1 setvar "bluetooth.authentication" "value"
```

Values

- `"off"` disables authentication (can connect to master device without PIN)
- `"setpin"` enables authentication (requires PIN or passkey to connect to a master device)

Default

`"off"`

Getvar

To retrieve the current Bluetooth authentication mode:

```
! U1 getvar "bluetooth.authentication"
```

Example

This `setvar` example shows the value set to `"setpin"`.

```
! U1 setvar "bluetooth.authentication" "setpin"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"setpin"`.

bluetooth.bluetooth_pin

This command is used to connect to the printer only when the command `bluetooth.authentication` is set to `"setpin"`.

Setvar

To set the Bluetooth pin value:

```
! U1 setvar "bluetooth.bluetooth_pin" "value"
```

Values

Any text string up to 10 characters

Default

`""`

Getvar

To retrieves the current Bluetooth pin:

```
! U1 getvar "bluetooth.bluetooth_pin"
```

Example

This `setvar` example shows the value set to `"1234567890"`.

```
! U1 setvar "bluetooth.bluetooth_pin" "1234567890"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"1234567890"`.

bluetooth.clear_bonding_cache

Deletes all information related to previous Bluetooth pairing events from the printer.

Do

To delete all information related to previous Bluetooth pairing events from the printer:

```
! U1 do "bluetooth.clear_bonding_cache"
```

Values

NA

Default

NA

bluetooth.date

This command shows the release date of the Bluetooth module.

Getvar

To show the release date of the Bluetooth module:

```
! U1 getvar "bluetooth.date"
```

Example

In this example, the `getvar` command returns the release date of the Bluetooth module.

```
! U1 getvar "bluetooth.date"
```

bluetooth.discoverable

This command enables or disables the Bluetooth discoverable mode.

Setvar

To enables or disables the Bluetooth discoverable mode:

```
! U1 setvar "bluetooth.discoverable" "value"
```

Values

`"on"` enables Bluetooth discoverable mode

`"off"` disables Bluetooth discoverable mode

Default

`"on"` for Printers running Link-OS v5.3 or earlier versions

`"off"` for Printers running Link-OS 6 or later versions)

Getvar

To retrieve the current Bluetooth discoverable mode:

```
! U1 getvar "bluetooth.discoverable"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "bluetooth.discoverable" "on"
```

What the setvar value is set to is the `getvar` result. In this example, the getvar result is "on".

bluetooth.enable

This command enables or disables the Bluetooth radio.

Setvar

To enable or disable the Bluetooth radio:

```
! U1 setvar "bluetooth.enable"
```

Values

- `"on"` enables the Bluetooth radio
- `"off"` disables the Bluetooth radio

Default

`"on"`

Getvar

To retrieve the current status of the Bluetooth radio:

```
! U1 getvar "bluetooth.enable"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "bluetooth.enable" "on"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"on"`.

bluetooth.enable_reconnect

Enables the Bluetooth reconnect feature required by iOS devices.

Setvar

To enable or disable the Bluetooth reconnect feature:

```
! U1 setvar "bluetooth.enable_reconnect" "value"
```

Values

- `"off"` disables the command
- `"iOS_only"` enables the command to work with iOS devices only

Getvar

To return the current setting value:

```
! U1 getvar "bluetooth.enable_reconnect"
```

bluetooth.friendly_name

This command sets the friendly name, which is used during service discovery. For changes to take effect, you must power cycle the printer or issue the `device.reset` command. If `bluetooth.friendly_name` is not set by you, it will default to the printer serial number.

Setvar

To set the Bluetooth discoverable mode:

```
! U1 setvar "bluetooth.friendly_name" "value"
```

Values

Any text string up to 17 characters

Getvar

To retrieve the current Bluetooth discoverable mode:

```
! U1 getvar "bluetooth.friendly_name"
```

Example

This `setvar` example shows the value set to `"1234567"`.

```
! U1 setvar "bluetooth.friendly_name" "1234567"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"1234567"`.

bluetooth.json_config_channel_enable

Enables or disables the Bluetooth JSON configuration channel. Changes to this command setting only take effect when the printer is reset or power cycled.

NOTE: The JSON config channel will stop being advertised when there is no connection to the JSON channel and there is no connection on the main SPP channel.

IMPORTANT: All Bluetooth devices must first make an SPP connection before being able to make the JSON channel connection to a printer.

Setvar

To enables or disables the Bluetooth JSON configuration channel:

```
! U1 setvar "bluetooth.json_config_channel_enable" "value"
```

Values

- `"on"` means the channel is advertised and available for use only when the main serial port protocol (SPP) channel is connected.
- `"off"` means the channel is not advertised or available for use.

Default

`"on"`

Example

```
! U1 setvar "bluetooth.json_config_channel_enable" "off"
```

Getvar

To return the current setting value:

```
! U1 getvar "bluetooth.json_config_channel_enable"
```

bluetooth.power_class

This command sets the maximum transmit power level of the Bluetooth radio. If the Bluetooth radio is configured for "1", then the maximum power level of the radio is 20 dBm. If the radio is configured for "2", then the maximum power level of the radio is 4 dBm.

Setvar

To set the bluetooth power class value for the printer:

```
! U1 setvar "bluetooth.power_class" "value"
```

Values

"1"	iMZ220, iMZ320, ZD500, ZQ610, ZQ620, ZQ630, ZR658, ZR668, ZT210, ZT220, ZT410, ZT411, ZT420, ZT421, ZT510, ZT610, ZT620
"2"	iMZ220, iMZ320, ZD500, ZQ610, ZQ620, ZQ630, ZR658, ZR668, ZT210, ZT220, ZT410, ZT411, ZT420, ZT421, ZT510, ZT610, ZT620
"3"	ZT510, ZT610, ZT620

Default

"2"	ZQ610, ZQ620, ZQ630, ZR658, ZR668, iMZ220 Non-Sdio radio, iMZ320 Non-Sdio radio, ZT410 Non-Sdio Radio, ZT411 Non-Sdio Radio, ZT420 Non-Sdio Radio, ZT510 8887 Sdio Radio, ZT620 Non-Sdio Radio
"1"	ZT620 8787 Sdio Radio, ZT610 8787 Sdio Radio, ZT421 8787 Sdio Radio, ZT420 8787 Sdio Radio, ZT410 Sdio Radio, ZD500 8787 Sdio Radio

Getvar

To return the current setting:

```
! U1 getvar "bluetooth.power_class"
```

bluetooth.le.controller_mode

For printers that support both Bluetooth Classic and Bluetooth Low-Energy mode, this command controls the mode of operation.

Setvar

To control the mode of operation:

```
! U1 setvar "bluetooth.le.controller_mode" "value"
```

Values

- `"both"` means the Bluetooth radio operates in both low energy and classic mode
- `"le"` means the Bluetooth radio operates in low energy mode
- `"classic"` means the Bluetooth radio operates in the classic mode

Default

`"both"`

Getvar

To return the current setting value:

```
! U1 getvar "bluetooth.le.controller_mode"
```

bluetooth.le.power_class

This command sets the maximum transmit power level of the Bluetooth low energy radio.

Setvar

To set the power class value of the bluetooth low energy printers:

```
! U1 setvar "bluetooth.le.power_class" "value"
```

Values

The values varies printer to printer as given below.

- "1" ZQ610, ZQ620, ZQ630, ZR658, ZR668, ZT510, ZT610, ZT620
- "2" ZQ610, ZQ620, ZQ630, ZR658, ZR668, ZT510, ZT610, ZT620
- "3" ZT510, ZT610, ZT620

Default Value

The default values varies printer to printer as given below.

- "2" ZQ610, ZQ620, ZQ630, ZR658, ZR668
- "3" ZT510, ZT610, ZT620

Getvar

To return the current setting:

```
! U1 getvar "bluetooth.le.power_class"
```

bluetooth.le.minimum_security

Determines the minimum device-level security settings the printer will use to connect for Bluetooth Low-Energy (LE) connections.

Setvar

To set the device-level security settings the printer will use to connect for Bluetooth Low-Energy (LE) connections:

```
! U1 setvar "bluetooth.le.minimum_security" "value"
```

Values

- `"none"` security is not required unless the particular Bluetooth LE service or characteristic being accessed requires security. For all other choices, all services will require pairing and apply some form of security.
- `"unauth_key_signing"` (un)authenticated pairing and signing keys are required
- `"auth_key_signing"` authenticated pairing and signing keys are required.
- `"unauth_key_encrypt"` (un)authenticated pairing and encryption are required.
- `"auth_key_encrypt"` authenticated pairing and encryption are required.

Default

`"none"`if the printer has a Bluetooth LE radio. If the printer does not have a Bluetooth LE radio installed, there is no default value.

Getvar

To return the current setting value:

```
! U1 getvar "bluetooth.le.minimum_security"
```

If the printer does not have a Bluetooth LE radio installed, the printer will return an empty string.

bluetooth.page_scan_window

This command sets the page scan window timing. It can be either 15 or 60 ms. 60 ms is the default value.

Setvar

To set the page scan window timing:

```
! U1 setvar "bluetooth.page_scan_window" "value"
```

Values

"15"

"60"

Default Value

"60"

Getvar

To view the current setting:

```
! U1 getvar "bluetooth.page_scan_window"
```

bluetooth.local_name

This command retrieves the local name that is provided during service discovery.

Getvar

To retrieve the local name that is provided during service discovery:

```
! U1 getvar "bluetooth.local_name"
```

Example

In this example, the `getvar` command returns the local name that is provided during service discovery.

```
! U1 getvar "bluetooth.local_name"
```

bluetooth.minimum_security_mode

This printer setting parameter sets the Bluetooth Minimum Security mode. Minimum Security Mode provides for three levels of security, depending on the printer radio version and printer firmware: “1”, “2”, and “3”.

IMPORTANT: This feature is available in printers with Bluetooth® radio version 2.0 or higher.

Setvar

To set the Bluetooth Minimum Security mode:

```
! U1 setvar "bluetooth.minimum_security_mode" "value"
```

Values

"1"Security mode 1 means the printer will:

- be discoverable
- not require a PIN to connect to

"2" means the printer will:

- switch to bluetooth.authentication = “setpin” (Connecting device must provide the printer’s bluetooth.bluetooth_pin),
- switch to Bluetooth.encryption = “on”, and
- use existing Bluetooth.discoverable setting

"3" means Link Level Enforced Security and the printer will:

- switch to bluetooth.authentication = “setpin” (Connecting device must provide the printer’s bluetooth.bluetooth_pin)
- switch to bluetooth.encryption = “on”
- switch to bluetooth.discoverable = “off”
- only connect to devices which use Link Level Enforced Security

"4"

- switch to bluetooth.authentication = “setpin” (Connecting device must provide the printer’s bluetooth.bluetooth_pin)
- switch to bluetooth.encryption = “on”
- switch to bluetooth.discoverable = “off”
- only connect to devices which use Link Level Enforced Security
- will not connect to a device with Bluetooth version 2.1 or lower

Default

"1"

Getvar

To retrieve the current Bluetooth Minimum Security Mode:

```
! U1 getvar "bluetooth.minimum_security_mode"
```

Example

This `setvar` example shows the value set to `"2"`.

```
! U1 setvar "bluetooth.minimum_security_mode" "2"
```

bluetooth.radio_auto_baud

This command retrieves the Bluetooth radio data rate.

Getvar

To retrieve Bluetooth radio data rate:

```
! U1 getvar "bluetooth.radio_auto_baud"
```

Example

In this example, the `getvar` retrieves the short Bluetooth address.

```
! U1 getvar "bluetooth.radio_auto_baud"
```

bluetooth.radio_version

This command returns the version of the currently installed Bluetooth radio.

Getvar

To return the version of the currently installed Bluetooth radio:

```
! U1 getvar "bluetooth.radio_version"
```

Example

In this example, the `getvar` command returns the currently installed Bluetooth radio.

```
! U1 getvar "bluetooth.radio_version"
```

bluetooth.short_address

This command shortens the Bluetooth address by removing the colons (`":"`).

Getvar

To retrieve the shortened Bluetooth address:

```
! U1 getvar "bluetooth.short_address"
```

Example

In this example, the `getvar` retrieves the short Bluetooth address.

```
! U1 getvar "bluetooth.short_address"
```

bluetooth.version

This command returns the Bluetooth library version number.

Getvar

To return the Bluetooth library version number:

```
! U1 getvar "bluetooth.version"
```

Example

In this example, the `getvar` command returns the Bluetooth library version number.

```
! U1 getvar "bluetooth.version"
```

card.mac_addr

This command retrieves the MAC address of the wireless radio card.

Getvar

To instruct the printer to respond with the MAC address:

```
! U1 getvar "card.mac_addr"
```

Example

In this example, the `getvar` result is the MAC address for the wireless radio card.

```
! U1 getvar "card.mac_addr"
```

card.inserted

This command indicates whether the wireless radio card is or is not inserted.

Getvar

To instruct the printer to respond with the wireless radio card status. It's inserted or it's not inserted:

```
! U1 getvar "card.inserted"
```

Example

In this example, the `getvar` result is `"Inserted"`.

```
! U1 getvar "card.inserted"
```

external_wired.check

This command controls whether to check for external print server during the network interface search.

Setvar

To instruct the printer to set the network interface search:

```
! U1 setvar "external_wired.check" "value"
```

Values

- `"on"` = turn on external wired network interface search
- `"off"` = turn off external wired network interface search

Default

- `"on"` = If wireless option board is not installed
- `"off"` = If wireless option board is installed

Getvar

To retrieve the status of the network interface search:

```
! U1 getvar "external_wired.check"
```

Example

This `setvar` example shows the value set to `"off"`.

```
! U1 setvar "external_wired.check" "off"
```

When the `setvar` value is set to `"off"`, the `getvar` result is `"off"`.

external_wired.ip.addr

This command allows you to get or set the external wired print server's IP address.

IMPORTANT: For a set IP address to take affect, the IP protocol must be set to permanent and the print server must be reset.

Setvar

To instruct the printer to change its current external wired print server IP address upon powering the printer on:

```
! U1 setvar "external_wired.ip.addr" "value"
```

Values

Any valid IP address

Default

`"0.0.0.0"`

Getvar

To respond with the current external wired print server IP address:

```
! U1 getvar "external_wired.ip.addr"
```

On SEH print server models PS102-Z or the PS105-Z, only the getvar command is supported.

NOTE: The `setvar` value of this command can be affected by the `external_wired.ip.dhcp.enable` command.

Example

This `setvar` example shows the value set to `"10.14.4.235"`.

```
! U1 setvar "external_wired.ip.addr" "10.14.4.235"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"10.14.4.235"`.

external_wired.ip.arp_interval

This print server setting allows you to specify the ARP (Address Resolution Protocol) interval or the ARP cache time out for the external wired print server.

Setvar

To instruct the printer to change the ARP interval or the ARP cache time out for the external wired print server:

```
! U1 setvar "external_wired.ip.arp_interval" "value"
```

Values

"0" - "30"

Default

"0"

Getvar

To instruct the printer to respond with the ARP interval or the ARP cache time out value for the external wired print server:

```
! U1 getvar "external_wired.ip.arp_interval"
```

On SEH print server models PS102-Z or the PS105-Z, only the getvar command is supported.

Example

This `setvar` example shows the value set to `"0"`.

```
! U1 setvar "external_wired.ip.arp_interval" "0"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"0"`.

external_wired.ip.default_addr_enable

This command allows you to default the external wired print server's IP address.

IMPORTANT: For a set IP address to take affect, the IP protocol must be set to permanent and the print server must be reset.

Setvar

To instruct the printer to use it's default address:

```
! U1 setvar "external_wired.ip.default_addr_enable" "value"
```

If no address is provided through DHCP or BOOTP. If you do not assign an IP address after 2 minutes, the 10/100 Internal PS defaults to IP address 192.168.254.254.

Values

- `"on"` enabled
- `"off"` disabled

Default

`"on"`

Getvar

To instruct the printer to show the status of the setting of external wired print server's default IP address feature:

```
! U1 getvar "external_wired.ip.default_addr_enable"
```

On SEH print server models PS102-Z or the PS105-Z, only the getvar command is supported.

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "external_wired.ip.default_addr_enable" "on"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"on"`.

external_wired.ip.dhcp.cid_all

This printer setting defines the entire client identifier (DHCP option 61) if DHCP is enabled on the external print server and `"external_wired.ip.dhcp.cid_type"` is set to `"0"`, or `"2"`. The MAC address is used if the type is set to `"1"`.

Setvar

To instruct the printer to change the client identifier prefix and suffix of the external wired print server:

```
! U1 setvar "external_wired.ip.dhcp.cid_all" "value"
```

Values

A maximum length of 60 characters if the CID type is ASCII, or 120 characters if the CID type is hexadecimal.

Default

`""`

Getvar

To instruct the printer to respond with the client identifier prefix and suffix of the external wired print server:

```
! U1 getvar "external_wired.ip.dhcp.cid_all"
```

On SEH print server models PS102-Z or the PS105-Z, only the getvar command is supported.

Example

This `setvar` example shows the value set to `"printer"`.

```
! U1 setvar "external_wired.ip.dhcp.cid_all" "printer"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"printer"`.

external_wired.ip.dhcp.cid_enable

This command determines if DHCP (option 61) on the external wired print server is turned on or off.

Setvar

To instructs the printer to set the status of the client identifier of the external wired print server:

```
! U1 setvar "external_wired.ip.dhcp.cid_enable" "value"
```

Values

- `"off"` client identifier is turned off
- `"on"` client identifier is turned on

Default

`"off"`

Getvar

To instruct the printer to respond with the status of the client identifier of the external wired print server:

```
! U1 getvar "external_wired.ip.dhcp.cid_enable"
```

On SEH print server models PS102-Z or the PS105-Z, only the getvar command is supported.

Example

This `setvar` example shows the value set to `"off"`.

```
! U1 setvar "external_wired.ip.dhcp.cid_enable" "off"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"off"`.

external_wired.ip.dhcp.cid_prefix

This printer setting defines the prefix to be prepended to the DHCP client identifier (option 61) when DHCP is enabled on the external wired print server and "external_wired.ip.dhcp.cid_type" is set to "0" or "2".

Setvar

To instructs the printer to change the CID prefix of the external wired print server:

```
! U1 setvar "external_wired.ip.dhcp.cid_prefix" "value"
```

Values

Any text string up to 10 characters if the CID type is ASCII, or 20 characters if the CID type is hexadecimal.

Default

""

Getvar

To instruct the printer to respond with the client identifier prefix of the external wired print server:

```
! U1 getvar "external_wired.ip.dhcp.cid_prefix"
```

On SEH print server models PS102-Z or the PS105-Z, only the getvar command is supported.

Example

This setvar example shows the value set to "PRT001".

```
! U1 setvar "external_wired.ip.dhcp.cid_prefix" "PRT001"
```

What the setvar value is set to is the getvar result. In this example, the getvar result is "PRT001".

external_wired.ip.dhcp.cid_suffix

This printer setting defines the unique suffix to be used as the client identifier (DHCP option 61) if DHCP is enabled repeated on the external wired print server and `external_wired.ip.dhcp.cid_type` on page 1077 is set to `"0"` or `"2"`, not `"1"`.

Setvar

To instruct the printer to change the client identifier suffix value:

```
! U1 setvar "external_wired.ip.dhcp.cid_suffix" "value"
```

Values

The maximum length of a value allowed is 60 ASCII characters when the CID type is ASCII, or 120 hexadecimal values when the CID type is hexadecimal.

Default

`""`

Getvar

To instruct the printer to respond with the client identifier suffix on the external wired print server:

```
! U1 getvar "external_wired.ip.dhcp.cid_suffix"
```

On SEH print server models PS102-Z or the PS105-Z, only the getvar command is supported.

Example

This `setvar` example shows setting the suffix to `"printer"`.

```
! U1 setvar "external_wired.ip.dhcp.cid_suffix" "printer"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"printer"`.

external_wired.ip.dhcp.cid_type

This printer setting defines the type of client identifier (DHCP option 61) that will be sent if DHCP is enabled on the external wired print server. A value of "1" means the type of "Ethernet" and the printer's MAC address will be used. A value of "0" or "2" means the client identifier sent will be "external_wired.ip.dhcp.cid_prefix" concatenated with "external_wired.ip.dhcp.cid_suffix".

Setvar

To instruct the printer to enable "synthetic" Client Identifier for the external wired print server:

```
! U1 setvar "external_wired.ip.dhcp.cid_type" "value"
```

Values

- "0" uses an ASCII string
- "1" uses MAC address of wired print server
- "2" uses HEX value

Default

"1"

Getvar

To instruct the printer to respond with the client identifier type for the external wired print server:

```
! U1 getvar "external_wired.ip.dhcp.cid_type"
```

On SEH print server models PS102-Z or the PS105-Z, only the getvar command is supported.

Example

This setvar example shows the value set to "1".

```
! U1 setvar "external_wired.ip.dhcp.cid_type" "1"
```

When the setvar value is set to "1", the getvar result is "1".

external_wired.ip.gateway

This command instructs the printer to change the external wired print server's gateway address.

IMPORTANT: This setting refers to the gateway address. A set value is ignored if the IP protocol is not set to permanent.

Setvar

To instruct the printer to change the external wired printer server's gateway address:

```
! U1 setvar "external_wired.ip.gateway" "value"
```

Values

Any valid gateway address

Default

`"0.0.0.0"`

Getvar

To instruct the printer to respond with the external wired printer server's gateway address:

```
! U1 getvar "external_wired.ip.gateway"
```

On SEH print server models PS102-Z or the PS105-Z, only the `getvar` command is supported.

Example

This `setvar` example shows the value set to `"10.3.5.1"`.

```
! U1 setvar "external_wired.ip.gateway" "10.3.5.1"
```

When the `setvar` value is set to `"10.3.5.1"`, the `getvar` result is `"10.3.5.1"`.

external_wired.ip.netmask

This setting refers to the external wired print server's subnet mask address. This value is ignored if the IP protocol is not set to permanent.

Setvar

To instruct the printer to change the external wired print servers subnet mask:

```
! U1 setvar "external_wired.ip.netmask" "value"
```

Values

Any valid subnet mask.

Default

`"255.255.255.0"`

Getvar

To instruct the printer to respond with the external wired print server's subnet mask:

```
! U1 getvar "external_wired.ip.netmask"
```

On SEH print server models PS102-Z or the PS105-Z, only the `getvar` command is supported.

Example

This `setvar` example shows the value set to `"255.255.255.0"`.

```
! U1 setvar "external_wired.ip.netmask" "255.255.255.0"
```

When the `setvar` value is set to `"255.255.255.0"`, the `getvar` result is `"255.255.255.0"`.

external_wired.ip.port

This printer setting refers to the external wired print server's port number that the TCP print service is listening on. Normal TCP communications from the host should be directed to this port.

Setvar

To instruct the printer to set the external wired print server's TCP/UDP port number:

```
! U1 setvar "external_wired.ip.port" "value"
```

Values

1 - 65535 (excluding any ports currently used by other services, such as 21, 23, 80, and 515).

Default

`"9100"`

Getvar

To instruct the printer to respond with the external wired printer server's TCP/UDP port number:

```
! U1 getvar "external_wired.ip.port"
```

On SEH print server models PS102-Z or the PS105-Z, only the `getvar` command is supported.

Example

This `setvar` example shows the value set to `"9100"`.

```
! U1 setvar "external_wired.ip.port" "9100"
```

When the `setvar` value is set to `"9100"`, the `getvar` result is `"9100"`.

external_wired.ip.protocol

This command configures the IP addressing method used by the external wired print server.

Setvar

To instruct the printer to configure the IP addressing method used by the external wired print server:

```
! U1 setvar "external_wired.ip.protocol" "value"
```

Values

- `"bootp"` uses the standard bootp addressing method to obtain an IP address and configuration
- `"dhcp"` uses the standard dhcp addressing method to obtain an IP address and configuration for a server specified period of time
- `"rarp"` uses the standard rarp addressing method to obtain an IP address
- `"glean"` uses the IP address from a PING packet that is sent to its hardware address (unicast address)
- `"permanent"` uses static values assigned through other commands
- `"all"` tries all of the dynamic addressing methods, not permanent, to obtain an IP address

Default

`"all"`

Getvar

To return the IP addressing method used by the external print server:

```
! U1 getvar "external_wired.ip.protocol"
```

On SEH print server models PS102-Z or the PS105-Z, only the getvar command is supported.

Example

In this example, the `setvar` result is the current programming language that the printer is using.

```
! U1 setvar "external_wired.ip.protocol" "bootp"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"bootp"`.

external_wired.ip.timeout.enable

This network setting refers to enabling the connection timeout on the external wired 10/100 print server. For this to take effect, the print server must be reset.

Setvar

To instruct the printer to enable or disable the timeout checking on the external wired print server:

```
! U1 setvar "external_wired.ip.timeout.enable" "value"
```

Values

- `"off"` turns off the connection checking
- `"on"` turns on the connection checking

Default

`"on"`

Getvar

To instruct the printer to return whether the timeout checking is enabled on the external wired print server:

```
! U1 getvar "external_wired.ip.timeout.enable"
```

On SEH print server models PS102-Z or the PS105-Z, only the getvar command is supported.

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "external_wired.ip.timeout.enable" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`.

external_wired.ip.timeout.value

This network setting refers to the number of seconds before the connection times out for the external wired print server.

Setvar

To instructs the printer to set the time of the external wired print server, in seconds, before the connection times out:

```
! U1 setvar "external_wired.ip.timeout.value" "value"
```

Values

"1" through "3600"

Default

"300"

Getvar

To instruct the printer to respond with the time of the external wired print server, in seconds, before the connection times out:

```
! U1 getvar "external_wired.ip.timeout.value"
```

On SEH print server models PS102-Z or the PS105-Z, only the getvar command is supported.

Example

This `setvar` example shows the value set to "300".

```
! U1 setvar "external_wired.ip.timeout.value" "300"
```

When the `setvar` value is set to "300", the `getvar` result is "300".

external_wired.ip.v6.addr

This command retrieves the IPv6 address of the SEH wired print server. This command is only supported on SEH print server models PS105-Z and PS102-Z with firmware version V60.16.5Z or V53.16.5Z and later.

Getvar

To retrieve the IPv6 address of the SEH wired print server:

```
! U1 getvar "external_wired.ip.v6.addr"
```

Values

8 group of four hexadecimal digits with a colon delimiter

character set A-F or 0-9 with a 39-character maximum

- SEH print server model PS105-Z with firmware version V60.16.5Z or V53.16.5Z and later.
- SEH print server model PS102-Z with firmware version V60.16.5Z or V53.16.5Z and later.

Example

In this example, the `getvar` returns the IPv6 address of the wired print server.

```
! U1 getvar "external_wired.ip.v6.addr"
```

external_wired.ip.v6.gateway

This command retrieves the IPv6 gateway of the SEH wired print server.

Getvar

To retrieve the IPv6 gateway of the SEH wired print server:

```
! U1 getvar "external_wired.ip.v6.gateway"
```

Values

8 group of four hexadecimal digits with a colon delimiter

character set = A-F or 0-9 with a 39-character maximum

- SEH print server model PS105-Z with firmware version V60.16.5Z or V53.16.5Z and later.
- SEH print server model PS102-Z with firmware version V60.16.5Z or V53.16.5Z and later.

Example

In this example, the `getvar` returns the IPv6 gateway of the wired print server.

```
! U1 getvar "external_wired.ip.v6.gateway"
```

external_wired.ip.v6.prefix_length

This command retrieves the IPv6 address prefix length of the SEH wired print server.

Getvar

To retrieve the IPv6 address prefix length of the SEH wired print server:

```
! U1 getvar "external_wired.ip.v6.prefix_length"
```

Values

character set = 0-9 (3-character maximum)

- SEH print server model PS105-Z with firmware version V60.16.5Z or V53.16.5Z and later.
- SEH print server model PS102-Z with firmware version V60.16.5Z or V53.16.5Z and later.

Example

In this example, the `getvar` returns the IPv6 address prefix length of the wired print server.

```
! U1 getvar "external_wired.ip.v6.prefix_length"
```

external_wired.mac_addr

This command retrieves the MAC address of the external wired print server.

Getvar

To instruct the printer to respond with the MAC address of the external wired print server:

```
! U1 getvar "external_wired.mac_addr"
```

Example

In this example, the `getvar` result is the MAC address of the external wired print server.

```
! U1 getvar "external_wired.mac_addr"
```

external_wired.mac_raw

This command specifies the RAW MAC address of the external print server. The raw mac address is the mac address without the colons (":").

Getvar

To retrieve the RAW MAC address of the external print server:

```
! U1 getvar "external_wired.mac_raw"
```

Example

In this example, the `getvar` retrieves the RAW MAC address of the external print server.

```
! U1 getvar "external_wired.mac_raw" "00074d2408ff"
```

interface.network.active.arp_interval

This command changes the arp interval.

Setvar

To retrieve the gateway address of the active print server:

```
! U1 getvar "interface.network.active.arp_interval" "value"
```

Values

Integer values from `"0"` to `"30"`

Default

`"0"`

Getvar

To retrieve the current arp interval setting, shown in minutes:

```
! U1 getvar "interface.network.active.arp_interval"
```

Example

In this example, the `setvar` changes the arp interval to three minutes.

```
! U1 getvar "interface.network.active.arp_interval" "3"
```

interface.network.active.cable_type

This command returns the cable type of the active network, either 10/100BaseT, Wireless 802.11b/g, or Wireless 802.11n.

NOTE: This command will only give a valid response once an IP address has been established.

Getvar

To retrieve the current cable type of the active network:

```
! U1 getvar "interface.network.active.cable_type"
```

Values

- `"10/100BaseT"`
- `"Wireless 802.11b/g"`
- `"Wireless 802.11n"`

Default

NA

interface.network.active.dhcp_received_host_name

This command reports the Host Name as assigned by the DHCP Server. If one is not assigned, or DHCP is not used, then the field will be blank.

NOTE: This command will only give a valid response once an IP address has been established.

Getvar

To report the host name:

```
! U1 getvar "interface.network.active.dhcp_received_host_name"
```

Example

```
! U1 getvar "interface.network.active.dhcp_received_host_name"
```

Result

```
"Zebra Printer on shelf 2112"
```

interface.network.active.gateway

This command retrieves the gateway address of the active print server. Getvar

Getvar

To retrieve the gateway address of the active print server:

```
! U1 getvar "interface.network.active.gateway"
```

Example

In this example, the `getvar` retrieves the gateway address of the active print server.

```
! U1 getvar "interface.network.active.gateway"
"10.3.5.1"
```

interface.network.active.ip_addr

This command retrieves the IP address of the active print server.

Getvar

To retrieve the IP address of the active print server:

```
! U1 getvar "interface.network.active.ip_addr"
```

Example

In this example, the `getvar` retrieves the IP address of the active print server.

```
! U1 getvar "interface.network.active.ip_addr"
"10.3.5.92"
```

interface.network.active.ipv6.addresses

This command returns a list of up to ten IPv6 addresses.

Getvar

To retrieve the active IPv6 addresses:

```
! U1 getvar "interface.network.active.ipv6.addresses"
```

interface.network.active.ipv6.address_type

This command displays the method, either static or automatic, to get IPv6 addresses with wired or wireless networking.

Getvar

To retrieve the active network address type:

```
! U1 getvar "interface.network.active.ipv6.address_type"
```

Example

```
! U1 getvar "interface.network.active.ipv6.address_type"
```

Result

```
"auto"
```

interface.network.active.ipv6.dhcp_server_duid

This command returns the DHCP unique identifier (DUID), as provided by the DHCPv6 server that provided the printer network information.

Getvar

To retrieve the DUID:

```
! U1 getvar "interface.network.active.ipv6.dhcp_server_duid"
```

interface.network.active.ipv6.gateways

This command returns the active IPv6 network gateway(s).

Getvar

To retrieve the active network address type:

```
! U1 getvar "interface.network.active.ipv6.gateways"
```

interface.network.active.mac_addr

This command retrieves the MAC address of the active print server.

Getvar

To retrieve the MAC address of the active print server:

```
! U1 getvar "interface.network.active.mac_addr"
```

Example

In this example, the `getvar` retrieves the MAC address of the active print server.

```
! U1 getvar "interface.network.active.mac_addr"
"00:07:4d:24:08:ff"
```

interface.network.active.mac_raw

This command identifies the RAW MAC address of the active print server. The raw mac address is the mac address without the colons (":").

Getvar

To retrieve the RAW MAC address of the active print server:

```
! U1 getvar "interface.network.active.mac_raw"
```

Example

In this example, the `getvar` retrieves the RAW MAC address of the active print server.

```
! U1 getvar "interface.network.active.mac_raw"
"00074d2408ff"
```

interface.network.active.netmask

This command retrieves the netmask of the active print server.

Getvar

To retrieve the netmask of the active print server:

```
! U1 getvar "interface.network.active.netmask"
```

Example

In this example, the `getvar` retrieves the netmask of the active print server.

```
! U1 getvar "interface.network.active.netmask"
"255.255.255.0"
```

interface.network.active.protocol

This command retrieves IP protocol of the active print server.

Getvar

To retrieve the IP protocol of the active print server:

```
! U1 getvar "interface.network.active.protocol"
```

interface.network.active.protocol_error

This command returns the last error reported by the active print server.

Getvar

To return the last error reported by the active print server:

```
! U1 getvar "interface.network.active.protocol_error"
```

interface.network.active.rx_errors

This command returns the current number of packet receive errors on the active network interface.

Getvar

To return the current number of packet receive errors:

```
! U1 getvar "interface.network.active.rx_errors"
```

Result

The current number of errors.

interface.network.active.rx_packets

This command returns the number of packets received on the active network interface since the last power cycle.

Getvar

To return the number of packets received on the active network interface since the last power cycle:

```
! U1 getvar "interface.network.active.rx_packets"
```

Result
An integer.

interface.network.active.server_address

This command returns the network IP address of the server that provided the printer's IP address.

Getvar

To return the network IP address of the server providing the printer’s IP address:

```
! U1 getvar "interface.network.server_address"
```

Result

An IP address.

interface.network.active.speed

This command returns the speed in megabits per second of the active print server network since the last power cycle.

Getvar

To return the active print server network speed since the last power cycle:

```
! U1 getvar "interface.network.active.speed"
```

Result

A number indicating megabits/second.

interface.network.active.tx_errors

This command returns the number of packet transmit errors on the active print server network since the last power cycle.

Getvar

To return the number of packet transmit errors on the active print server network since the last power cycle:

```
! U1 getvar "interface.network.active.tx_errors"
```

Result

An integer number.

interface.network.active.tx_packets

This command returns the number of packets transmitted on the active print server network since the last power cycle.

Getvar

To return the number of packets transmitted on the active print server network since the last power cycle:

```
! U1 getvar "interface.network.active.tx_packets"
```

Result

A number.

interface.network.active.wins_addr

This command sets and retrieves the active WINS address.

Setvar

To set the WINS address of the active print server:

```
! U1 setvar "interface.network.active.wins_addr" "value"
```

Values

`"0.0.0.0" - "255.255.255.255"`

Getvar

To retrieve the WINS address of the active print server:

```
! U1 getvar "interface.network.active.wins_addr"
```

Result

A WINS address.

Example

In this example, the `setvar` sets the Wins address of the active print server.

```
! U1 setvar "interface.network.active.wins.addr" "10.3.5.120"
```

interface.network.settings_require_reset

Displays whether or not the network system has to be reset for a new configuration setting to take effect.

Getvar

To display whether or not the network system has to be reset for a new configuration setting to take effect:

```
! U1 getvar "interface.network.settings_require_reset"
```

Values

- `"no"` means no settings have been changed that require a reset to take effect.
- `"yes"` means one or more settings has been changed that requires a reset to take effect.

internal_wired.8021x.password

Sets the password to be used for authentication with the wired network.

Setvar

To set the command:

```
! U1 setvar "internal_wired.8021x.password" "value"
```

Values

The value is a string of up to 32 ASCII characters.

internal_wired.8021x.peap.validate_server_certificate

When using PEAP, this command determines if the printer requires the server certificate to be signed by a CA in Zebra's CA chain of trust.

Setvar

To set the command:

```
! U1 setvar "internal_wired.8021x.peap.validate_server_certificate" "value"
```

Values

- `"on"`
- `"off"`

Default

`"on"`

Getvar

To have the printer return the current setting value:

```
! U1 getvar "internal_wired.8021x.peap.validate_server_certificate"
```

internal_wired.8021x.peap.anonymous_identity

Sets the phase 1 ID name when authenticating with the wired network.

Setvar

To set the command:

```
! U1 setvar "internal_wired.8021x.peap.anonymous_identity" "value"
```

Values

The value is a string of up to 32 ASCII characters.

Default

""

Getvar

To confirm the command is set:

```
! U1 getvar "internal_wired.8021x.peap.anonymous_identity"
```

internal_wired.8021x.private_key_password

Sets the private key password for encryption in the certificate file.

Setvar

To set the command:

```
! U1 setvar "internal_wired.8021x.private_key_password" "value"
```

Values

The value is a string of up to 32 ASCII characters.

Default

""

Getvar

To have the printer return the current setting value:

```
! U1 getvar "internal_wired.8021x.private_key_password"
```

internal_wired.8021x.security

Returns the security type used for the wired network.

Setvar

To set the command:

```
! U1 setvar "internal_wired.8021x.security" "none"
```

Values

- `"none"`
- `"peap"`
- `"eap-tls"`
- `"eap-ttls"`

Default

`"none"`

Getvar

To have the printer return the current setting value:

```
! U1 getvar "internal_wired.8021x.security"
```

internal_wired.8021x.ttls_anonymous_identity

This command returns the anonymous identity string for EAP-TTLS that is used as the unencrypted identity for a wired Ethernet connection. The value of the identity string can be up to 32 characters.

Getvar

To return the anonymous identity string:

```
! U1 getvar "internal_wired.8021x.ttls_anonymous_identity"
```

Default
NA

internal_wired.8021x.ttls_tunnel

Sets the TTLS tunnel protocol to use in the authentication process.

Setvar

To set the command:

```
! U1 setvar "internal_wired.8021x.ttls_tunnel" "value"
```

Values

- `"pap"`
- `"chap"`
- `"mschap"`
- `"mschapv2"`

Default

`"mschapv2"`

Getvar

To have the printer return the current setting value:

```
! U1 getvar "internal_wired.8021x.ttls_tunnel"
```

internal_wired.8021x.username

Sets the user name to use for authentication with the wired network.

Setvar

To set the command:

```
! U1 setvar "internal_wired.8021x.username" "value"
```

Values

The value is a string of up to 32 ASCII characters.

Default

""

Getvar

To have the printer return the current setting value:

```
! U1 getvar "internal_wired.8021x.username"
```

The value is a string of up to 32 ASCII characters.

internal_wired.activity_led

Controls whether the Ethernet activity LED will blink or be solid when there is link.

Setvar

To set the Ethernet activity LED value:

```
! U1 setvar "internal_wired.activity_led" "value"
```

Values

- `"blink"` means the LED will blink
- `"solid"` means the LED will be solid

Default

`"blink"`

Getvar

To return the current setting value:

```
! U1 getvar "internal_wired.activity_led"
```

internal_wired.auto_switchover

This command instructs the printer to switch from wireless to the internal wired print server when an Ethernet cable is plugged into the printer and the printer detects an active data link.

Setvar

To configure switches between the wireless and wired interfaces:

```
! U1 setvar "internal_wired.auto_switchover" "value"
```

Values

- `"on"` indicates switchover enabled
- `"off"` indicates switchover disabled

Default

`"off"`

Getvar

To retrieve the current automatic switchover value:

```
! U1 getvar "internal_wired.auto_switchover"
```

Example

This `setvar` example shows the value set to `"off"`.

```
! U1 setvar "internal_wired.auto_switchover" "off"
```

When the `setvar` value is set to `"off"`, the `getvar` result is `"off"`.

IMPORTANT: For this command to work, be sure:

- you are using a ZM400/ZM600 or RZ400/RZ600 printer with both the internal 10/100 wired print server and wireless option board installed
- the value for this command is set to `"on"` (switchover enabled)
- the printer is currently communicating to the network through a wireless connection
- a Ethernet cable is plugged into the ZM400/ZM600 or RZ400/RZ600 printer and the printer recognizes a data link connection

When the above conditions exist and an active Ethernet cable is plugged into an internal wired print server, the printer will detect the wired data link and automatically switch to the wired interface. The printer will automatically switch back to the wireless interface when the Ethernet cable is disconnected.

internal_wired.enable

Enables or disables the wired network interface.

IMPORTANT: A network or printer reset is required for this setting to take effect.

Setvar

To set the command:

```
! U1 setvar "internal_wired.enable" "value"
```

Values

- `"on"`
- `"off"`

Default

"on"

Getvar

To have the printer return the current setting value:

```
! U1 getvar "internal_wired.enable"
```

internal_wired.installed

Returns if the wired option is installed in the printer.

Getvar

To return if the wired option is installed in the printer:

```
! U1 getvar "internal_wired.installed"
```

Values

"yes" the wired print server is installed

"no" the wired print server is not installed

internal_wired.ip.addr

This command allows you to get or set the internal wired IP address of the print server.

NOTE: For a set IP address to take affect, the IP protocol must be set to permanent and the print server must be reset.

Setvar

To instruct the printer to change its current internal wired print server IP address upon powering the printer on:

```
! U1 setvar "internal_wired.ip.addr" "value"
```

Values

Any valid IP address

Default

`"0.0.0.0"`

Getvar

To respond with the current internal wired print server IP address:

```
! U1 getvar "internal_wired.ip.addr"
```

NOTE: The `setvar` value of this command can be affected by the `internal_wired.ip.dhcp.enable` command.

Example

This `setvar` example shows the value set to `"10.14.4.235"`.

```
! U1 setvar "internal_wired.ip.addr" "10.14.4.235"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"10.14.4.235"`.

internal_wired.ip.arp_interval

This print server setting allows you to specify the ARP (Address Resolution Protocol) interval or the ARP cache time out for the internal wired print server.

Setvar

To instruct the printer to change the ARP interval or the ARP cache time out for the internal wired print server:

```
! U1 setvar "internal_wired.ip.arp_interval" "value"
```

Values

"0" - "30"

Default

"0"

Getvar

To instruct the printer to respond with the ARP interval or the ARP cache time out value for the internal wired print server:

```
! U1 getvar "internal_wired.ip.arp_interval"
```

Example

This `setvar` example shows the value set to `"0"`.

```
! U1 setvar "internal_wired.ip.arp_interval" "0"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"0"`.

internal_wired.ip.default_addr_enable

This command allows you to default the internal wired print server's IP address.

IMPORTANT: For a set IP address to take affect, the IP protocol must be set to permanent and the print server must be reset.

Setvar

To tell the printer to use it's default address:

```
! U1 setvar "internal_wired.ip.default_addr_enable" "value"
```

Values

- `"on"` means enabled
- `"off"` means disabled

Default

`"on"`

Getvar

To instruct the printer to show the status of the setting of internal wired print server's default IP address feature:

```
! U1 getvar "internal_wired.ip.default_addr_enable"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "internal_wired.ip.default_addr_enable" "on"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"on"`.

internal_wired.ip.dhcp.arp_verify

This command controls whether the internal wired print server will send an ARP response after receiving an IP address via DHCP.

Setvar

To control whether the internal wired print server will send an ARP response after receiving an IP address via DHCP:

```
! U1 setvar "internal_wired.ip.dhcp.arp_verify" "value"
```

Values

"off" ARP response will not be sent.

"on" ARP response will be sent.

Default

"off" for mobile Link-OS printers

"on" for desktop and industrial printers

Getvar

To return the current setting value:

```
! U1 getvar "internal_wired.ip.dhcp.arp_verify"
```

internal_wired.ip.dhcp.cache_ip

This command enables or disables the IP cache on the internal wired print server.

Setvar

To set the status of the IP cache:

```
! U1 setvar "internal_wired.ip.dhcp.cache_ip" "value"
```

Values

- `"on"` means enabled
- `"off"` means disabled

Default

`"off"`

Getvar

To retrieve the status of the IP cache on the internal wired print server:

```
! U1 getvar "internal_wired.ip.dhcp.cache_ip"
```

Example

This `setvar` example shows the value set to `"off"`.

```
! U1 setvar "internal_wired.ip.dhcp.cache_ip" "off"
```

When the `setvar` value is set to `"off"`, the `getvar` result is `"off"`.

internal_wired.ip.dhcp.cid_all

This printer setting defines the entire client identifier (DHCP option 61) if DHCP is enabled on the internal print server and "internal_wired.ip.dhcp.cid_type" is set to "0", or "2". The MAC address is used if the type is set to "1".

Setvar

To instruct the printer to change the client identifier prefix and suffix of the internal wired print server:

```
! U1 setvar "internal_wired.ip.dhcp.cid_all" "value"
```

The prefix gets cleared and the suffix contains the entire client identifier.

Values

A maximum length of 60 characters if the CID type is ASCII, or 120 characters if the CID type is hexadecimal.

Default

""

Getvar

To instruct the printer to respond with the client identifier prefix and suffix of the internal wired print server:

```
! U1 getvar "internal_wired.ip.dhcp.cid_all"
```

Example

This setvar example shows the value set to "printer".

```
! U1 setvar "internal_wired.ip.dhcp.cid_all" "printer"
```

What the setvar value is set to is the getvar result. In this example, the getvar result is "printer".

internal_wired.ip.dhcp.cid_enable

This command determines if DHCP (option 61) is turned on or off of the internal wired print server.

Setvar

To instruct the printer to set the status of the client identifier of the internal wired print server:

```
! U1 setvar "internal_wired.ip.dhcp.cid_enable" "value"
```

Values

- `"off"` client identifier is turned off
- `"on"` client identifier is turned on

Default

`"off"`

Getvar

To instruct the printer to respond with the status of the client identifier of the internal wired print server:

```
! U1 getvar "internal_wired.ip.dhcp.cid_enable"
```

Example

This `setvar` example shows the value set to `"off"`.

```
! U1 setvar "internal_wired.ip.dhcp.cid_enable" "off"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"off"`.

internal_wired.ip.dhcp.cid_prefix

This printer setting defines the prefix to be prepended to the DHCP client identifier (option 61) when DHCP is enabled on the internal wired print server and "internal_wired.ip.dhcp.cid_type" is set to "0" or "2".

Setvar

To instruct the printer to change the CID prefix of the internal wired print server:

```
! U1 setvar "internal_wired.ip.dhcp.cid_prefix" "value"
```

Values

Any text string up to 10 characters if the CID type is ASCII, or 20 characters if the CID type is hexadecimal.

Default

""

Getvar

To instruct the printer to respond with the client identifier prefix of the internal wired print server:

```
! U1 getvar "internal_wired.ip.dhcp.cid_prefix"
```

Example

This setvar example shows the value set to "PRT001".

```
! U1 setvar "internal_wired.ip.dhcp.cid_prefix" "PRT001"
```

What the setvar value is set to is the getvar result. In this example, the getvar result is "PRT001".

internal_wired.ip.dhcp.cid_suffix

This printer setting defines the unique suffix to be used as the client identifier (DHCP option 61) if DHCP is enabled on the internal wired 10/100 print server and `"internal_wired.ip.dhcp.cid_type"` is set to `"0"` or `"2"`.

Setvar

To instruct the printer to change the client identifier suffix value of the internal wired 10/100 print server:

```
! U1 setvar "internal_wired.ip.dhcp.cid_suffix" "value"
```

Values

The maximum length of a value allowed is 60 ASCII characters when the CID type is ASCII, or 120 hexadecimal values when the CID type is hexadecimal.

Default

`""`

Getvar

To instruct the printer to respond with the client identifier suffix of the internal wired 10/100 print server:

```
! U1 getvar "internal_wired.ip.dhcp.cid_suffix"
```

Example

This `setvar` example shows the value set to `"printer"`.

```
! U1 setvar "internal_wired.ip.dhcp.cid_suffix" "printer"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"printer"`.

internal_wired.ip.dhcp.cid_type

This printer setting defines the type of client identifier (DHCP option 61) that will be sent if DHCP is enabled on the internal wired print server. A value of `"1"` means the type of "Ethernet" and the printer's MAC address will be used.A value of `"0"` or `"2"` means the client identifier sent will be `"internal_wired.ip.dhcp.cid_prefix"` concatenated with `"internal_wired.ip.dhcp.cid_suffix"`.

Setvar

To instruct the printer to enable "synthetic" Client Identifier for the internal wired print server:

```
! U1 setvar "internal_wired.ip.dhcp.cid_type" "value"
```

Values

- `"0"` ASCII string
- `"1"` wired print server's MAC address
- `"2"` HEX value

Getvar

To instruct the printer to respond with the client identifier type for the internal wired print server:

```
! U1 getvar "internal_wired.ip.dhcp.cid_type"
```

Example

This `setvar` example shows the value set to `"1"`.

```
 ! U1 setvar "internal_wired.ip.dhcp.cid_type" "1"
```

When the `setvar` value is set to `"1"`, the `getvar` result is `"1"`.

internal_wired.ip.dhcp.lease.last_attempt

This command retrieves the last time a DHCP request was sent from the internal wired print server.

Getvar

To retrieve the last time a DHCP request was sent from the internal wired print server:

```
! U1 getvar "internal_wired.ip.dhcp.lease.last_attempt"
```

Example

In this example, the `getvar` retrieves the last time a DHCP request was sent to the internal wired print server.

```
! U1 getvar "internal_wired.ip.dhcp.lease.last_attempt"
```

internal_wired.ip.dhcp.lease.length

This command retrieves the original length (in seconds) of the DHCP lease on the internal wired print server.

Getvar

To retrieve the original length (in seconds) of the DHCP lease on the internal wired print server:

```
! U1 getvar "internal_wired.ip.dhcp.lease.length"
```

Example

In this example, the `getvar` returns the original length of the DHCP lease on the internal wired print server.

```
! U1 getvar "internal_wired.ip.dhcp.lease.length"
"691200"
```

internal_wired.ip.dhcp.lease.server

This command retrieves the address of the server that provided the DHCP lease on the internal wired print server.

Getvar

To retrieve the address of the server that provided the DHCP lease on the internal wired print server:

```
! U1 getvar "internal_wired.ip.dhcp.lease.server"
```

Example

In this example, the `getvar` retrieves the address of the server that provided the DHCP lease on the internal wired print server.

```
! U1 getvar "internal_wired.ip.dhcp.lease.server" "10.3.1.98"
```

internal_wired.ip.dhcp.lease.time_left

This command retrieves the time (in seconds) left in the current DHCP lease on the internal wired print server.

Getvar

To retrieve the time (in seconds) left in the current DHCP lease on the internal wired print server:

```
! U1 getvar "internal_wired.ip.dhcp.lease.time_left"
```

Example

In this example, the `getvar` retrieves the time left in the current DHCP lease on the wired internal print server.

```
! U1 getvar "internal_wired.ip.dhcp.lease.time_left" "10.3.1.98"
```

internal_wired.ip.dhcp.option12

This command specifies if the DHCP option 12 (host name) is on or off in the discovery packet that is sent from the internal wired print server.

Setvar

To instruct the printer to set the DHCP option 12 (host name) in the discovery packet of the internal wired print server:

```
! U1 setvar "internal_wired.ip.dhcp.option12" "value"
```

Values

- `"on"` turns on option 12
- `"off"` turns off option 12

Default

`"on"`

Getvar

To retrieve the status of the DHCP option 12 (host name) in the discovery packet of the internal wired print server:

```
! U1 getvar "internal_wired.ip.dhcp.option12"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "internal_wired.ip.dhcp.option12" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`.

internal_wired.ip.dhcp.option12_format

This command specifies the value which will be used for option 12 (host name) to be used in the DHCP discovery packet of the internal wired print server.

Setvar

To instruct the printer to set value which will be used for option 12 (host name) to be used in the DHCP discovery packet of the internal wired print server:

```
! U1 setvar "internal_wired.ip.dhcp.option12_format" "value"
```

Values

0 to 109 alphanumeric characters

Default

""

Getvar

To retrieve the value which will be used for option 12 (host name) to be used in the DHCP discovery packet of the internal wired print server:

```
! U1 getvar "internal_wired.ip.dhcp.option12_format"
```

Example

This `setvar` example shows configuring the internal_wired.ip.dhcp.option12_format to the value contained in the `device.friendly_name` parameter.

It is necessary to surround the SGD entry to be used as source for the data with the < and > characters.

```
! U1 setvar "internal_wired.ip.dhcp.option12_format"
 "<device.friendly_name>"
```

To further explain, if the above command was issued and the value currently stored in the `device.friendly_name` parameter was `"ShipPrinter"`, then the response to following command would be `"ShipPrinter"`:

```
! U1 getvar "internal_wired.ip.dhcp.option12_value"
```

internal_wired.ip.dhcp.option12_value

This command retrieves the actual value which will be used in the discovery packet of the internal wired print server.

Getvar

To retrieve the actual value which will be used in the discovery packet of the internal wired print server:

```
! U1 getvar "internal_wired.ip.dhcp.option12_value"
```

Example

This `setvar` example shows configuring the internal_wired.ip.dhcp.option12_format to the value contained in the device.friendly_name.

It is necessary to surround the SGD entry to be used as source for the data with the < and > characters.

```
! U1 setvar "internal_wired.ip.dhcp.option12_format"
 "<device.friendly_name>"
```

To further explain, if the above command was issued and the value currently stored in the device.friendly_name parameter was `"`, then the response to following command would be `"ShipPrinter"`:

```
! U1 getvar "internal_wired.ip.dhcp.option12_value"
```

internal_wired.ip.dhcp.requests_per_session

This command retrieves the maximum amount of DHCP discover requests for a single DHCP session on the internal wired print server.

Setvar

To instruct the printer to set the maximum amount of DHCP discover requests for a single DHCP session on the internal wired print server:

```
! U1 setvar "internal_wired.ip.dhcp.requests_per_session" "value"
```

Values

1-10

Default

"2"

Getvar

To retrieve the currently set maximum amount of DHCP discover requests for a single DHCP session on the internal wired print server:

```
! U1 getvar "internal_wired.ip.dhcp.requests_per_session"
```

Example

This `setvar` example shows the value set to "2".

```
! U1 setvar "internal_wired.ip.dhcp.requests_per_session" "2"
```

When the `setvar` value is set to "2", the `getvar` result is "2".

internal_wired.ip.dns.domain

This command sets the value for DNS domains in a wired network. Up to five domains are allowed to be set, separated by spaces.

Setvar

To set values to be used by the device as DNS domains when `internal_wired.ipv6.address_type` is `"static"` (IPv6) or `internal_wired.ip.protocol` is `"permanent"` (IPv4):

```
! U1 setvar "internal_wired.ip.dns.domain" "value"
```

Getvar

To retrieve a space-delimited list of domains currently in use by the device, up to five domains. The list will contain a mixture of domains received from DHCPv4, DHCPv6, and user-set values depending upon address acquisition settings, using the following rules:

- IPv4 sources are the priority.
- If IPv4 the source provides five domains, there will only be IPv4-sourced domains.
- If there are slots to fit more domains, they will be filled with IPv6 sources.

```
! U1 getvar "internal_wired.ip.dns.domain"
```

NOTE: Retrieved values are always the values currently in use, which may not match values that were just set, depending on if you are using a static or auto address type.

Example

The following example sets `internal_wired.ip.dns.domain` to the domains `zebra.com` and `zebra-lab.lan.com`.

These values will be retrieved immediately if `internal_wired.ipv6.address_type` is `"static"` (IPv6) or `internal_wired.ip.protocol` is `"permanent"` (IPv4).

```
! U1 setvar "internal_wired.ip.dns.domain" "zebra.com zebra-lab.lan.com"
```

```
! U1 getvar "internal_wired.ip.dns.domain"
```

Result

```
"zebra.com zebra-lab.lan.com"
```

internal_wired.ip.dns.domain_user_value

This command displays the value that the user has set to `internal_wired.ip.dns.domain`. Whether this value is used will depend on the rules for `internal_wired.ip.dns.domain`.

Getvar

To retrieve the domain name(s) from an internal wired print server:

```
! U1 getvar "internal_wired.ip.dns.domain_user_value"
```

internal_wired.ip.dns.servers

This command sets a space-delimited list of the domain name servers from an internal wired print server. Up to three addresses may be set. Both IPv4 and IPv6 are supported.

Setvar

To set the list of DNS internal wired print servers, specify a list of space-delimited IP addresses, separated by spaces, to be used as DNS servers when `internal_wired.ipv6.address_type` is `"static"` (IPv6) or `internal_wired.ip.protocol` is `"permanent"` (IPv4).

```
! U1 setvar "internal_wired.ip.dns.servers" "value"
```

Getvar

To retrieve a space-delimited list of IP address(es) of DNS server(s), up to three, that are currently in use by the device. The values to be used are a combination of addresses received from DHCPv4, DHCPv6, and user-set values using the following rules:

- At least one spot will be allocated to any enabled IP version.
- IPv4 addresses will take at least two slots if at least two IPv4 addresses are provided.
- User-set values will be included if static/permanent addresses are used.

NOTE: Retrieved values are always the values currently in use, which may not match values that were just set, depending on if you are using a static or auto address type.

```
! U1 getvar "internal_wired.ip.dns.servers"
```

Example

The following example sets `internal_wired.dns.servers` to `2001::123:4567:89ab:0:cdef` .

This value will be retrieved immediately if `internal_wired.ipv6.address_type` is `"static"` (IPv6) or `internal_wired.ip.protocol` is `"permanent"` (IPv4).

```
! U1 setvar "internal_wired.ip.dns.servers" "2001::123:4567:89ab:0:cdef"
```

```
! U1 getvar "internal_wired.ip.dns.servers"
```

Result

```
"2001::123:4567:89ab:0:cdef"
```

internal_wired.ip.dns.servers_user_value

This command displays the value that the user has set to `internal_wired.ip.dns.servers`. Whether this value is used will depend on the rules for `internal_wired.ip.dns.servers`.

Getvar

To retrieve the domain name server(s):

```
! U1 getvar "internal_wired.ip.dns.servers_user_value"
```

internal_wired.ip.gateway

This command instructs the printer to change the internal wired print servers gateway address.

IMPORTANT: This setting refers to the gateway address. A set value is ignored if the IP protocol is not set to permanent.

Setvar

To instruct the printer to change the internal wired printer servers gateway address:

```
! U1 setvar "internal_wired.ip.gateway" "value"
```

Values

Any valid gateway address

Default

`"0.0.0.0"`

Getvar

To instruct the printer to respond with the internal wired printer servers gateway address:

```
! U1 getvar "internal_wired.ip.gateway"
```

Example

This `setvar` example shows the value set to `"10.3.5.1"`.

```
! U1 setvar "internal_wired.ip.gateway" "10.3.5.1"
```

When the `setvar` value is set to `"10.3.5.1"`, the `getvar` result is `"10.3.5.1"`.

internal_wired.ip.netmask

This setting refers to the internal wired print server's subnet mask address. This value is ignored if the IP protocol is not set to permanent.

Setvar

To instruct the printer to change the internal wired print servers subnet mask:

```
! U1 setvar "internal_wired.ip.netmask" "value"
```

Values

Any valid subnet mask.

Default

`"255.255.255.0"`

Getvar

To instruct the printer to respond with internal wired print servers subnet mask:

```
! U1 getvar "internal_wired.ip.netmask"
```

Example

This `setvar` example shows the value set to `"255.255.255.0"`.

```
! U1 setvar "internal_wired.ip.netmask" "255.255.255.0"
```

When the `setvar` value is set to `"255.255.255.0"`, the `getvar` result is `"255.255.255.0"`.

internal_wired.ip.port

This printer setting refers to the internal wired print servers port number that the TCP print service is listening on. Normal TCP communications from the host should be directed to this port.

Setvar

To instruct the printer to set the internal wired print servers TCP/UDP port number:

```
! U1 setvar "internal_wired.ip.port" "value"
```

Values

1 - 65535 (excluding any ports currently used by other services, such as 21, 23, 80, and 515).

Default

`"9100"`

Getvar

To instruct the printer to respond with the internal wired printer servers TCP/UDP port number:

```
! U1 getvar "internal_wired.ip.port"
```

Example

This `setvar` example shows the value set to `"9100"`.

```
! U1 setvar "internal_wired.ip.port" "9100"
```

When the `setvar` value is set to `"9100"`, the `getvar` result is `"9100"`.

internal_wired.ip.port_alternate

This command sets the port number of the alternate port.

NOTE: Print servers supporting this command will monitor both the primary port and the alternate port for connections at the same time.

Setvar

To set the alternate port for the print server:

```
! U1 setvar "internal_wired.ip.port_alternate" "value"
```

Values

Any valid TCP port address.

Default

"9100"

Getvar

To return the current alternate port setting:

```
! U1 setvar "internal_wired.ip.port_alternate"
```

internal_wired.ip.port_json_config

This command determines the TCP port number to listen on for JSON configuration packets.

Setvar

To set the TCP port number to listen on for JSON configuration packets:

```
! U1 setvar "internal_wired.ip.port_json_config" "value"
```

Values

- `"0"` disables the port
- `1-65535` Specifies the port number to listen on.

Ports that are already is use or the standard network ports are invalid values.#

Default

`"9200"`

Getvar

To retrieve the port number:

```
! U1 getvar "internal_wired.ip.port_json_config"
```

Example

This example sets the port value to listen on as 1234.

```
! U1 setvar "internal_wired.ip.port_json_config" "1234"
```

internal_wired.ip.protocol

This command configures the IP addressing method used by the internal wired print server.

Setvar

To instruct the printer to configure the IP addressing method used by the internal wired print server:

```
! U1 setvar "internal_wired.ip.protocol" "value"
```

Values

- `"bootp"` uses the standard bootp addressing method to obtain an IP address and configuration
- `"dhcp"` uses the standard dhcp addressing method to obtain an IP address and configuration for a server specified period of time
- `"rarp"` uses the standard rarp addressing method to obtain an IP address
- `"glean"` uses the IP address from a PING packet that is sent to its hardware address (unicast address)
- `"permanent"` uses static values assigned through other commands
- `"all"` = tries all of the dynamic addressing methods, not permanent, to obtain an IP address

Default

`"all"`

Getvar

To return the IP addressing method used by the internal wired print server:

```
! U1 getvar "internal_wired.ip.protocol"
```

Example

In this example, the `setvar` result is the current programming language that the printer is using.

```
! U1 setvar "internal_wired.ip.protocol" "bootp"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"bootp"`.

internal_wired.ip.timeout.enable

This network setting refers to enabling the connection timeout on the internal wired print server. For this to take effect, the print server must be reset.

Setvar

To instruct the printer to enable or disable the timeout checking on the internal wired print server:

```
! U1 setvar "internal_wired.ip.timeout.enable" "value"
```

Values

- `"off"` turns off the connection checking
- `"on"` turns on the connection checking

Default

`"on"`

Getvar

To instruct the printer to return whether the timeout checking is enabled on the internal wired print server:

```
! U1 getvar "internal_wired.ip.timeout.enable"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "internal_wired.ip.timeout.enable" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`.

internal_wired.ip.timeout.value

This network setting refers to the number of seconds before the connection times out for the internal wired print server. For this to take effect, the print server must be reset.

Setvar

To instruct the printer to set the time of the internal wired print server, in seconds, before the connection times out:

```
! U1 setvar "internal_wired.ip.timeout.value" "value"
```

Values

"1" through "3600"

Default

"300"

Getvar

To instruct the printer to respond with the time of the internal wired print server, in seconds, before the connection times out:

```
! U1 getvar "internal_wired.ip.timeout.value"
```

Example

This `setvar` example shows the value set to "300".

```
! U1 setvar "internal_wired.ip.timeout.value" "300"
```

When the `setvar` value is set to "300", the `getvar` result is "300".

internal_wired.ip.wins.addr

Sets or returns the current WINS server address. If WinsAddressing is DHCP, then this will automatically be filled by the DHCP server.

Setvar

To set the current WINS server address:

```
! U1 setvar "internal_wired.ip.wins.addr" "value"
```

Values

A valid WINS IP address.

Default

`"0.0.0.0"`

Getvar

To return the WINS server address:

```
! U1 getvar "internal_wired.ip.wins.addr"
```

Result

The current WINS server address

internal_wired.ip.wins.permanent_source

Specifies if the WINS address will be permanent or set via DHCP.

Setvar

To specify if the WINS address will be permanent or set via DHCP:

```
! U1 setvar "internal_wired.ip.wins.permanent_source" "value"
```

Values

- `"off"` use DHCP-assigned WINS address
- `"on"` use currently stored WINS address

The WINS address can be set using the `interface.network.active.wins_address` command.

Default

`"off"`

Getvar

To return the current setting value:

```
! U1 getvar "internal_wired.ip.wins.permanent_source"
```

internal_wired.ipv6.addresses

This command returns a list of up to ten IPv6 addresses in use by the printer with a wired connection when `internal_wired.ipv6.address_type` is set to `auto`. The command returns`"::"`when no address has been set.

Getvar

To retrieve the IPv6 addresses:

```
! U1 getvar "internal_wired.ipv6.addresses"
```

internal_wired.ipv6.address_type

This command controls how the interface gets IPv6 address(es). When set to `"auto"`, the Router Advertisement packets determine the usage of SLAAC and/or DHCP.

Setvar

To set the active network address type:

```
! U1 setvar "internal_wired.ipv6.address_type" "value"
```

Values

`"static"` or `"auto"`

Default

`"auto"`

Getvar

To retrieve the address type:

```
! U1 getvar "internal_wired.ipv6.address_type"
```

Example

This example changes the address type to `"static"`.

```
! U1 setvar "internal_wired.ipv6.address_type" "static"
```

```
! U1 getvar "internal_wired.ipv6.address_type"
```

Result

```
"static"
```

internal_wired.ipv6.dhcp.lease.last_attempt

This command retrieves the Unix timestamp of the last attempt to obtain a DHCPv6 lease for wired communication. The value is a whole number up to 4,294,967,295 in seconds..

Getvar

To retrieve the time of the last attempt:

```
! U1 getvar "internal_wired.ipv6.dhcp.lease.last_attempt"
```

internal_wired.ipv6.dhcp.lease.length

This command retrieves the value of the lease duration for an IP address provided by the DHCPv6 in seconds with a wired connection. The value is a whole number up to 4,294,967,295 in seconds.

Getvar

To retrieve the lease length:

```
! U1 getvar "internal_wired.ipv6.dhcp.lease.length"
```

internal_wired.ipv6.dhcp.lease.time_left

This command retrieves the value of the remaining lease duration for an IP address provided by the DHCPv6 in seconds with a wired connection. The value is a whole number up to 4,294,967,295 in seconds.

Getvar

To retrieve the lease time left:

```
! U1 getvar "internal_wired.ipv6.dhcp.lease.time_left"
```

internal_wired.ipv6.dhcp.option39_enable

This command enables option 39 (to configure the Fully Qualified Domain Name or FQDN) in DHCPv6 communication with a wired connection.

Setvar

To enable or disable option 39:

```
! U1 setvar "internal_wired.ipv6.dhcp.option39_enable" "value"
```

Values

`"on"` or `"off"`

Default

`"on"`

Getvar

To retrieve the option 39 status:

```
! U1 getvar "internal_wired.ipv6.dhcp.option39_enable"
```

Example

```
! U1 setvar "internal_wired.ipv6.dhcp.option39_enable" "off"
```

```
! U1 getvar "internal_wired.ipv6.dhcp.option39_enable"
```

Result

```
"off"
```

internal_wired.ipv6.dhcp.option39_format

This command sets the value of the format field for option 39 (to configure the Fully Qualified Domain Name or FQDN) in DHCPv6 with a wired connection. The value is a string up to 127 characters, and the default is the value set by `device.friendly_name`.

Setvar

To set the format string:

```
! U1 setvar "internal_wired.ipv6.dhcp.option39_format" "value"
```

where `"value"` is a string of up to 127 characters.

If the value is a source SGD command, it must be bracketed with `"<"and">"`, such as `"<device.friendly_name>"`.

Getvar

To retrieve the string:

```
! U1 getvar "internal_wired.ipv6.dhcp.option39_format"
```

Example

This command sets the value to `"printer2"`.

```
! U1 setvar "internal_wired.ipv6.dhcp.option39_format" "printer2"
```

```
! U1 getvar "internal_wired.ipv6.dhcp.option39_format"
```

Result

```
"printer2"
```

internal_wired.ipv6.dhcp.option39_fqdn

This command retrieves the fully qualified domain name (FQDN) of the printer as given by the DHCPv6 server for option 39 with a wired connection. The value is a string up to 127 characters.

Getvar

To retrieve the FQDN:

```
! U1 getvar "internal_wired.ipv6.dhcp.option39_fqdn"
```

internal_wired.ipv6.dhcp.option39_value

This command retrieves the value to be used for option 39 in DHCPv6 after processing the `internal_wired.ipv6.dhcp.option39_format` command with a wired connection. The value is a string of up to 127 characters.

Getvar

To retrieve the option 39 value:

```
! U1 getvar "internal_wired.ipv6.dhcp.option39_value"
```

internal_wired.ipv6.gateways

This command retrieves the IPv6 gateway(s) for internal wired devices in use by the printer (maximum list of 10).

Getvar

To retrieve the gateway(s):

```
! U1 getvar "internal_wired.ipv6.gateways"
```

internal_wired.ipv6.static.addresses

This command specifies the IPv6 address(es) to be used when `internal_wired.ipv6.address_type` is set to `"static"`. Up to three addresses may be set, separated by commas. The command returns `"::"` when no static address has been set.

Setvar

To set the IPv6 address(es):

```
! U1 setvar "internal_wired.ipv6.static.addresses" "value"
```

where `"value"` is a string of up to 152 characters.

Getvar

To retrieve the list of IPv6 address(es):

```
! U1 getvar "internal_wired.ipv6.static.addresses"
```

Example

To set three addresses:

```
! U1 setvar "internal_wired.ipv6.static.addresses"
 "fc04:1795::fe94:1704/32,fd04:1795::207:4dff:fe94:1704/64,fd04:1796::e0b/64"
```

```
! U1 getvar "internal_wired.ipv6.static.addresses"
```

Result

```
"fc04:1795::fe94:1704/32,fd04:1795::207:4dff:fe94:1704/64,
 fd04:1796::e0b/64"
```

internal_wired.ipv6.static.gateways

This command sets the IPv6 gateway to be used when `internal_wired.ipv6.address_type` is set to `"static"`. Only one gateway is supported. The command returns `"::"` when no static address has been set.

Setvar

To set the gateway:

```
! U1 setvar "internal_wired.ipv6.static.gateways" "value"
```

where `"value"` is a string of up to 50 characters. To clear a previously set gateway, use `""` or `"::"`.

Getvar

To retrieve the active network address type:

```
! U1 getvar "internal_wired.ipv6.static.gateways"
```

Example

To set the gateway to `"fe80::202:b3ff:febf:9d18"`:

```
! U1 setvar "internal_wired.ipv6.static.gateways" "fe80::202:b3ff:febf:9d18"
```

```
! U1 getvar "internal_wired.ipv6.static.gateways"
```

Result

```
"fe80::202:b3ff:febf:9d18"
```

internal_wired.mac_addr

This command retrieves the MAC address of the internal wired print server.

Getvar

To instruct the printer to respond with the MAC address of the internal wired print server:

```
! U1 getvar "internal_wired.mac_addr"
```

Example

In this example, the `getvar` result is the MAC address of the internal wired print server.

```
! U1 getvar "internal_wired.mac_addr"
```

internal_wired.mac_raw

This command identifies the RAW MAC address of the internal wired print server. The raw mac address is the mac address without the colons (":").

Getvar

To retrieve the RAW MAC address of the internal wired print server:

```
! U1 getvar "internal_wired.mac_raw"
```

Example

In this example, the `getvar` retrieves the RAW MAC address of the internal wired print server.

```
! U1 getvar "internal_wired.mac_raw"
```

ip.active_network

This command displays if the printer is actively connected to wireless, external wired, or internal wired.

Getvar

To return the current active network the printer is connected to:

```
! U1 getvar "ip.active_network"
```

The Printer Response table below details on the potential return values.

Example

In this example, the `getvar` will return the current active network the printer is connected to.

```
! U1 getvar "ip.active_network"
```

Table 27 Printer Response

Return Values	Details
`"internal wired"`	This is the return value when an internal wired device is detected.
`"wireless"`	This is the return value when a wireless device is detected.
`"external wired"`	This is the return value when an external wired device is detected.
`"unknown"`	This is the return value: • if the printer has not established a network connection on any of the devices • if you don't have any of the network devices plugged in • if the printer is still trying to establish a connection (i.e. on wireless it is going through the association process).

ip.addr

This command allows you to get or set the printer's IP address.

IMPORTANT: For a set IP address to take affect, the IP protocol must be set to permanent and the print server must be reset.

Setvar

To instruct the printer to change its current IP address upon powering the printer on:

```
! U1 setvar "ip.addr" "value"
```

Values

Any valid IP address

Default

`"0.0.0.0"`

Getvar

To instruct the printer to respond with its current IP address:

```
! U1 getvar "ip.addr"
```

NOTE: The `setvar` value of this command can be affected by the `ip.dhcp.enable` command.

Example

This `setvar` example shows the value set to `"10.14.4.235"`.

```
! U1 setvar "ip.addr" "10.14.4.235"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"10.14.4.235"`.

ip.address_mode

This command sets the IP protocol value for the printer.

Setvar

To set the IP protocol:

```
! U1 setvar "ip.address_mode" "value"
```

Values

`"ipv4"`, `"ipv6"`, `"all"`

Default

`"ipv4"`

Getvar

To retrieve the IP protocol value:

```
! U1 getvar "ip.address_mode"
```

Example

```
! U1 setvar "ip.address_mode" "ipv6"
```

```
! U1 getvar "ip.address_mode"
```

Result

```
"ipv6"
```

ip.arp_interval

This printer setting allows you to specify the ARP (Address Resolution Protocol) interval or the ARP cache time out.

Setvar

To instruct the printer to change the ARP interval or the ARP cache time out:

```
! U1 setvar "ip.arp_interval" "value"
```

Values

"0" through "30"

Default

"0"

Getvar

To instruct the printer to respond with the ARP interval or the ARP cache time out value in seconds:

```
! U1 getvar "ip.arp_interval"
```

Example

This `setvar` example shows the value set to "0".

```
! U1 setvar "ip.arp_interval" "0"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is "0".

ip.bootp.enable

This printer setting turns BOOTP on or off. BOOTP is a method for acquiring an IP address, netmask, and gateway automatically on printer power-up. It requires a BOOTP server on the local network.

NOTE: If you are using static IP addressing, the IP protocol must be set to permanent.

Setvar

To instruct the printer to turn BOOTP on or off:

```
! U1 setvar "ip.bootp.enable" "value"
```

Values

- `"off"` printer does not use BOOTP to get the IP address
- `"on"` printer uses BOOTP to get the IP address

Default

`"on"`

Getvar

To instructs the printer to respond with the current BOOTP setting:

```
! U1 getvar "ip.bootp.enable"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "ip.bootp.enable" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`.

ip.dhcp.arp_verify

This command configures the print server to send out an ARP request during the DHCP address negotiation. This is used to verify if the IP address received from the DHCP server is in use.

- If an ARP reply is received, a DHCP DECLINE message is sent to the DHCP server telling it that the received address cannot be used and then the normal DHCP procedure is restarted.
- If no ARP reply is received the DHCP address is used.

Setvar

To instruct the printer to turn on ARP verify:

```
! U1 setvar "ip.dhcp.arp_verify" "value"
```

Values

- `"on"`
- `"off"`

Default

`"off"`

Getvar

To return whether the printer will send the ARP request during the DHCP negotiation:

```
! U1 getvar "ip.dhcp.arp_verify"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "ip.dhcp.arp_verify" "on"
```

ip.dhcp.auto_provision_enable

This command prepares the printer to receive Weblink settings from the DHCP server. These settings are used to allow the printer to connect to on-premise or web-based servers.

If DHCP receives option 43 in the format of "10642 <auto provision code> < auto provision settings>" and the Client is not already configured, the client will use the configuration to connect to the server.

Using this feature requires configuring your DHCP server to send option 43 information during address assignment. The feature allows the printer to obtain the settings used to control Cloud Connect weblink connections or Mirror events as part of receiving a DHCP assigned IP address. This requires that DHCP option 60 is not empty and that `ip.dhcp.auto_provision_enable` is set to "on".

The package of Cloud Connect/weblink information sent from the DHCP server in the Option 43 response can include the:

- Server address
- Authentication server name
- User name and password for proxy logins

The package of Mirror information sent from the DHCP server in the Option 43 packet can include the:

- Server address
- Mirror path
- Mirror feedback path
- Mirror appl path
- Mirror mode

Setvar

To enable or disable the DHCP auto provision feature of the printer:

```
! U1 setvar "ip.dhcp.auto_provision_enable" "value"
```

Values

- "on" enabled
- "off" disabled

Default

"off"

Getvar

To retrieve the current value of on or off for the DHCP auto provision feature:

```
! U1 getvar "ip.dhcp.auto_provision_enable"
```

ip.dhcp.cache_ip

This command enables or disables the IP caching.

Setvar

To set the status of the IP cache:

```
! U1 setvar "ip.dhcp.cache_ip" "value"
```

Values

- `"on"` enabled
- `"off"` disabled

Default

`"off"`

Getvar

To retrieve the status of the IP cache:

```
! U1 getvar "ip.dhcp.cache_ip"
```

Example

This `setvar` example shows the value set to `"off”`.

```
! U1 setvar "ip.dhcp.cache_ip" "off”
```

When the `setvar` value is set to `"off"`, the `getvar` result is `"off"`.

ip.dhcp.cid_all

This printer setting defines the entire client identifier (DHCP option 61) if the DHCP is enabled and `"ip.dhcp.cid_type"` is set to `"0"`, or `"2"`. The MAC address is used if the type is set to `"1"`.

Setvar

To instruct the printer to change the CID prefix and suffix:

```
! U1 setvar "ip.dhcp.cid_all" "value"
```

Values

A maximum length of 60 characters if the CID type is ASCII, or 120 characters if the CID type is hexadecimal.

Default

`""`

Getvar

To instruct the printer to respond with the client identifier prefix and suffix:

```
! U1 getvar "ip.dhcp.cid_all"
```

Example

This `setvar` example shows the value set to `"printer"`.

```
! U1 setvar "ip.dhcp.cid_all" "printer"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"printer"`.

ip.dhcp.cid_enable

This command determines if DHCP (option 61) is turned on or off.

Setvar

To set the status of the client identifier:

```
! U1 setvar "ip.dhcp.cid_enable" "value"
```

Values

- `"off"` client identifier is turned off
- `"on"` client identifier is turned on

Default

`"off"`

Getvar

To return the current setting value:

```
! U1 getvar "ip.dhcp.cid_enable"
```

Example

This `setvar` example shows the value set to `"off"`.

```
! U1 setvar "ip.dhcp.cid_enable" "off"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"off"`.

ip.dhcp.cid_prefix

This printer setting defines the prefix to be prepended to the DHCP client identifier (option 61) when DHCP is enabled and "ip.dhcp.cid_type" is set to "0"or "2".

Setvar

To instruct the printer to change the CID prefix:

```
! U1 setvar "ip.dhcp.cid_prefix" "value"
```

Values

Any text string up to 10 characters if the CID type is ASCII, or 20 characters if the CID type is hexadecimal.

Default

""

Getvar

To instruct the printer to respond with the client identifier prefix:

```
! U1 getvar "ip.dhcp.cid_prefix"
```

Example

This setvar example sets the value to "ZEB".

```
! U1 setvar "ip.dhcp.cid_prefix" "ZEB"
```

What the setvar value is set to is the getvar result. In this example, the getvar result is "ZEB".

The next time the printer sends a DHCP request, if ip.dhcp.cid_type is "0", the client identifier sent will be prefixed with the string "ZEB". For example, if ip.dhcp.cid_value is "PRT001", the actual client identifier sent will be "ZEBPRT001"

.

ip.dhcp.cid_suffix

This printer setting defines the unique suffix to be used as the client identifier (DHCP option 61) if DHCP is enabled and `"ip.dhcp.cid_type"` is set to `"0"` or `"2"`.

Setvar

To instruct the printer to change the CID value:

```
! U1 setvar "ip.dhcp.cid_suffix" "value"
```

Values

The maximum length of a value allowed is 60 ASCII characters when the CID type is ASCII, or 120 hexadecimal values when the CID type is hexadecimal.

Default

`""`

Getvar

To instruct the printer to respond with the client identifier suffix:

```
! U1 getvar "ip.dhcp.cid_suffix"
```

Example

This `setvar` example shows the value set to `"printer"`.

```
! U1 setvar "ip.dhcp.cid_suffix" "printer"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"printer"`.

ip.dhcp.cid_type

This printer setting defines the type of client identifier (DHCP option 61) that will be sent if DHCP is enabled. A value of "1" means the type of "Ethernet" and the printer's MAC address will be used. A value of "0" or "2" means the client identifier sent will be "ip.dhcp.cid_prefix" concatenated with "ip.dhcp.cid_suffix".

Setvar

To instruct the printer to set the Client Identifier type:

```
! U1 setvar "ip.dhcp.cid_type" "value"
```

Values

- "0" ASCII string
- "1"MAC address of the wireless radio card
- "2"HEX value

Default

"1"

Getvar

To instruct the printer to respond with the client identifier type:

```
! U1 getvar "ip.dhcp.cid_type"
```

Example

This setvar example shows the value set to "1".

```
! U1 setvar "ip.dhcp.cid_type" "1"
```

When the setvar value is set to "1", the getvar result is "1".

ip.dhcp.cid_value

This parameter defines the unique value to be used as the client identifier (option 61) if DHCP is enabled and "ip.dhcp.cid_type" is "1".

NOTE: This parameter is only applicable if ip.dhcp.cid_enable is set to "on".

Setvar

To instruct the printer to change the CID value:

```
! U1 setvar "ip.dhcp.cid_value" "value"
```

Values

Any text string up to 20 characters in length.

Default

If ip.dhcp.cid_type is set to:

- "0" the default is the printer's friendly name.
- "1" the default is the MAC address of the printer.

Getvar

To instruct the printer to respond with the client identifier value:

```
! U1 getvar "ip.dhcp.cid_value"
```

Example

This setvar example changes the cid value to "PRT001".

```
! U1 setvar "ip.dhcp.cid_value" "PRT001"
```

The next time the printer sends a DHCP request, if ip.dhcp.cid_type is "0", the client identifier sent will be ip.dhcp.cid_prefix plus "PRT001". For example, if ip.dhcp.cid_prefix is "ZEB", the actual client identifier sent will be "ZEBPRT001".

ip.dhcp.dhcpv6_duid

This command retrieves the DHCPv6 DUID that the printer uses in DHCPv6 communication. It is listed in groups of quad hex digits separated by colons. The value is used for both wired and wireless DHCPv6.

Getvar

To retrieve the DHCPv6 DUID value:

```
! U1 getvar "ip.dhcp.dhcpv6_duid"
```

ip.dhcp.enable

This printer setting turns DHCP on or off. DHCP is a method for acquiring an IP address, netmask, and gateway automatically on printer power-up. It requires a DHCP server on the local network.

NOTE: If you are using static IP addressing, the IP protocol must be set to permanent.

Setvar

To instruct the printer to turn DHCP on or off:

```
! U1 setvar "ip.dhcp.enable" "value"
```

Values

- `"off"` printer does not use DHCP to get the IP address
- `"on"` printer uses DHCP to get the IP address

Default

`"on"`

Getvar

To instruct the printer to respond with the DHCP status:

```
! U1 getvar "ip.dhcp.enable"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "ip.dhcp.enable" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`.

ip.dhcp.lease.last_attempt

This command retrieves the time from the DHCP server of when the last DHCP request was sent.

Getvar

To retrieve the last time a DHCP request was sent:

```
! U1 getvar "ip.dhcp.lease.last_attempt"
```

Example

In this example, the `getvar` retrieves the last time a DHCP request was sent to the wireless print server.

```
! U1 getvar "ip.dhcp.lease.last_attempt"
```

ip.dhcp.lease.server

This command retrieves the address of the server that provided the DHCP lease on the wireless print server.

Getvar

To retrieve the address of the server that provided the DHCP lease on the wireless print server:

```
! U1 getvar "ip.dhcp.lease.server"
```

Example

In this example, the `getvar` retrieves the server that provided the DHCP lease on the wireless print server.

```
! U1 getvar "ip.dhcp.lease.server"
"10.3.5.1"
```

ip.dhcp.lease.time_left

This command retrieves the time left in the current DHCP lease on the wireless print server.

Getvar

To retrieve the time left in the current DHCP lease on the wireless print server:

```
! U1 getvar "ip.dhcp.lease.time_left"
```

Example

In this example, the `getvar` retrieves the time left in the current DHCP lease on the wireless print server.

```
! U1 getvar "ip.dhcp.lease.time_left"
"1192518"
```

ip.dhcp.ntp.enable

This command controls whether or not the printer retrieves the address of a Network Time Protocol (NTP) server during DHCP address assignment.

Setvar

To set whether or not the printer retrieves the address of a Network Time Protocol (NTP) server during DHCP address assignment:

```
! U1 setvar "ip.dhcp.ntp.enable" "value"
```

Values

- `"off"` does not request the NTP server address
- `"on"` requests the NTP server address

Default

`"off"`

Getvar

To return the current setting value:

```
! U1 getvar "ip.dhcp.ntp.enable"
```

ip.dhcp.ntp.received_servers

Returns the list of Network Time Protocol (NTP) server IP Addresses received via DHCP.

Getvar

To return the current setting value:

```
! U1 getvar "ip.dhcp.ntp.received_servers"
```

Values

A comma-separated list of IP address. The maximum number of servers listed is three (3).

Default

""

Example

```
"10.4.3.24,172.30.16.52"
```

ip.dhcp.option12

This command specifies if the DHCP option 12 (host name) is on or off in the discovery packet that is sent from the wireless print server.

Setvar

To instructs the printer to set the DHCP option 12 (host name) in the discovery packet that is sent from the wireless print server:

```
! U1 setvar "ip.dhcp.option12" "value"
```

Values

- `"on"` turns on option 12
- `"off"` turns off option 12

Default

`"on"`

Getvar

To retrieve the status of the DHCP option 12 (host name) in the discovery packet that is sent from the wireless print server:

```
! U1 getvar "ip.dhcp.option12"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "ip.dhcp.option12" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`.

ip.dhcp.option12_format

This command specifies the value which will be used for option 12 (host name) to be used in the DHCP discovery packet of the wireless print server.

Setvar

To instruct the printer to set the value which will be used for option 12 (host name) to be used in the DHCP discovery packet of the wireless print server:

```
! U1 setvar "ip.dhcp.option12_format" "value"
```

Values

string

Default

""

Getvar

To retrieve the value which will be used for option 12 (host name) to be used in the DHCP discovery packet of the wireless print server:

```
! U1 getvar "ip.dhcp.option12_format"
```

Example

This `setvar` example shows configuring the `ip.dhcp.option12` format to the value contained in the `device.friendly_name`.

It is necessary to surround the SGD entry to be used as source for the data with the < and > characters.

```
! U1 setvar "ip.dhcp.option12_format" "<device.friendly_name>"
```

To further explain, if the above command was issued and the value currently stored in the `device.friendly_name` parameter was `"ShipPrinter"`, then the response to following command would be `"ShipPrinter"`:

```
! U1 getvar "ip.dhcp.option12_value"
```

ip.dhcp.option12_value

This command retrieves the actual value which will be used in the discovery packet of the wireless print server.

Getvar

To retrieve the actual value which will be used in the discovery packet of the wireless print server:

```
! U1 getvar "ip.dhcp.option12_value"
```

Example

This `setvar` example shows configuring the ip.dhcp.option12_format to the value contained in the device.friendly_name.

It is necessary to surround the SGD entry to be used as source for the data with the < and > characters.

```
! U1 setvar "ip.dhcp.option12_format" "<device.friendly_name>"
```

To further explain, if the above command was issued and the value currently stored in the device.friendly_name parameter was `"ShipPrinter"`, then the response to following command would be `"ShipPrinter"`:

```
! U1 getvar "ip.dhcp.option12_value"
```

ip.dhcp.request_timeout

This command retrieves the maximum amount of time (in seconds) for a DHCP discovery requests on the wireless print server.

Setvar

To set the amount of time (in seconds) to wait before timing out a DHCP discovery request:

```
! U1 setvar "ip.dhcp.request_timeout" "value"
```

Values

"2" through "30"

Default

"2"

Getvar

To retrieve the currently set the amount of time (in seconds) to wait before timing out a DHCP discovery request:

```
! U1 getvar "ip.dhcp.request_timeout"
```

Example

This `setvar` example shows the value set to "2".

```
!
U1 setvar "ip.dhcp.request_timeout" "2"
```

When the `setvar` value is set to "2", the `getvar` result is "2".

ip.dhcp.requests_per_session

This command retrieves the maximum amount of DHCP discovery requests for a single DHCP session on the wireless print server.

Setvar

To set the maximum amount of DHCP discovery requests for a single DHCP session on the wireless print server:

```
! U1 setvar "ip.dhcp.requests_per_session" "value"
```

Values

"1" through "10"

Default

"2"

Getvar

To retrieve the currently set maximum amount of DHCP discovery requests for a single DHCP session on the wireless print server:

```
! U1 getvar "ip.dhcp.requests_per_session"
```

Example

This `setvar` example shows the value set to "2".

```
! U1 setvar "ip.dhcp.requests_per_session" "2"
```

When the `setvar` value is set to "2", the `getvar` result is "2".

ip.dhcp.session_interval

This command configures the time interval (in seconds) before a new DHCP session is started on the wireless print server.

Setvar

To set the DHCP session time out value (in seconds):

```
! U1 setvar "ip.dhcp.session_interval" "value"
```

Values

`"0"` through `"60"`

Default

`"10"`

Getvar

To retrieve the current DHCP session time out value (in seconds):

```
! U1 getvar "ip.dhcp.session_interval"
```

Example

This `setvar` example shows the value set to `"10"`.

```
! U1 setvar "ip.dhcp.session_interval" "10"
```

When the `setvar` value is set to `"10"`, the `getvar` result is `"10"`.

ip.dhcp.user_class_id

This command sets or retrieves the current DHCP class ID setting.

Setvar

To set the current DHCP class ID setting:

```
! U1 setvar "ip.dhcp.user_class_id" "value"
```

Values

any class ID string,

Default

""

Getvar

To retrieve the current DHCP class ID setting:

```
! U1 getvar "ip.dhcp.user_class_id"
```

Result

The current DHCP class ID setting.

ip.dhcp.vendor_class_id

This command configures the DHCP vendor class ID setting.

Setvar

To set the DHCP vendor class ID setting:

```
! U1 setvar "ip.dhcp.vendor_class_id" "value"
```

Values

`Max string length of 64.`

This command builds a string using the following values: `(device.company_name) (device.product_name)-(head.resolution.in_dpi) (device.pnp_option)`

Default

The default varies by printer.

Getvar

To retrieve the current DHCP vendor class ID setting:

```
! U1 getvar "ip.dhcp.vendor_class_id"
```

ip.dns.domain

This command identifies the network domain of the wireless print server.

Getvar

To retrieve the network domain of the wireless print server:

```
! U1 getvar "ip.dns.domain"
```

Example

In this example, the `getvar` retrieves the network domain of the wireless print server.

```
! U1 getvar "ip.dns.domain"
```

ip.dns.servers

This command retrieves a space delimited list of the domain name servers from a wireless print server.

Getvar

To retrieve a list of space delimited DNS wireless print servers

```
! U1 getvar "ip.dns.servers"
```

Example

In this example, the `getvar` retrieves a list of space delimited DNS wireless print servers.

```
! U1 getvar "ip.dns.servers"
```

ip.firewall.authentication.add

This command allows a user to add a single server/username/password triplet into the list of authentication entries. This authentication entry is applied before making an outgoing HTTP/HTTPS connection in case the printer must go through an authentication server beforehand. This setting is separate from the proxy setting.

The server, username, and password parameters are separated by a space, not a tab or other white space character. The server name is required. If no username is supplied, but a password is, there must be two spaces between the server and the password fields. If there is a username but no password, or simply just the server name, no space is required at the end of the entry. Both DNS names and IP addresses are acceptable for the server name.

Setvar

To add a server, username, and password to the list of authentication entries:

```
U1 setvar "ip.firewall.authentication.add" "servername[ username][ password]"
```

Values

- `servername` is required. DNS or IP address is acceptable.
- `username` is an optional value. When a username is provided, it must be separated from the server name by a space.
- `password` is an optional value. When a password is provided, it must be separated from the username by a space. If the username is omitted, the password must be separated from the servername by two spaces.

The maximum length of the authentication entry is up to 1024 characters.

ip.firewall.authentication.entries

This command retrieves a list of the server names added to the authentication entries list. The list includes only the server names, the usernames and passwords are not shown.

Getvar

To have the printer return a list of server names on the authentication entries list:

```
! U1 getvar "ip.firewall.authentication.entries"
```

Result

The list of server names on the authentication entries list. A carriage return line feed delimits server names.

ip.firewall.authentication.remove

This command allows a user to remove a single server/username/password triplet from the list of authentication entries. Only the servername is required to remove a complete entry. If an invalid entry is supplied, no action is taken.

Setvar

To remove an entry from the list of authentication entries:

```
! U1 setvar "ip.firewall.authentication.remove" "value"
```

Value

`"value"` is the server name. A DNS name or IP address is acceptable.

ip.firewall.proxy

Use this command when a printer must go through a proxy server before making an outgoing HTTP/HTTPS connection. This setting is not connection specific and acts as a general value.

Setvar

To specify a proxy server:

```
! U1 setvar "ip.firewall.proxy" "[http|https]://[user:password@]host[:port][/
path]"
```

Values

- `HTTP` or `HTTPS` is a required value.
- `user:password` is an optional value.
- `host` is a required value. DNS or IP address is acceptable.
- `:port` is an optional value
- `/path` is an optional value.

The maximum length of the string is 2048 characters.

Getvar

To have the printer return the value of the proxy server setting:

```
 U1 getvar "ip.firewall.proxy"
```

If the proxy value includes a `user:password`, the are replaced with a `*`.

The maximum length of the string is 2048 characters.

ip.firewall.whitelist_in

Returns a comma-separated list of IP addresses and/or IP address ranges that will be allowed to communicate with the printer. If the list is empty then the firewall will be disabled.

Setvar

To set the command:

```
! U1 setvar "ip.firewall.whitelist_in" "value"
```

Value

The value is a string of up to 256 alphanumeric characters.

Default

```
""
```

Examples

Single IP address:

```
! U1 setvar "ip.firewall.whitelist_in" "192.168.1.20"
```

Multiple IP addresses:

```
! U1 setvar "ip.firewall.whitelist_in" "192.168.1.20,192.168.100.21"
```

IP address ranges:

```
! U1 setvar "ip.firewall.whitelist_in" "192.168.1.20-192.168.1.100"
```

IP ranges and Single/Multiple IPs

```
"ip.firewall.whitelist_in" "192.168.1.20-192.168.1.40, 192.168.1.50,
 192.168.1.75"
```

Getvar

To have the printer return the current setting value:

```
! U1 getvar "ip.firewall.whitelist_in"
```

ip.ftp.enable

This printer setting refers to the FTP protocol setting. This command tells the printer to turn FTP on or off.

Setvar

To turn FTP on or off:

```
! U1 setvar "ip.ftp.enable" "value"
```

Values

`"off"` disables FTP

`"on"` enables FTP

Default

`"on"`

Getvar

To respond with the FTP status:

```
! U1 getvar "ip.ftp.enable"
```

Example

This `setvar` example shows the FTP status set to `"on"`.

```
! U1 setvar "ip.ftp.enable" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is that the FTP status is `"on"`.

ip.ftp.execute_file

This command setting controls the ability of the printer to process or not process commands received via the FTP protocol using the printers ZPL engine.

Setvar

To turn FTP processing ability on or off:

```
! U1 setvar "ip.ftp.execute_file" "value"
```

Values

`"off"` disables the printer's ability to process FTP commands

`"on"` enables the printer's ability to process FTP commands

Default

`"on"`

Getvar

To respond with the FTP processing ability status:

```
! U1 getvar "ip.ftp.execute_file"
```

Example

This `setvar` example shows the FTP processing ability set to `"on"`.

```
! U1 setvar "ip.ftp.execute_file" "on"
```

ip.ftp.request_password

This command controls whether the ftp client will prompt the user to enter a password at the beginning of an ftp session.

Setvar

To turn on or off the ftp session password request:

```
! U1 setvar "ip.ftp.request_password" "value"
```

Values

"no"

"yes"

Default

"no"

Getvar

To return whether the ftp client is set to prompt the user for a password at the start of an ftp session:

```
! U1 getvar "ip.ftp.request_password"
```

ip.gateway

This command instructs the printer to change the gateway address.

IMPORTANT: This setting refers to the gateway address. A set value is ignored if the IP protocol is not set to permanent.

Setvar

To change the gateway address:

```
! U1 setvar "ip.gateway" "value"
```

Values

Any valid gateway address.

Default

`"0.0.0.0"`

Getvar

To respond with the gateway address:

```
 ! U1 getvar "ip.gateway"
```

Example

This `setvar` example shows the value set to `"10.3.5.1"`.

```
! U1 setvar "ip.gateway" "10.3.5.1"
```

When the `setvar` value is set to `"10.3.5.1"`, the `getvar` result is `"10.3.5.1"`.

ip.http.admin_name

This command sets the name to be used for authentication on the print server web pages.

Setvar

To set the admin user name to the specified value:

```
! U1 setvar "ip.http.admin_name" "value"
```

Values

A string with a maximum of 25 characters.

Default

`"admin"`

Getvar

To respond with the admin user name:

```
! U1 getvar "ip.http.admin_name"
```

Example

This `setvar` example shows the value set to `"useradmin101"`.

```
! U1 setvar "ip.http.admin_name" "useradmin101"
```

ip.http.admin_password

This command sets the password to be used for authentication on the print server web pages.

Setvar

To set print server web page password:

```
! U1 setvar "ip.http.admin_password" "value"
```

Values

A string with a maximum of 25 characters.

Default

`"1234"`

Getvar

To respond with the print server web page password:

```
! U1 getvar "ip.http.admin_password"
```

Example

This `setvar` example shows the value set to `"P@ssword101"`.

```
! U1 setvar "ip.http.admin_password" "P@ssword101"
```

ip.http.custom_link_name

This command creates a custom link below the FAQ link on the print server settings page. Use `ip.http.custom_link_url` to define the URL for your link name.

Setvar

To set the custom link name:

```
! U1 setvar "ip.http.custom_link_name" "value"
```

Values

Any string, maximum of 25 characters.

Default

""

Getvar

To respond with the custom link name

```
! U1 getvar "ip.http.custom_link_name"
```

Example

This `setvar` example shows the value set to `"Click Here for Info"`.

```
! U1 setvar "ip.http.custom_link_name" "Click Here for Info"
```

ip.http.custom_link_url

This command creates a custom link on the printer web page for the printer's URL. Use `ip.http.custom_link_name` to define the text that will display for your link.

Setvar

To change the custom URL:

```
! U1 setvar "ip.http.custom_link_url" "value"
```

Values

A string of 64 characters or less.

Default

""

Getvar

To respond with the custom URL:

```
! U1 getvar "ip.http.custom_link_url"
```

Example

```
! U1 setvar "ip.http.custom_link_url" "http://www.zebra.com/sdk"
```

ip.http.enable

This printer setting refers to the HTTP protocol/web server setting.

Setvar

To change HTTP to on or off:

```
! U1 setvar "ip.http.enable" "value"
```

Values

`"off"` disables HTTP protocol

`"on"` enables HTTP protocol

Default

`"on"`

Getvar

To respond with the HTTP status:

```
! U1 getvar "ip.http.enable"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "ip.http.enable" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`.

ip.http.faq_url

This command sets the FAQ URL displayed on the printer web pages.

Setvar

To change the FAQ URL:

```
! U1 setvar "ip.http.faq_url" "value"
```

Values

A string of 64 characters or less.

Default

""

Getvar

To respond with the current FAQ URL:

```
! U1 getvar "ip.http.faq_url"
```

Example

```
! U1 getvar "ip.http.faq_url"
```

ip.http.port

This command sets the port number at which the printer web pages will be served.

Setvar

To change the port setting for the printer web pages:

```
! U1 setvar "ip.http.port" "value"
```

Values

`"0"` to `"65535"`

Default

`"80"`

Getvar

To respond with the current port setting for the printer web pages:

```
! U1 getvar "ip.http.port"
```

Example

This `setvar` example shows the value set to `"8080"`.

```
! U1 setvar "ip.http.enable" "8080"
```

ip.https.enable

Enables/disables the HTTPS web connections.

IMPORTANT: A network or printer reset is required for this setting to take effect.

Setvar

To set the command:

```
! U1 setvar "ip.https.enable" "value"
```

Values

- `"off"` disables HTTPS protocol
- `"on"` enables HTTPS protocol

Default

```
"on"
```

Getvar

To have the printer return the current setting value:

```
! U1 getvar "ip.https.enable"
```

ip.https.port

Sets the port that will listen for HTTPS connections.

Setvar

To set the command:

```
! U1 setvar "ip.https.port" "value"
```

Values

Any valid https port from 0-65535.

Default

```
"443"
```

Getvar

To have the printer return the current setting value:

```
! U1 getvar "ip.https.port"
```

ip.lpd.enable

This printer setting refers to the LPD (Line Printer Daemon) protocol setting.

IMPORTANT: LPD communications from the host should be directed to port 515.

Setvar

To turn LPD on or off:

```
! U1 setvar "ip.lpd.enable" "value"
```

Values

`"off"` disables LPD protocol

`"on"` enables LPD protocol

Default

`"on"`

Getvar

To respond with the LPD status:

```
! U1 getvar "ip.lpd.enable"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "ip.lpd.enable" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`.

ip.mac_raw

This command specifies the RAW MAC address of the wireless print server. The raw mac address is the mac address without the colons (":").

Getvar

To retrieve the RAW MAC address of the wireless print server:

```
! U1 getvar "ip.mac_raw"
```

Example

In this example, the `getvar` retrieves the RAW MAC address of the wireless print server.

```
! U1 getvar "ip.mac_raw"
```

ip.mirror.appl_path

This command overrides the default location of the mirror application path (`"appl"` in `ip.mirror.path`). If the SGD is empty, the default path is used. If an error occurs retrieving application from path (i.e. path not found or application not there), the default path is NOT used. This path must be fully defined in relation to the mirror server root and is not relative to `ip.mirror.path`.

Setvar

To set the path to the application on the mirror server:

```
! U1 setvar "ip.mirror.appl_path" "values"
```

Values

A valid application path (location).

Default

`""`

Getvar

To retrieve the path to the application on the mirror server:

```
! U1 getvar "ip.mirror.appl_path"
```

Example

If the default value is used, the application would be copied from `"ip.mirror.path"\appl`

If you send `! U1 setvar "ip.mirror.appl_path" "program\current"` and `"ip.mirror.path"` has the value `"c:\mirror"`, then the application would be copied from `c:\mirror\program\current`.

ip.mirror.auto

This command enables the ability to automatically perform a mirror update (fetch) command on power up.

Setvar

To perform a mirror update (fetch) command when the printer is turned on using the interval that is set for `"ip.mirror.freq"` or `"ip.mirror.freq_hours":`

```
! U1 setvar "ip.mirror.auto" "values"
```

Values

- `"on"` turns on the auto mirroring feature
- `"off"` turns off the auto mirroring feature

Default

`"off"`

Getvar

To report whether the printer will perform a mirror update (fetch) automatically on power up:

```
! U1 getvar "ip.mirror.auto"
```

Example

This `setvar` example shows the value set to `"off"`.

```
! U1 setvar "ip.mirror.auto" "off"
```

When the `setvar` value is set to `"off"`, the `getvar` result is `"off"`.

ip.mirror.error_retry

This command identifies how many times mirroring is retried when an error occurs.

Setvar

To set the required times that mirroring retries when an error occurs:

```
! U1 setvar "ip.mirror.error_retry" "value"
```

Getvar

To return the current setting value:

```
! U1 getvar "ip.mirror.error_retry"
```

Example

This `setvar` example shows the value set to `"0"`.

```
! U1 setvar "ip.mirror.error_retry" "0"
```

When the `setvar` value is set to `"0"`, the `getvar` result is `"0"`.

ip.mirror.feedback.auto

This command identifies if a feedback file is pushed to the mirroring server by the printer when a mirroring update (fetch) is complete.

Setvar

To set the mirror feedback feature to on or off:

```
! U1 setvar "ip.mirror.feedback.auto" "value"
```

Values

"on" turns on mirror feedback

"off" turns off mirror feedback

Default

"off"

Getvar

To return the current setting value:

```
! U1 getvar "ip.mirror.feedback.auto"
```

Example

This setvar example shows the value set to "off".

```
! U1 setvar "ip.mirror.feedback.auto" "off"
```

When the setvar value is set to "off", the getvar result is "off".

ip.mirror.feedback.freq

This command specifies the time interval (in minutes) between performing feedback file uploads.

Setvar

To set the number of minutes to wait between feedback file uploads:

```
! U1 setvar "ip.mirror.feedback.freq" "value"
```

Values

A numeric value (0 - 65535)

Default

`"0"`

Getvar

To retrieve the number of minutes set to wait between feedback file uploads:

```
! U1 getvar "ip.mirror.feedback.freq"
```

Example

This `setvar` example shows the value set to `"0"`.

```
! U1 setvar "ip.mirror.feedback.freq" "0"
```

When the `setvar` value is set to `"0"`, the `getvar` result is `"0"`.

ip.mirror.feedback.odometer

This command instructs the printer to set the mirror feedback odometer.

Setvar

To set the odometer counter:

```
! U1 setvar "ip.mirror.feedback.odometer" "values"
```

Values

A numeric value between `"0"` and `"4294967295"`.

Default

`"0"`

Getvar

To retrieve the mirror feedback odometer value:

```
! U1 getvar "ip.mirror.feedback.odometer"
```

Example

This `setvar` example shows the value set to `"0"`.

```
! U1 setvar "ip.mirror.feedback.odometer" "0"
```

When the `setvar` value is set to `"0"`, the `getvar` result is `"0"`.

ip.mirror.feedback.path

This command identifies where the feedback file is stored on the mirroring server.

Setvar

To set the path on the mirroring server that stores the feedback file:

```
! U1 setvar "ip.mirror.feedback.path" "value"
```

Values

Alphanumeric text (1 to 50 characters)

Default

`"Zebra/feedback"`

Getvar

To retrieve the path on the mirroring sever that the printer is currently using to store the feedback file:

```
! U1 getvar "ip.mirror.feedback.path"
```

Example

This `setvar` example shows the value set to `"Zebra/feedback"`.

```
! U1 setvar "ip.mirror.feedback.path" "Zebra/feedback"
```

When the `setvar` value is set to `"Zebra/feedback"`, the `getvar` result is `"Zebra/feedback"`.

ip.mirror.fetch

This command forces a mirroring update sequence.

Setvar

To force a mirroring update sequence:

```
! U1 setvar "ip.mirror.fetch" ""
```

Do

To force a mirroring update sequence:

```
! U1 do "ip.mirror.fetch" ""
```

Example

This `setvar` example forces a mirroring update sequence.

```
! U1 setvar "ip.mirror.fetch" ""
```

ip.mirror.freq

This command defines the frequency of mirroring updates (in minutes).

Setvar

To set the mirror frequency:

```
! U1 setvar "ip.mirror.freq" "value"
```

Values

"0" through "65535" (minutes)

Default

"0" (disables this feature)

IMPORTANT: When the "ip.mirror.freq" is set to a low value (other than zero) the printer will spend a lot of time performing the mirroring process.

Getvar

To retrieve the number of minutes to wait before performing another mirror update:

```
! U1 getvar "ip.mirror.freq"
```

Example

This setvar example shows the value set to "1000".

```
! U1 setvar "ip.mirror.freq" "1000"
```

When the setvar value is set to "1000", the getvar result is "1000" and mirroring will be attempted every 1000 minutes.

ip.mirror.freq_hours

This command defines the frequency of mirroring updates (in hours).

Setvar

To set the frequency of mirroring updates (in hours):

```
! U1 setvar "ip.mirror.freq_hours" "values"
```

Values

`"0"` through `"100"`

Default

`"0"` (disables this feature)

Getvar

To retrieve the frequency of mirroring updates (in hours) that the printer is currently using:

```
! U1 getvar "ip.mirror.freq_hours"
```

Example

This `setvar` example shows the value set to `"8"`.

```
! U1 setvar "ip.mirror.freq_hours" "8"
```

When the `setvar` value is set to `"8"`, the `getvar` result is `"8"` and mirroring will be attempted every 8 hours.

ip.mirror.interface

Determines the interface over which Mirror will operate.

Setvar

To set the interface over which Mirror will operate:

```
! U1 setvar "ip.mirror.interface" "value"
```

Values

`"both"` internal or external wired and WLAN

`"wired"` internal or external wired

`"wireless"` WLAN

Default

`"both"`

Getvar

To return the current setting value:

```
! U1 getvar "ip.mirror.interface"
```

ip.mirror.last_error

This command retrieves the last error encountered during a mirroring operation.

Getvar

To retrieve the error code of the last mirroring update (fetch):

```
! U1 getvar "ip.mirror.last_error"
```

Example

In this example, the `getvar` retrieves the error code of the last mirroring update.

```
! U1 getvar "ip.mirror.last_error"
```

ip.mirror.last_time

This command retrieves the timestamp, in seconds, of the last time the system attempted a mirror update (fetch).

Getvar

To retrieve the timestamp, in seconds, of the last time the system attempted a mirror update (fetch):

```
! U1 getvar "ip.mirror.last_time"
```

Example

In this example, the `getvar` retrieves the timestamp, in seconds, of the last time the system attempted a mirror update.

```
! U1 getvar "ip.mirror.last_time"
```

ip.mirror.mode

This command sets the protocol used to perform mirror tasks.

Setvar

To sets the protocol for mirror tasks:

```
! U1 setvar "ip.mirror.mode" "values"
```

Values

`ftp` means the FTP protocol will be used to perform mirror tasks

`sftp` means the SFTP protocol will be used to perform mirror tasks

Getvar

To retrieve the path to the application on the mirror server:

```
! U1 getvar "ip.mirror.mode"
```

ip.mirror.password

This command specifies the user password on the mirroring server assigned for mirroring updates (fetch).

Setvar

To specify a password for mirroring updates (fetch):

```
! U1 setvar "ip.mirror.password" "value"
```

Values

Alphanumeric text string (1 to 20 characters)

Default

`"password"`

Getvar

To retrieve the user password the printer is currently using for mirroring updates (fetch):

```
! U1 getvar "ip.mirror.password"
```

Example

This `setvar` example shows the value set to `"password"`.

```
! U1 setvar "ip.mirror.password" "password"
```

When the `setvar` value is set to `"password"`, the `getvar` result is `"*"`. For security purposes, the printer does not return password information.

ip.mirror.path

This command identifies the base path on the FTP server where the mirror directory resides.

Setvar

To set the base path on the FTP server where the mirror directory resides:

```
! U1 setvar "ip.mirror.path" "value"
```

Values

Alphanumeric text string (1 to 50 characters)

Default

`"zebra"`

Getvar

To retrieve the base path of the FTP server where the mirror directory resides:

```
! U1 getvar "ip.mirror.path"
```

Example

This `setvar` example shows the value set to `"zebra"`.

```
! U1 setvar "ip.mirror.path" "zebra"
```

When the `setvar` value is set to `"zebra"`, the `getvar` result is `"zebra"`.

ip.mirror.reset_delay

This command specifies the number of seconds between when the printer receives the last byte of the last file from the `/commands` directory and when the printer resets during a mirror event.

Setvar

To set the number of seconds between when the printer receives the last byte of the last file from the `/commands` directory and when the printer resets during a mirror event:

```
! U1 setvar "ip.mirror.reset_delay" "value"
```

Values

`"0"` to `"900"` (seconds)

Default

`"5"`

NOTE: The default setting for the `ip.mirror.reset_delay` command is 5 seconds; in some cases it may be necessary to use a longer delay to allow for full processing of longer or more complex files.

Getvar

To retrieve the number of seconds between when the printer receives the last byte of the last file from the /commands directory and when the printer resets during a mirror event:

```
! U1 getvar "ip.mirror.reset_delay"
```

Example

This `setvar` example shows the value set to `"10"`.

```
! U1 setvar "ip.mirror.reset_delay" "10"
```

When the `setvar` value is set to `"10"`, the `getvar` result is `"10"`.

ip.mirror.server

This command identifies the IP address of the mirroring server.

Setvar

To set the IP address of the mirroring server:

```
! U1 setvar "ip.mirror.server" "value"
```

Values

A valid IP address

Default

`"127.0.0.1"`

Getvar

To retrieve the IP address of the mirroring server:

```
! U1 getvar "ip.mirror.server"
```

Example

This `setvar` example shows the value set to `"10.3.1.1"`.

```
! U1 setvar "ip.mirror.server" "10.3.1.1"
```

When the `setvar` value is set to `"10.3.1.1"`, the `getvar` result is `"10.3.1.1"`.

ip.mirror.success

This command reports the success or failure of the last mirroring update (fetch).

Getvar

To retrieve the success or failure of the last mirroring update (fetch):

```
! U1 getvar "ip.mirror.success"
```

Values

`"yes"` means successful

`"no"` means unsuccessful

Example

In this example, the `getvar` retrieves the success or failure of the last mirroring update (fetch).

```
! U1 getvar "ip.mirror.success"
```

ip.mirror.success_time

This command provides the timestamp, in seconds, of the last time the system successfully completed a mirror update (fetch).

Getvar

To retrieve the timestamp, in seconds, of the last time the system successfully completed a mirror update (fetch):

```
! U1 getvar "ip.mirror.success_time"
```

Example

In this example, the `getvar` retrieves the timestamp of the last time the system successfully completed a mirror update.

```
! U1 getvar "ip.mirror.success_time"
```

ip.mirror.username

This command specifies the user name on the mirroring server assigned for mirroring updates (fetch).

Setvar

To set a specific user name for mirroring updates (fetch):

```
! U1 setvar "ip.mirror.username" "value"
```

Values

Alphanumeric text string (1 to 20 characters)

Default

`"user"`

Getvar

To retrieve the user name the printer is currently using for mirroring updates (fetch):

```
! U1 getvar "ip.mirror.username"
```

Example

This `setvar` example shows the value set to `"user"`.

```
! U1 setvar "ip.mirror.username" "user"
```

When the `setvar` value is set to `"user"`, the `getvar` result is `"user"`.

ip.mirror.version

This command retrieves the mirror code build date.

Getvar

To retrieve the mirror code build date:

```
! U1 getvar "ip.mirror.version"
```

Example

In this example, the `getvar` retrieves the mirror code build date.

```
! U1 getvar "ip.mirror.version"
```

ip.netmask

This setting refers to the subnet mask address. This value is ignored if the IP protocol is not set to permanent.

Setvar

To instruct the printer to change the subnet mask:

```
! U1 setvar "ip.netmask" "value"
```

Values

Any valid subnet mask.

Default

`"255.255.255.0"`

Getvar

To respond with the subnet mask value:

```
! U1 getvar "ip.netmask"
```

Example

This `setvar` example shows the value set to `"255.255.255.0"`.

```
! U1 setvar "ip.netmask" "255.255.255.0"
```

When the `setvar` value is set to `"255.255.255.0"`, the `getvar` result is `"255.255.255.0"`.

ip.ntp.enable

Enables or disables the Network Time Protocol (NTP) feature.

Setvar

To enable or disable the Network Time Protocol (NTP) feature:

```
! U1 setvar "ip.ntp.enable" "value"
```

Values

`"on"` means enabled

`"off"` means disabled

Default

`"off"`

Getvar

To return the current setting value:

```
! U1 getvar "ip.ntp.enable"
```

ip.ntp.log

Retrieves the Network Time Protocol (NTP) status and error log. If NTP is not running this will return `"File Not Found"` for the error log.

Getvar

To retrieve the Network Time Protocol (NTP) status and error log:

```
! U1 getvar "ip.ntp.log"
```

Default

`"File Not Found"`

ip.ntp.servers

Sets the list of NTP (Network Time Protocol) servers which the printer will use to set the time.

Setvar

To set the list of NTP (Network Time Protocol) servers which the printer will use to set the time:

```
! U1 setvar "ip.ntp.servers" "value"
```

Values

A comma delimited string of server name(s) or ip address(es), with a length of 0-1024 characters.

Default

""

Example

```
! U1 setvar "ip.ntp.servers" "0.us.pool.ntp.org,10.3.17.124"
```

Getvar

To retrieve the current setting value:

```
! U1 getvar "ip.ntp.servers"
```

Result

A comma delimited string of server name(s) or ip address(es).

ip.ping_gateway_interval

Determines the interval in minutes at which to send ICMP PING packets to the default gateway.

Setvar

To set the command:

```
! U1 setvar "ip.ping_gateway_interval" "value"
```

Value

An integer from "0" to "30". "0" is disabled

Default

"0"

Getvar

To have the printer return the current setting value:

```
! U1 getvar "ip.ping_gateway_interval"
```

ip.pop3.enable

This printer setting determines if the printer queries a POP3 mailbox for mail.

Setvar

To turn POP3 on or off:

```
! U1 setvar "ip.pop3.enable" "value"
```

Values

`"off"` disables POP3

`"on"` enables POP3

Default

`"on"`

Getvar

To respond with the POP3 status:

```
! U1 getvar "ip.pop3.enable"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "ip.pop3.enable" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`.

ip.pop3.password

This printer setting refers to the POP3 mailbox password. This only applies if "`ip.pop3.enable`" is set to on.

Setvar

To change the POP3 password:

```
! U1 setvar "ip.pop3.password" "value"
```

Values

A maximum of 20 alphanumeric characters

Default

`" "`

Getvar

To respond with the POP3 password:

```
! U1 getvar "ip.pop3.password"
```

For protection a single `"*"` prints.

Example

This `setvar` example shows the value set to `"password"`.

```
! U1 setvar "ip.pop3.password" "password"
```

When the `setvar` value is set to `"password"`, the `getvar` result is `"*"`.

ip.pop3.poll

This printer setting refers to how frequent (in seconds) the printer queries a POP3 mailbox for new mail. This only applies if the "`ip.pop3.enable`" is set to on.

Setvar

To change the POP3 poll interval:

```
! U1 setvar "ip.pop3.poll" "value"
```

A value of "`0`" causes the printer to only query the POP3 mailbox one time, on printer power up, or following a network reset.

Values

"`0`" through "`65535`"

Default

```
"0"
```

NOTE: A poll value of less then thirty seconds is not recommended. The printer is unresponsive for several seconds when polling for email depending on data transfer time from the server to the printer.

Getvar

To respond with the POP3 poll frequency (in seconds):

```
! U1 getvar "ip.pop3.poll"
```

Example

This `setvar` example shows the value set to "`0`".

```
! U1 setvar "ip.pop3.poll" "0"
```

When the `setvar` value is set to "`0`", the `getvar` result is "`0`".

ip.pop3.print_body

Determines if the email body will be printed when the email is retrieved via POP3. This only applies if ip.pop3.enable is set to "on".

Setvar

To set the command:

```
! U1 setvar "ip.pop3.print_body" "value"
```

Value

- `"on"`
- `"off"`

Default

```
"off"
```

Getvar

To have the printer return the current setting value:

```
! U1 getvar "ip.pop3.print_body"
```

ip.pop3.print_headers

Determines if the email header will be printed when the email is retrieved via POP3. This only applies if ip.pop3.enable is set to "on".

Setvar

To set the command:

```
! U1 setvar "ip.pop3.print_headers" "value"
```

Values

- `"on"`
- `"off"`

Default

`"off"`

Getvar

To have the printer return the current setting value:

```
! U1 getvar "ip.pop3.print_headers"
```

ip.pop3.save_attachments

Determines if the email attachments are to be saved to the flash file system when the email is retrieved via POP3. This only applies if ip.pop3.enable is set to "on".

Setvar

To set the command:

```
! U1 setvar "ip.pop3.save_attachments" "value"
```

Values

- `"on"`
- `"off"`

Default

```
"on"
```

Getvar

To have the printer return the current setting value:

```
! U1 getvar "ip.pop3.save_attachments"
```

ip.pop3.server_addr

This printer setting refers to the POP3 server IP address that the printer contacts when checking for new mail. This only applies if "`ip.pop3.enable`" is set to on.

Setvar

To change the POP3 server address:

```
! U1 setvar "ip.pop3.server_addr" "value"
```

Values

Any valid POP3 server address

Default

`"0.0.0.0"`

Getvar

To respond with the POP3 server address:

```
! U1 getvar "ip.pop3.server_addr"
```

ip.pop3.username

This printer setting refers to the POP3 user name. This only applies if the "`ip.pop3.enable`" is set to on.

Setvar

To change the POP3 user name:

```
! U1 setvar "ip.pop3.username" "value"
```

Values

A maximum of 20 alphanumeric characters

Default

""

Getvar

To respond with the POP3 user name:

```
! U1 getvar "ip.pop3.username"
```

Example

This `setvar` example shows the value set to `"user"`.

```
! U1 setvar "ip.pop3.username" "user"
```

When the `setvar` value is set to `"user"`, the `getvar` result is `"user"`.

ip.pop3.verbose_headers

Instructs the printer to respond with teh POP3 verbose header value.

Setvar

To set the command:

```
! U1 setvar "ip.pop3.verbose_headers" "value"
```

Values

- `"on"`
- `"off"`

Default

```
"off"
```

Getvar

To have the printer return the current setting value:

```
! U1 getvar "ip.pop3.verbose_headers"
```

ip.port

This printer setting refers to the port number that the TCP print service is listening on. Normal TCP communications from the host should be directed to this port.

Setvar

To set the TCP/UDP port number:

```
! U1 setvar "ip.port" "value"
```

Values

"1" through "65535" (excluding any ports currently used by other services, such as 21, 23, 80, and 515).

Default

"9100"

Getvar

To respond with the TCP/UDP port number:

```
! U1 getvar "ip.port"
```

Example

This `setvar` example shows the value set to "9100".

```
! U1 setvar "ip.port" "9100"
```

When the `setvar` value is set to "9100", the `getvar` result is "9100".

ip.port_alternate

This command sets the port number of the alternate port.

NOTE: Print servers supporting this command will monitor both the primary port and the alternate port for connections at the same time.

Setvar

To set the alternate port for the print server:

```
! U1 setvar "ip.port_alternate" "value"
```

Values

Any valid TCP port address.

Default

`"9100"`

Getvar

To return the current alternate port setting:

```
! U1 getvar "ip.port_alternate"
```

Values

The current port setting.

Example

This `setvar` example shows the value set to `"6101"`.

```
U1 setvar "ip.port_alternate" "6101"
```

ip.port_json_config

This printer setting refers to the port number that the TCP print service is listening on for JSON configuration packets. JSON TCP communications from the host should be directed to this port.

Setvar

To set the JSON TCP port number:

```
! U1 setvar "ip.port_json_config" "value"
```

Values

- `"0"` disables the port
- `"1"` through `"65535"` (excluding any ports currently used by other services, such as 21, 23, 80, and 515).

Default

`"9200"`

Getvar

To respond with the JSON TCP port number:

```
! U1 getvar "ip.port_json_config"
```

Example

This `setvar` example shows the value set to `"9200"`.

```
! U1 setvar "ip.port_json_config" "9200"
```

When the `setvar` value is set to `"9200"`, the `getvar` result is `"9200"`.

ip.port_single_conn

This command configures the port number for the single connection IP port. The single-connection IP port allows only one connection at a time.

See `"ip.port_single_conn_idle_timeout"` for additional information.

Setvar

To configure the port number for the single connection IP port:

```
! U1 setvar "ip.port_single_conn" "value"
```

Values

A number between `"1"` and `"65535"`. This number specifies the port.

Default

`"9300"`

Getvar

To return the current setting value:

```
! U1 getvar "ip.port_single_conn"
```

Result

A port number between 1 and 65535.

ip.port_single_conn_idle_timeout

Sets the amount of time that must elapse with no activity for the single-connection IP port to be considered in the idle state.

When the port is idle and a new connection is requested, the currently open connection will be closed and a new connection will be opened. If the port is not in the idle state, the current connection will be maintained and the connection request will be refused with an error response.

Setvar

To set the single connection timeout time:

```
! U1 setvar "ip.port_single_conn_idle_timeout" "value"
```

Values

A number between `"1"` and `"65535"`. The value is in seconds. If the value is `"0"`, the port will be considered to be in the idle state.

Default

`"180"`

Getvar

To return the current setting value:

```
! U1 getvar "ip.port_single_conn_idle_timeout"
```

ip.primary_network

This command allows you to set the primary network to either wired or wireless.

Setvar

To set the primary network device:

```
! U1 setvar "ip.primary_network" "value"
```

Values

`"1"` means wired

`"2"` means wireless

Default

`"1"`

Getvar

To respond with the name of the current primary network device:

```
! U1 getvar "ip.primary_network"
```

Example

This `setvar` example shows the value set to `"1"`.

```
! U1 setvar "ip.primary_network" "1"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"1"`.

ip.smtp.domain

This printer setting refers to the domain name used by the printer in sending email with respect to the SMTP server.

Setvar

To change the SMTP domain name:

```
! U1 setvar "ip.smtp.domain" "value"
```

Values

A maximum of 24 alphanumeric characters.

Default

`"ZBRPrintServer"`

Getvar

To return the SMTP domain name:

```
! U1 getvar "ip.smtp.domain"
```

Example

This `setvar` example shows the value set to `"ZBRPrintServer.com"`.

```
! U1 setvar "ip.smtp.domain" "ZBRPrintServer.com"
```

When the `setvar` value is set to `"ZBRPrintServer.com"`, the `getvar` result is `"ZBRPrintServer.com"`.

ip.smtp.enable

This printer setting refers to the SMTP protocol.

Setvar

To turn SMTP on or off:

```
! U1 setvar "ip.smtp.enable" "value"
```

Values

`"off"` disables SMTP

`"on"` enables SMTP

Default

`"on"`

Getvar

To return the SMTP status:

```
! U1 getvar "ip.smtp.enable"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "ip.smtp.enable" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`.

ip.smtp.server_addr

This printer setting refers to the IP address of the SMTP server used for sending email.

Setvar

To change the SMTP server address:

```
! U1 setvar "ip.smtp.server_addr" "value"
```

Values

Any valid IP address.

Default

`0.0.0.0`

Getvar

To respond with the current SMTP server address:

```
! U1 getvar "ip.smtp.server_addr"
```

Example

This `setvar` example shows the value set to `10.10.10.10`.

```
! U1 setvar "ip.smtp.server_addr" "10.10.10.10"
```

When the `setvar` value is set to `"10.10.10.10"`, the `getvar` result is `"10.10.10.10"`.

ip.snmp.enable

This printer setting refers to the SNMP protocol.

Setvar

To enable or disable the SNMP protocol:

```
! U1 setvar "ip.snmp.enable" "value"
```

Values

`"on"` enables the SNMP protocol

`"off"` disables the SNMP protocol

Default

`"on"`

Getvar

To respond with the SNMP status:

```
! U1 getvar "ip.snmp.enable"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "ip.snmp.enable" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`.

ip.snmp.get_community_name

This printer setting is used when making SNMP queries. The SNMP client must supply the get community name that matches the printer's get community name in order to query any SNMP data.

Setvar

To set the SNMP get community name string:

```
! U1 setvar "ip.snmp.get_community_name" "value"
```

Values

A maximum of 19 alphanumeric characters.

Default

`"public"`

Getvar

To get the SNMP get community name string:

```
! U1 getvar "ip.snmp.get_community_name"
```

For protection a single "*" prints.

Example

This `setvar` example shows the value set to `"public"`.

```
! U1 setvar "ip.snmp.get_community_name" "public"
```

When the `setvar` value is set to `"public"`, the `getvar` result is `"*"`.

ip.snmp.set_community_name

This printer setting is used when changing SNMP data remotely. To alter any SNMP data, the SNMP client must supply the set community name that matches the printer's set community name.

Setvar

To set the SNMP set community name string:

```
! U1 setvar "ip.snmp.set_community_name" "value"
```

Values

A maximum of 19 alphanumeric characters

Default

`"public"`

Getvar

To return the printer's SNMP set community name string:

```
! U1 getvar "ip.snmp.set_community_name"
```

For protection a single `"*"` returns.

Example

This `setvar` example shows the value set to `"public"`.

```
! U1 setvar "ip.snmp.set_community_name" "public"
```

When the `setvar` value is set to `"public"`, the `getvar` result is `"*"`.

ip.snmp.trap_community_name

This command sets the SNMP Trap Community name of the print server.

Setvar

To set the SNMP trap community name:

```
! U1 setvar "ip.snmp.get_community_name" "value"
```

Values

A maximum of 20 alphanumeric characters.

Default

`"public"`

Getvar

To get the SNMP trap community name:

```
! U1 getvar "ip.snmp.trap_community_name"
```

Example

```
! U1 setvar "ip.snmp.trap_community_name" "public"
```

ip.snmpv3.admin.auth_protocol

This command returns the assigned authentication protocol to be used for the SNMPv3 admin user. The value with be either"MD5"or"SHA".

Getvar

To retrieve the assigned authentication protocol:

```
! U1 getvar "ip.snmpv3.admin.auth_protocol"
```

ip.snmpv3.admin.name

This command returns the assigned username for the SNMPv3 admin user. By default, the value is an empty string.

Getvar

To retrieve the assigned admin usesrname:

```
! U1 getvar "ip.snmpv3.admin.name"
```

ip.snmpv3.admin.priv_protocol

This command returns the assigned privacy protocol to be used for the SNMPv3 admin user. It will report either"AES"or"DES".

Getvar

To retrieve the assigned privacy protocol:

```
! U1 getvar "ip.snmpv3.admin.priv_protocol"
```

ip.snmpv3.enable

This command returns the status of SNMPv3. If it is enabled, the command returns `"on"`. If SNMPv3 is disabled, the command returns `"off"`.

Getvar

To view if SNMPv3 status is enabled:

```
! U1 getvar "ip.snmpv3.enable"
```

ip.snmpv3.monitor.auth_protocol

This command returns the assigned authentication protocol to be used for the SNMPv3 monitor user. It will report either "MD5" or "SHA".

Getvar

To retrieve the assigned authentication protocol:

```
! U1 getvar "ip.snmpv3.monitor.auth_protocol"
```

ip.snmpv3.monitor.name

This command returns the assigned username for the SNMPv3 monitor user (a user with read-only access). The value is a string with up to 32 characters. By default, the value is an empty string.

Getvar

To retrieve the assigned username:

```
! U1 getvar "ip.snmpv3.monitor.name"
```

ip.snmpv3.monitor.priv_protocol

This command returns the value of the privacy protocol to be used for the SNMPv3 monitor user. It will report a value of either "AES" or "DES".

Getvar

To retrieve the privacy protocol value:

```
! U1 getvar "ip.snmpv3.monitor.priv_protocol"
```

ip.tcp.enable

This printer setting refers to the TCP socket protocol.

Setvar

To turn the TCP on or off:

```
! U1 setvar "ip.tcp.enable" "value"
```

Values

`"off"` disables TCP protocol

`"on"` enables TCP protocol

Default

"on"

Getvar

To respond with the TCP status:

```
! U1 getvar "ip.tcp.enable"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "ip.tcp.enable" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`.

ip.tcp.nagle_algorithm

Enables or disables the use of the Nagle algorithm on TCP connections.

Setvar

To enables or disable the use of the Nagle algorithm on TCP connections:

```
! U1 setvar "ip.tcp.nagle_algorithm" "value"
```

Values

`"enabled"` allows the use of the Nagle algorithm

`"disabled"` disables use of the Nagle algorithm

Default

`"enabled"`

Getvar

To return the current setting value:

```
! U1 getvar "ip.tcp.nagle_algorithm"
```

ip.tls.enable

Enables the TLS connections to the encrypted channels on the printer specified by `ip.tls.port` and `ip.tls.port_json_config.`

Setvar

To set the command:

```
! U1 setvar "ip.tls.enable" "value"
```

Values

- `"on"`
- `"off"`

Default

`"on"`

Getvar

To have the printer return the current setting value:

```
! U1 getvar "ip.tls.enable"
```

Example

This example disables the TLS port.

```
! U1 setvar "ip.tls.enable" "off"
```

ip.tls.port

Sets the port number to which encrypted connections can be made to communicate with the printing channel.

Setvar

To set the command:

```
! U1 setvar "ip.tls.port" "value"
```

Values

Any valid https port from 0-65535.

Default

```
"9143"
```

Getvar

To have the printer return the current setting value:

```
! U1 getvar "ip.tls.port"
```

ip.tls.port_json_config

Sets the port number used to create an encrypted connection to the JSON configuration channel.

Setvar

To set the command:

```
! U1 setvar "ip.tls.port_json_config" "value"
```

Values

Any valid HTTPS port from 0-65535.

Default

```
"9243"
```

Getvar

To have the printer return the current setting value:

```
! U1 getvar "ip.tls.port_json_config"
```

ip.udp.enable

This printer setting refers to the UDP socket protocol.

Setvar

To turn UDP on or off:

```
! U1 setvar "ip.udp.enable" "value"
```

Values

`"off"` disables UDP protocol

`"on"` enables UDP protocol

Default

`"off"`

Getvar

To respond with the UDP status:

```
! U1 getvar "ip.udp.enable"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "ip.udp.enable" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`.

weblink.cloud_connect.enable

Enables the CloudConnect menu feature.

Setvar

To set the command:

```
! U1 setvar "weblink.cloud_connect.enable" "value"
```

Values

- `"on"` enables the CloudConnect menu
- `"off"` disables the CloudConnect menu

Default

`"on"`

Getvar

To have the printer return the current setting value:

```
! U1 getvar "weblink.cloud_connect.enable"
```

weblink.enable

This command indicates if one or more of the weblink connections are active.

If there is more than one connection under the weblink branch (for example, `weblink.ip.conn1`) and if any of the `.location` values are set, then this SGD will be set to `"on"`. If all connections are disabled (all connection `.location` values set to `""`), then this value will be set to `"off"`.

`^JUF, ^JUS, ^JUN, ^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Getvar

To indicate if one or more of the weblink connections are active:

```
! U1 getvar "weblink.enable"
```

Result

- `"yes"` if any of the location values are set
- `"off"` if all connections are disabled

weblink.ip.conn[1|2].authentication.add

This command allows the user to add a single server/username/password triplet into the list of authentication entries.

When the printer attempts to connect to the server (url specified in `weblink.ip.conn[1|2].location`) the server may require HTTP authentication (e.g. digest, basic, DNS, etc.). There may be multiple authentication requests along the route to the destination (e.g. a locals erver first requires HTTP authentication as well as on the remote server).

For each HTTP authentication request received while attempting to connect, the printer will enumerate the authentication entries and attempt to satisfy the request with the username/password pair provided for the respective server. The server name in the entry is what determines which username/password pair should be used for which authentication request. Both DNS names and IP addresses are acceptable. The server, username, and password are separated by a single space (not a tab or other white space character). The server name is the only required field. If no username is supplied, but a password is, there must be two spaces between the server and the password fields. If there is a username but no password, or simply just the servername, no space is required at the end of the entry.

If the command is changed when the connection is enabled (`weblink.enable is set to "on"`), it will not take effect until the connection is disabled, and then re-enabled.

NOTE: This setting only be changed when `weblink.enable` is set to `"off"`.

`^JUF, ^JUS, ^JUN, ^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Setvar

To add a single server/username/password triplet to the list of authentication entries:

```
! U1 setvar "weblink.ip.conn1.authentication.add" "servername[ username]
[ password]"
```

```
! U1 setvar "weblink.ip.conn2.authentication.add" "servername[ username]
[ password]"
```

Values

`"servername [username][ password]"` has a maximum length of 2048 characters

Default

NA

Example

In this example, a username and a password is supplied:

```
! U1 setvar "weblink.ip.conn1.authentication.add" "my.server.lan johndoe
 password"
```

In this example, no password is supplied

```
! U1 setvar "weblink.ip.conn1.authentication.add" "my.server.lan johndoe"
```

In this example, no username is supplied (note the double space)

```
! U1 setvar "weblink.ip.conn1.authentication.add" "my.server.lan password"
```

In this example, no username or password is supplied

```
! U1 setvar "weblink.ip.conn1.authentication.add" "my.server.lan"
```

weblink.ip.conn[1|2].authentication.entries

This command lists the server names added to the authentication entries list.

Only the server names will be shown. The username and passwords will not be shown.The server names are separated by a `\r\n` so that each shows up on its own line.

`^JUF, ^JUS, ^JUN, ^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Getvar

To list the server names for the specified connection:

```
! U1 getvar "weblink.ip.conn1.authentication.entries"
```

```
! U1 getvar "weblink.ip.conn2.authentication.entries"
```

weblink.ip.conn[1|2].authentication.remove

This command allows the user to remove a single server/username/password triplet from the list of authentication entries.

To remove an entry only the server name is supplied. If an invalid entry is supplied no action is taken. If the SGD is changed when the connection is enabled (`weblink.ip.conn[1|2].enable`), it will not take effect until the connection is disabled, and then re-enabled. It is therefore recommended that this setting only be changed when `weblink.ip.conn[1|2].enable` is set to `"off"`.

`^JUF, ^JUS, ^JUN, ^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Setvar

To remove a single server/username/password triplet from the list of authenticaiton entries:

```
! U1 setvar "weblink.ip.conn1.authentication.remove" "servername"
```

```
! U1 setvar "weblink.ip.conn2.authentication.remove" "servername"
```

Values

`"servername"` has a maximum length of string is 2048 characters.

Default

NA

Example

In this example, a username and a password is supplied

```
! U1 setvar "weblink.ip.conn1.authentication.remove" "my.server.lan"
```

weblink.ip.conn[1|2].location

This command assigns the URL of the server for this connection. The URL must follow the URL rules for the HTTP[S] protocol outlined in RFC2396 (http://www.ietf.org/rfc/rfc2396.txt).

The setting will not take effect until the printer is reset. Changing this setting will set weblink.printer_reset_required to "yes".

^JUF, ^JUS, ^JUN, ^JUA, and device.restore_defaults do not have any affect on this setting.

Setvar

To set the server URL for the specified connection:

```
! U1 setvar "weblink.ip.conn1.location" "value"
```

```
! U1 setvar "weblink.ip.conn2.location" "value"
```

Values

Any HTTPS URL up to 2048 characters

Default

""

Getvar

To return the server URL currently assigned to the connection:

```
! U1 getvar "weblink.ip.conn1.location"
```

```
! U1 getvar "weblink.ip.conn2.location"
```

Example

```
! U1 setvar "weblink.ip.conn2.location"
```

```
"https://my.linkos.server.com:8080/link/os"
```

weblink.ip.conn[1|2].num_connections

This command reports the number of established connections on conn1 and conn2. Once the connection is established, this number will be incremented. When a channel is closed or the connection times out, the number is decremented.

Getvar

To retrieve the number of active connections on conn1 and conn2:

```
! U1 getvar "weblink.ip.conn1.num_connections"
```

```
! U1 getvar "weblink.ip.conn2.num_connections"
```

weblink.ip.conn[1|2].maximum_simultaneous_connections

This command indicates the maximum number of simultaneous connections that can be initiated by the printer.

Via the main connection (the original connection initiated by the printer to the remote server), the remote server can request that additional connetions from the printer be initated (e.g. a connection that supports only JSON SGDs, one that behaves similar to the RAW TCP port.

The server is free to request as many as it thinks it needs, but the printer will prevent more than N number of connections, where N is the value of this command.

`^JUF, ^JUS, ^JUN, ^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Setvar

To set the maximum number of connections:

```
! U1 setvar "weblink.ip.conn1.maximum_simultaneous_connections" "value"
```

```
! U1 setvar "weblink.ip.conn2.maximum_simultaneous_connections" "value"
```

Values

Any integer from 1-100

Default

`"10"`

Getvar

To retrieve the maximum set number of connections:

```
! U1 getvar "weblink.ip.conn1.maximum_simultaneous_connections"
```

```
! U1 getvar "weblink.ip.conn2.maximum_simultaneous_connections"
```

Example

This example sets the conn1 maximum connections to 3.

```
! U1 setvar "weblink.ip.conn1.maximum_simultaneous_connections" "3"
```

weblink.ip.conn[1|2].proxy

This command assigns the URL of the proxy for the connection.

The proxy server protocol, port, domain, username, and password are all encoded into the URL via the format outlined in RFC2396 (http://www.ietf.org/rfc/rfc2396.txt).

The username and password must avoid the invalid characters listed in RFC2396 (e.g. ':', '@', '/', etc). If an invalid character must be used it needs to be escaped using '%' as described in RFC2396.

`^JUF, ^JUS, ^JUN, ^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Setvar

To assign the URL of the connection proxy:

```
! U1 setvar "weblink.ip.conn1.proxy" "url"
```

```
! U1 setvar "weblink.ip.conn2.proxy" "url"
```

Values

- Any valid URL up to 2048 characters
- Expected URL format: [http|https]://[user:pass@]domain[:port]/[path]
- The URL will need to be built according to the server/proxy environment the printer is running within.

Default

""

- The user:pass, port, and path are all optional.
- The default scheme must be either HTTPS or HTTP. The default is HTTP.
- The default port is 1080.
- The default is to omit the username and password.

Getvar

To retrieve the URL of the connection proxy:

```
! U1 getvar "weblink.ip.conn1.proxy"
```

```
! U1 getvar "weblink.ip.conn2.proxy"
```

Do

To assign the URL of the connection proxy:

```
! U1 do "weblink.ip.conn1.proxy" "url"
```

```
! U1 do "weblink.ip.conn2.proxy" "url"
```

Values

- Any valid URL up to 2048 characters
- Expected URL format: [http|https]://[user:pass@]domain[:port]/[path]
- The URL will need to be built according to the server/proxy environment the printer is running within.

Default

""

- The user:pass, port, and path are all optional.
- The default scheme must be either HTTPS or HTTP. The default is HTTP.
- The default port is 1080.
- The default is to omit the username and password.

Example

Examples of how to connect to various proxy servers:

```
http://username:password@mydomain.com:3128/
```

```
http://mydomain.com/
```

weblink.ip.conn[1|2].retry_interval

This command sets the number of seconds between attempts to connect to the server URL provided in `weblink.ip.conn1.location`. If an attempt is unsuccessful or the connection is lost, the printer will wait `'retry_interval'` seconds before attempting to connect again.

`^JUF, ^JUS, ^JUN, ^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Setvar

To set the number of seconds to wait before attempting to reconnect to the server:

```
! U1 setvar "weblink.ip.conn1.retry_interval" "value"
```

```
! U1 setvar "weblink.ip.conn2.retry_interval" "value"
```

Values

`"1"` through `"600"`

Default

`"10"`

Getvar

To return the number of seconds to wait between connection attempts:

```
! U1 getvar "weblink.ip.conn1.retry_interval"
```

```
! U1 getvar "weblink.ip.conn2.retry_interval"
```

Do

To set the number of seconds to wait before attempting to reconnect to the server:

```
! U1 do "weblink.ip.conn1.retry_interval" "value"
```

```
! U1 do "weblink.ip.conn2.retry_interval" "value"
```

Values

1 - 600

Default

`"10"`

weblink.ip.conn1.test.location

This command holds the URL for testing a connection to the internet. This is meant to assist users in debugging their printer's connection to remote servers when there are issues with the main weblink connection (conn1 or conn2).

The URL must follow the URL rules for the HTTP[S] protocol outlined in RFC2396 (http://www.ietf.org/rfc/rfc2396.txt).

`^JUF, ^JUS, ^JUN, ^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Setvar

To set the URL to hold for testing a connection:

```
! U1 setvar "weblink.ip.conn1.test.location" "url"
```

```
! U1 setvar "weblink.ip.conn2.test.location" "url"
```

Values

Any HTTPS URL up to 2048 characters

Default

`"http://www.zebra.com/apps/linktest"`

Getvar

To retrieve the printer's test connection URL:

```
! U1 getvar "weblink.ip.conn1.test.location"
```

```
! U1 getvar "weblink.ip.conn2.test.location"
```

Do

To set the URL to hold for testing a connection:

```
! U1 do "weblink.ip.conn1.test.location" "url"
```

```
! U1 do "weblink.ip.conn2.test.location" "url"
```

Values

Any HTTPS URL up to 2048 characters

Default

`"http://www.zebra.com/apps/linktest"`

Example

The test connection can assist the user in several ways/scenarios:

If the `test.test_on` value is set to `"failure"`, any time the main weblink (`conn[1|2].location`) connection fails to connect then the `test.location` URL will be used. In this situation, an attempt will be made to contact the remote URL in `test.location`, using authentication and proxy configuration that is specified by the main connection.

If the `test.test_on` value is set to `"interval"` an attempt will be made to contact the remote URL in `test.location` every `test.retry_interval` seconds, using authentication and proxy configuration that is specified by the main connection.

If the `test.test_on` value is set to `"both"`, then scenario 1 and 2 will both occur. This is useful for users who will use an HTTP connection to move through their firewall - and thereafter frequently refresh the connection to indicate to their firewall that there is still activity for the purpose of keeping the connection alive.

weblink.ip.conn[1|2].test.retry_interval

This command determines how often, in seconds, a connection to the test.location URL should be attempted. This setting is only applicable when the `test.test_on` SGD is set to `"interval"` or `"both"`.

`^JUF, ^JUS, ^JUN, ^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Setvar

To set the interval for how often a connection to the test.location URL should be attempted:

```
! U1 setvar "weblink.ip.conn1.test.retry_interval" "value"
```

```
! U1 setvar "weblink.ip.conn2.test.retry_interval" "value"
```

Values

0-1800 (in seconds, providing 0 second - 30 minute interval)

Default

`"900"`

Getvar

To retrieve the retry interval:

```
! U1 getvar "weblink.ip.conn1.test.retry_interval"
```

```
! U1 getvar "weblink.ip.conn2.test.retry_interval"
```

Do

To set the interval for how often a connection to the test.location URL should be attempted:

```
! U1 do "weblink.ip.conn1.test.retry_interval" "value"
```

```
! U1 do "weblink.ip.conn2.test.retry_interval" "value"
```

Values

0-1800 (in seconds, providing 0 second - 30 minute interval)

Default

`"900"`

weblink.ip.conn1.retry_interval_random_max

Specifies the maximum random wait time in seconds for weblink connection retries.

`^JUF, ^JUS, ^JUN, ^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Setvar

To set the command:

```
! U1 setvar "weblink.ip.conn[1|2].retry_interval_random_max" "value"
```

Values

`"0"` through `"600"`

Default

`"120"`

Examples

```
! U1 setvar "weblink.ip.conn[1].retry_interval_random_max" "120"
```

```
! U1 setvar "weblink.ip.conn[2].retry_interval_random_max" "60"
```

Getvar

To have the printer return the current setting value:

```
! U1 getvar "weblink.ip.conn[1|2].retry_interval_random_max"
```

weblink.ip.conn1.test.test_on

This command determines when the test connection should be attempted. This assists in debugging the printer's connection to remote servers when there are issues with the main weblink connection (conn1 or conn2).

`^JUF, ^JUS, ^JUN, ^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Setvar

To indicate when the test connection should be attempted:

```
! U1 setvar "weblink.ip.conn1.test.test_on" "value"
```

```
! U1 setvar "weblink.ip.conn2.test.test_on" "value"
```

Values

`off,failure,interval,both`

Default

`"failure"`

Getvar

To retrieve the test connection setting:

```
! U1 getvar "weblink.ip.conn1.test.test_on"
```

```
! U1 getvar "weblink.ip.conn2.test.test_on"
```

Do

To set when the test connection should be attempted:

```
! U1 do "weblink.ip.conn1.test.test_on" "value"
```

```
! U1 do "weblink.ip.conn2.test.test_on" "value"
```

Values

`off,failure,interval,both`

Default

`"failure"`

Example

The test connection can assist the user is several ways/scenarios:

If the `test.test_on` value is set to `"failure"`, any time the main weblink (`conn[1|2].location`) connection fails to connect then the `test.location` URL will be used. An attempt will be made to

contact the remote URL in `test.location`, using authentication and proxy configuration that is specified by the main connection.

If the `test.test_on` value is set to `"interval"` an attempt will be made to contact the remote URL in `test.location` every `test.retry_interval` seconds, using authentication and proxy configuration that is specified by the main connection.

If the `test.test_on` value is set to `"both"`, then scenario 1 and 2 will both occur. This is useful for users who will use an HTTP connection to move through their firewall - and thereafter frequently refresh the connection to indicate to their firewall that there is still activity for the purpose of keeping the connection alive.

weblink.logging.clear

This command clears the weblink log. Setting this value to anything will clear it, including an empty string.

`^JUF, ^JUS, ^JUN, ^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Setvar

To clear the weblink log entries:

```
! U1 setvar "weblink.logging.clear" "value"
```

Values

Any string value, including an empty string.

Default

NA

Do

To clear the weblink log entries:

```
! U1 do "weblink.logging.clear" "value"
```

Values

Any string value, including an empty string.

Default

NA

Example

This example clears the weblink log entries with an empty string value.

```
! U1 setvar "weblink.logging.clear" ""
```

weblink.logging.entries

This command returns the N number of entries in the weblink log, where N has a maximum value that is set by `weblink.logging.max_entries`.

The weblink log is a collection of events related to connecting to a remote Link-OS™ server. The log entries range anywhere from general status to errors that prevented a successful connection. The log contains entries from all connections and are labeled so that it is clear which log entries are for which connection. Each log entry also contains a timestamp for when it was logged by the system. The newest events will appear at the bottom of the list.

`^JUF`, `^JUS`, `^JUN`, `^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Getvar

To return a lists of entries in the weblink log:

```
! U1 getvar "weblink.logging.entries"
```

Values

NA

Default

""

Example

This example shows the result from `weblink.logging.entries`:

```
[01-04-2013 08:40:45.655] [conn1.1] HTTP/1.1 404 Not Found
[01-04-2013 08:40:45.659] [conn1.1] Received HTTP code 404 from proxy after
 CONNECT
[01-04-2013 08:40:45.660] [conn1.1] Closing connection
[01-04-2013 08:40:45.662] [conn1.1] Failed to connect (SP = 0, CU = 0, UW =
 0, AC = 0, PC = 0)
```

weblink.logging.max_entries

This command specifies the maximum number of individual log entries that will be stored in the `weblink.logging.entries` command.

NOTE: Changes to this command are immediate and may result in some log entries being lost. If there are N log entries currently in the log, the user sets the max_entires to M, where M is less than N, the oldest (N-M) log entries will be removed.

`^JUF, ^JUS, ^JUN, ^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Setvar

To set the maximum number of log entires that will be stored:

```
! U1 setvar "weblink.logging.max_entries" "value"
```

Values

`"0"` through `"10000"`

Setting the value to `"0"` disables logging.

Getvar

To return the setting for the maximum number of log entries that will be stored:

```
! U1 getvar "weblink.logging.max_entries"
```

Do

To set the maximum number of log entires that will be stored:

```
! U1 getvar "weblink.logging.max_entries"
```

Values

`"0"` through `"10000"`

Setting the value to `"0"` disables logging.

Default

`"0"`

Example

In this example, `weblink.logging.max_entries` is set to 3:

```
[01-04-2013 08:40:45.659] [conn1.1] Received HTTP code 404 from proxy after
 CONNECT
[01-04-2013 08:40:45.660] [conn1.1] Closing connection
[01-04-2013 08:40:45.662] [conn1.1] Failed to connect (SP = 0, CU = 0, UW =
 0, AC = 0, PC = 0)
```

In this example, `weblink.logging.max_entries` is set to 2: `weblink.logging.entries` becomes:

```
[01-04-2013 08:40:45.660] [conn1.1] Closing connection
[01-04-2013 08:40:45.662] [conn1.1] Failed to connect (SP = 0, CU = 0, UW =
 0, AC = 0, PC = 0)
```

weblink.printer_reset_required

This command retrieves a "yes" or "no" value indicating whether any of the weblink settings have been modified.

^JUF, ^JUS, ^JUN, ^JUA, and device.restore_defaults do not have any affect on this setting.

Getvar

To retrieve whether any of the weblink settings are modified:

```
! U1 getvar "weblink.printer_reset_required"
```

Values

- "yes"
- "no"

Default

"no"

weblink.restore_defaults

This command defaults, and saves, the weblink branch settings. Any value, including an empty string, will default the weblink branch settings.

NOTE: The entire weblink branch of settings will be defaulted and the settings are saved; however, the weblink connections will not use the new settings until the printer is restarted (e.g. the `weblink.printer_reset_required` SGD will be `"yes"` after a default).

`^JUF`, `^JUS`, `^JUN`, `^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Setvar

To default the weblink branch settings:

```
! U1 setvar "weblink.restore_defaults" "value"
```

Values

Any value, including an empty string, will default the branch

Default

NA

Do

To default the weblink branch settings:

```
! U1 do "weblink.restore_defaults" "value"
```

Values

Any value, including an empty string, will default the branch

Default

NA

Example

These all default the branch:

```
! U1 setvar "weblink.restore_defaults" ""
! U1 setvar "weblink.restore_defaults" "foo"
! U1 do "weblink.restore_defaults" ""
! U1 do "weblink.restore_defaults" "foo"
```

weblink.zebra_connector.authentication.add

This command allows the user to add a single server/username/password triplet into the list of authentication entries.

When the printer attempts to connect to the Zebra Printer Connector, the local server may require HTTP authentication (e.g. digest, basic, DNS, etc.). There may be multiple authentication requests along the route to the destination (e.g. a local server first requires HTTP authentication as well as on the remote server).

For each HTTP authentication request received while attempting to connect, the printer will enumerate the authentication entries and attempt to satisfy the request with the username/password pair provided for the respective server. The server name in the entry is what determines which username/password pair should be used for which authentication request. Both DNS names and IP addresses are acceptable. The server, username, and password are separated by a single space (not a tab or other white space character). The server name is the only required field. If no username is supplied, but a password is, there must be two spaces between the server and the password fields. If there is a username but no password, or simply just the servername, no space is required at the end of the entry.

If the command is changed while the Visibility Agent is enabled, it will not take effect until the connection is disabled, and then re-enabled.

IMPORTANT: This setting can only be changed when weblink.zebra_connector.enable is set to `"off"`.

`^JUF, ^JUS, ^JUN, ^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Setvar

To allow the user to add a single server/username/password triplet into the list of authentication entries:

```
! U1 setvar "weblink.zebra_connector.authentication.add" "server username
 password"
```

Values

- `"server"` is the IP address or a DNS name
- `"username"` is the user name on this server
- `"password"` is the password for this username on this server

Default

`""`

Result

```
! U1 setvar "weblink.zebra_connector.authentication.add"
```

```
"10.3.5.70 jsmith LedZepR0cks!"
```

weblink.zebra_connector.authentication.entries

This command lists the server names added to the authentication entries list.

Only the server names will be shown. The username and passwords will not be shown.The server names are separated by a `\r\n` so that each shows up on its own line.

`^JUF, ^JUS, ^JUN, ^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Getvar

To list the server names added to the authentication entries list:

```
! U1 getvar "weblink.zebra_connector.authentication.entries"
```

Result

Returns the list of servers with authentication entries. It does not return the username or passwords for those servers.

Default

""

weblink.zebra_connector.authentication.remove

Removes a single server/username/password triplet from the list of authentication entries.

To remove an entry only the server name is supplied, however the entire entry will be removed. If an invalid entry is supplied no action is taken.

`^JUF, ^JUS, ^JUN, ^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Setvar

To remove a single server/username/password triplet from the list of authentication entries:

```
! U1 setvar "weblink.zebra_connector.authentication.remove" "server"
```

Values

`"server"` an IP address or a DNS name of the server to remove

Default

NA

Example

```
! U1 setvar "weblink.zebra_connector.authentication.remove" "10.3.5.70"
```

weblink.zebra_connector.enable

Enables the Visibility Agent feature.

Setvar

To enable or disable the Visibility Agent feature:

```
! U1 setvar "weblink.zebra_connector.enable" "value"
```

Values

- `"on"` enables Visibility Agent
- `"off` disables Visibility Agent

Default

`"on"`

Example

```
! U1 setvar "weblink.zebra_connector.enable" "enable"
```

Getvar

To return the current setting value:

```
! U1 getvar "weblink.zebra_connector.enable"
```

Result

"on" or "off"

weblink.zebra_connector.proxy

This command assigns the URL for the proxy used to connect to the Zebra connector.

The proxy server protocol, port, domain, username, and password are all encoded into the URL via the format outlined in RFC2396 (http://www.ietf.org/rfc/rfc2396.txt).

The username and password must avoid the invalid characters listed in RFC2396 (e.g. ':', '@', '/', etc). If an invalid character must be used it needs to be escaped using '%' as described in RFC2396.

`^JUF, ^JUS, ^JUN, ^JUA`, and `device.restore_defaults` do not have any affect on this setting.

Setvar

To assign the URL for the proxy used to connect to the Zebra connector:

```
! U1 setvar "weblink.zebra_connector.proxy" "url"
```

Values

Any valid URL up to 2048 characters

Expected URL format: [http|https]://[user:pass@]domain[:port]/[path]

The URL will need to be built according to the server/proxy environment the printer is running where:

- `"user"` username
- `"password"` password
- `"host"` either the hostname or IP address
- `"port"` port number
- `"other"` anything else needed in the path

Default

`""`

- The user:pass, port, and path are all optional.
- The default scheme must be either HTTPS or HTTP. The default is HTTP.
- The default port is 1080.

The default is to omit the username and password.

Example

```
! U1 setvar "weblink.zebra_connector.proxy"
"https://user:pass@my.internal.proxy:7840/init"
```

Getvar

To return the current setting value:

```
! U1 getvar "weblink.zebra_connector.proxy"
```

Result

```
"https://user:pass@my.internal.proxy:7840/init"
```

weblink.zebra_connector.version

Displays the current version of the Zebra Connector.

Getvar

To display the current version of the Zebra connector:

```
! U1 getvar "weblink.zebra_connector.version"
```

Values

A version number in the form of xx.yy.

Default

""

Example

```
! U1 getvar "weblink.zebra_connector.version" "0.04"
```

wlan.11ac.80mhz_enable

Enables the 80MHz wide channel support for the 802.11AC radio.

Setvar

To enable or disable the 80MHz wide channel support for the 802.11 AC radio:

```
! U1 setvar "wlan.11ac.80mhz_enable" "value"
```

Values

- `"on"` enables the 80MHz wide channel support
- `"off"` disables the 80MHz wide channel support

Default

""

Getvar

To return the current setting value:

```
! U1 getvar "wlan.11ac.80mhz_enable"
```

wlan.11d.enable

Allows the client to automatically configure themselves to their local regulatory domain.

Setvar

To set the command:

```
! U1 setvar "wlan.11d.enable" "value"
```

Values

- `"off"`
- `"on"`

Default

`"off"`

Getvar

To have the printer return the current setting value:

```
! U1 getvar "wlan.11d.enable"
```

wlan.11n.20mhz_only

This command forces 20 MHz operation only in printers with 802.11n radios.

Setvar

To force 20 MHz operation only in printers with 802.11n radios:

```
! U1 setvar "wlan.11n.20mhz_only" "value"
```

Values

- `"on"`
- `"off"`

Default

`"off"`

Getvar

To return the current setting for 20 MHz operation only:

```
! U1 getvar "wlan.11n.20mhz_only"
```

Example

```
U1 setvar "wlan.11n.20mhz_only" "on"
```

wlan.11n.aggregation

This command enable or disables Aggregation MAC Service Data Unit (A-MSDU) in 802.11n radio communications.

Setvar

To enable or disable A-MSDU in 802.11n radio communications:

```
! U1 setvar "wlan.11n.aggregation" "value"
```

Values

- `"on"`
- `"off"`

Default

`"off"`

Getvar

To return the current setting for A-MSDU:

```
! U1 getvar "wlan.11n.aggregation"
```

Example

```
! U1 setvar "wlan.11n.aggregation" "on"
```

wlan.11n.greenfield

This command will enable or disable greenfield mode in 802.11n radio communications.

Setvar

To enable or disable the greenfield mode:

```
! U1 setvar "wlan.11n.greenfield" "value"
```

Values

- `"on"`
- `"off"`

Default

`"off"`

Getvar

To return the current setting for greenfield mode:

```
! U1 getvar "wlan.11n.greenfield"
```

Example

```
! U1 setvar "wlan.11n.greenfield" "on"
```

wlan.11n.rifs

This command enable/disables Reduced Interframe Space (RIFS) in 802.11n radio communications.

Setvar

To enable or disable RIFS in 802.11n:

```
! U1 setvar "wlan.11n.rifs" "value"
```

Values

- `"on"`
- `"off"`

Default

`"off"`

Getvar

To return the current setting for RIFS:

```
! U1 getvar "wlan.11n.rifs"
```

Example

```
! U1 setvar "wlan.11n.rifs" "on"
```

wlan.11n.short_gi_40mhz

This command enables/disables short Guard Interval in 40 mhz mode.

Setvar

To enable or disable short Guard Interval in 40 mhz mode:

```
! U1 setvar "wlan.11n.short_gi_40mhz" "value"
```

Values

- `"on"`
- `"off"`

Default

`"off"`

Getvar

To return the current setting for short Guard Interval 40mhz operation:

```
! U1 getvar "wlan.11n.short_gi_40mhz"
```

Example

```
! U1 setvar "wlan.11n.short_gi_40mhz" "on"
```

wlan.11n.short_gi_20mhz

This command enables/disables short Guard Interval (GI) in 20 mhz mode.

Setvar

To enable or disable short Guard Interval in 20 mhz mode:

```
! U1 setvar "wlan.11n.short_gi_20mhz" "value"
```

Values

- `"on"`
- `"off"`

Default

`"off"`

Getvar

To return the current setting value:

```
! U1 getvar "wlan.11n.short_gi_20mhz"
```

Example

```
! U1 setvar "wlan.11n.short_gi_20mhz" "on"
```

wlan.8021x.enable

Enables or disables the 802.1x security protocol, with the option to set it to WPA security protocol.

Setvar

To set the command:

```
! U1 setvar "wlan.8021x.enable" "value"
```

Values

- `"off"` 802.1x security protocol is disabled.
- `"on"` 802.1x security protocol is enabled.
- `"wpa"` 802.1x security protocol is enabled and uses WPA.

Default

`"off"`

Getvar

To have the printer return the current setting value:

```
! U1 getvar "wlan.8021x.enable"
```

wlan.8021x.validate_peap_server_certificate

This command determines if the printer will validate the PEAP server certificate.

Setvar

To determine if the printer will validate the PEAP server certificate:

```
! U1 setvar "wlan.8021x.validate_peap_server_certificate" "value"
```

Values

- `"on"`
- `"off"`

Default

`"on"`

Getvar

To return the current setting:

```
! U1 getvar "wlan.8021x.validate_peap_server_certificate"
```

Example

```
! U1 setvar "wlan.8021x.validate_peap_server_certificate"
```

IMPORTANT: When using certificate files, the time on the printer must be set correctly for the websocket connection to succeed, as the time is used in the certificate validation.

wlan.8021x.peap.anonymous_identity

This command is used to specify the phase 1 ID when using PEAP to authenticate with the wireless network.

Setvar

To specify the phase 1 ID to be used during peap authentication:

```
! U1 setvar "wlan.8021x.peap.anonymous_identity" "<value>"
```

Values

"<value>" is less than or equal to 32 characters

Default

""

Getvar

To retrieve the current value:

```
! U1 getvar "wlan.8021x.peap.anonymous_identity"
```

wlan.8021x.authentication

Sets the authentication type used in the 802.1x security protocol.

Setvar

To set the authentication type:

```
! U1 setvar "wlan.8021x.authentication" "value"
```

Values

`"psk"`	Pre-Shared Key
`"leap"`	Lightweight Extensible Authentication Protocol
`"eap-tls"`	EAP-Transport Layer Security
`"peap"`	Protected Extensible Authentication Protocol
`"ttls"`	Tunneled Transport Layer Security
`"fast"`	Flexible Authentication via Secure Tunneling

Default

`"psk"`

Getvar

To return the current setting value:

```
! U1 getvar "wlan.8021x.authentication"
```

wlan.8021x.eap.password

Sets the EAP authentication password.

Setvar

To set the EAP authentcation password:

```
! U1 setvar "wlan.8021x.eap.password" "password"
```

Values

32 characters or less representing the EAP password.

Default

""

Getvar

To return the current setting value:

```
! U1 getvar "wlan.8021x.eap.password"
```

Result

"*"

Password is not readable. Printer reports "*" in response to this command.

wlan.8021x.eap.username

Sets or returns the user name associated with the EAP authentication protocol.

Setvar

To set the EAP user name:

```
! U1 setvar "wlan.8021x.eap.username" "username"
```

Values

32 characters or less representing the EAP user name.

Default

""

Getvar

To return the current setting value:

```
! U1 getvar "wlan.8021x.eap.username"
```

Result

"username"

wlan.8021x.eap.privkey_password

Sets the EAP private key password to the specified password.

Setvar

To set the EAP private key password:

```
! U1 setvar "wlan.8021x.eap.privkey_password" "password"
```

Values

32 characters or less representing the EAP private key password.

Default

""

Getvar

To return the current setting value:

```
! U1 getvar "wlan.8021x.eap.privkey_password"
```

Result

"*"

Printer always retrieves "*" regardless of the length of the stored EAP authentication private key password, and does not reveal the actual password.

wlan.8021x.peap.peap_password

Sets the password associated with the PEAP authentication protocol.

Setvar

To set the PEAP password:

```
! U1 setvar "wlan.8021x.peap.peap_password" "password"
```

Values

32 characters or less representing the PEAP password.

Default

`"password"`

Getvar

To return the current setting value:

```
! U1 setvar "wlan.8021x.peap.peap_password"
```

Result

`"*"`

Printer reports one `"*"` regardless of the length of the stored PEAP authentication password, and does not reveal the actual password.

wlan.8021x.peap.privkey_password

Sets or returns the PEAP authentication private key password.

Setvar

To set the PEAP authentication password:

```
! U1 setvar "wlan.8021x.peap.privkey_password" "password"
```

Values

32 characters or less representing the PEAP private key password.

Default

"*"

Getvar

To return the current setting value:

```
! U1 getvar "wlan.8021x.peap.privkey_password"
```

Result

"*"

Printer reports one "*" regardless of the length of the stored PEAP private key password, and does not reveal the actual password.

wlan.8021x.peap.peap_username

Sets the user name associated with the PEAP authentication protocol.

Setvar

To set the user name associated with the PEAP authentication protocol:

```
! U1 getvar "wlan.8021x.peap.peap_username" "username"
```

Values

32 characters or less representing the PEAP username.

Default

`"username"`

Getvar

To return the current setting value:

```
! U1 getvar "wlan.8021x.peap.peap_username"
```

Result

`"username"`

wlan.8021x.ttls_anonymous_identity

This command returns the anonymous identity string for EAP-TTLS that is used as the unencrypted identity for a WLAN. The value of the identity string can be up to 32 characters.

Getvar

To return the anonymous identity string:

```
! U1 getvar "wlan.8021x.ttls_anonymous_identity"
```

Default
NA

wlan.active_channels

This command returns list of current active channels defined by `wlan.allowed_band` and `wlan.user_channel_list`.

Getvar

To return the list of current active channels:

```
! U1 getvar "wlan.active_channels"
```

Result

One or more of the following channels:
`1,2,3,4,5,6,7,8,9,10,11,36,40,44,48,52,56,60,64,100,104,108,112,116,132,136,140,149,153,157,161,165,all`

wlan.adhoc_last_channel

Reports the last channel used for Wi-Fi ad hoc communication.

Getvar

To return the last channel used for Wi-Fi ad hoc communication:

```
! U1 getvar "wlan.adhoc_last_channel"
```

Values

A numeric value between `"1"` and `"11"`.

Default

`"1"`

wlan.authenticated

Returns the authentication state of the wireless interface.

Getvar

To return the authentication state of the wireless interface:

```
! U1 getvar "wlan.authenticated"
```

Values

- `"yes"` the printer WLAN connection has been authenticated
- `"no"` the printer WLAN connection has not been authenticated

wlan.authentication_error

Reports the last error that occurred during the WLAN authentication process.

Getvar

To report the last error that occurred during the WLAN authentication process:

```
! U1 getvar "wlan.authentication_error"
```

Values

- `"none"` the authentication was successful.
- `"timed out"` the authentication did not succeed in the allotted time.

wlan.available

Determines if a WLAN radio has been installed in the printer.

Getvar

To determine if a WLAN radio has been installed in the printer:

```
! U1 getvar "wlan.available"
```

Values

- `"yes"` a WLAN radio is installed
- `"no"` a WLAN radio is not installed

wlan.allowed_band

This command provides a method to restrict the 802.11n radio to either the 2.4 GHz or 5 GHz band.

Use this option when it is beneficial to have the radio use only one frequency band. Setting the 802.11n radio to one band will reduce roaming and radio association times since the radio will not scan as many channels.

Setvar

To restrict the 802.11n radio to either the 2.4 GHz or 5 GHz band:

```
! U1 setvar "wlan.allowed_band" "value"
```

Values

`"2.4"`, `"5"`,or `"all"`

Default

`"all"`

If both bands are desired, use `"all"`

Getvar

To return the current setting for allowed bands:

```
! U1 getvar "wlan.allowed_band"
```

Example

This example sets the allowed band to only the 2.4 GHz band.

```
! U1 setvar "wlan.allowed_band" "2.4"
```

wlan.adhocautomode

This printer setting refers to enabling or disabling the adhoc auto mode.

Setvar

To instruct the printer to set the adhoc auto mode:

```
! U1 setvar "wlan.adhocautomode" "value"
```

Values

- `"on"` adhoc auto mode enabled
- `"off"` adhoc auto mode disabled

Default

`"off"`

Getvar

To instruct the printer to respond with the adhoc auto mode status:

```
! U1 getvar "wlan.adhocautomode"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "wlan.adhocautomode" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`.

wlan.adhocchannel

This printer setting refers to specifying the wireless channel for adhoc channel.

Setvar

To set the printer's wireless channel for adhoc channel mode:

```
! U1 setvar "wlan.adhocchannel" "value"
```

Values

Decimal value between 1 and 16 inclusive

Default

`"1"`

Getvar

To respond with the printer's wireless channel for adhoc channel mode:

```
! U1 getvar "wlan.adhocchannel"
```

Example

This `setvar` example shows the value set to `"1"`.

```
! U1 setvar "wlan.adhocchannel" "1"
```

When the `setvar` value is set to `"1"`, the `getvar` result is `"1"`.

wlan.associated

This command refers to if the printer is or is not associated with an access point (AP).

Getvar

To instruct the printer to respond with yes or no, which identifies if it is associated with the AP:

```
! U1 getvar "wlan.associated"
```

Example

In this example, the `getvar` result is `"yes"`.

```
! U1 getvar "wlan.associated"
```

wlan.auth_type

This parameter selects the authentication service to be used between the printer and the Access Point. Open System and Shared Key are the two types of authentication services.

Setvar

To instruct the printer to set the authentication type to the specified value:

```
! U1 setvar "wlan.auth_type" "value"
```

Values

- `"open"`
- `"shared"`

Default

`"open"`

Getvar

To instruct the printer to retrieve the current authentication type:

```
! U1 getvar "wlan.auth_type"
```

Example

This example instructs the printer to set the authentication type to Shared Key.

```
! U1 setvar "wlan.auth_type" "shared"
```

The authentication type will ble set to Shared Key after power cycle.

wlan.band_preference

This command sets a preferred band to connect with using Wi-Fi.

Setvar

To specify the WLAN band preference:

```
! U1 getvar "wlan.band_preference" "none"
```

Values

- `"2.4"`
- `"5"`
- `"none"`

Default

`"none"`

Getvar

To return the current WLAN band preference value:

```
! U1 getvar "wlan.band_preference"
```

Example

In the `setvar` example below, the WLAN band preference is set to `"5"`.

```
! U1 getvar "wlan.band_preference" "5"
```

wlan.bssid

This command returns the MAC address of the access point (AP) with which the printer is associated.

Getvar

To retrieve the MAC address of the access point (AP):

```
! U1 getvar "wlan.bssid"
```

Example

In this example, the `getvar` result is the MAC address of the access point.

```
! U1 getvar "wlan.bssid"
```

wlan.channel

This command retrieves the current Wi-Fi channel the printer is using.

Getvar

To retrieve the current Wi-Fi channel the printer is using:

```
! U1 getvar "wlan.channel"
```

Example

In this example, the `getvar` retrieves the current Wi-Fi channel the printer is using.

```
! U1 getvar "wlan.channel"
```

wlan.channel_mask

This command specifies the wireless channel masks for enabling and disabling various channels. This controls which b/g radio channels can be used by the radio for network connections.

The value for this command is a bit field where a 0 disables a channel and a 1 enables the channel. Starting from the right, bit 0 is for channel 1, bit 1 for channel 2, etc. This can be used to limit the channels scanned for networks, which may slightly improve connection and roaming speed. It also used to ensure compliance with the regulatory domains of your location.

Commonly Used Channel Mask Settings		
Region	**Channels**	**Channel Mask**
United States, Canada, Latin America	1 - 11	0x7FF
Europe, Middle East, Africa, other	1 - 13	0x1FFF
Japan	1 - 14	0x3FFF

NOTE: This command is not supported by all radios. Ensure the channel masks are set in accordance with the regulatory domains of your country.

Setvar

To instruct the printer to set the wireless channel mask value:

```
! U1 setvar "wlan.channel_mask" "value"
```

Values

`"0x0000"` to `"0xFFFF"` (4 hexadecimal digits preceded by `"0x"`)

Default

`"0x7FF"`

Getvar

To instruct the printer to set the wireless channel mask value:

```
! U1 getvar "wlan.channel_mask"
```

Example 1

This `setvar` example shows the value set to `"0x7FF" for common North American channels`.

```
! U1 setvar "wlan.channel_mask" "0x7FF"
```

Example 2

This setvar example sets the channel mask to use only channels 1,6,11.

```
! U1 setvar "wlan.channel_mask" "0x421"
```

Only channels 1, 6, and 11 will be used by the radio.

wlan.country_code

This command defines the regulatory country for which the radio is currently configured.

Setvar

To set the country code for which the radio is to be configured:

```
! U1 setvar "wlan.country_code" "value"
```

IMPORTANT: The list of country codes is specific to each printer and depends on the printer model and its wireless radio configuration. The list is subject to change, addition, or deletion with any firmware update, at any time, without notice.

To determine the country codes available on your printer, issue the `! U1 getvar "wlan"` command to return all commands related to WLAN settings. Locate the `wlan.country.code` command in the results and view the country codes available for your printer.

Getvar

To retrieve the country code for which the radio is currently configured:

```
! U1 getvar "wlan.country_code"
```

Example

In this example, the `setvar` sets the country code to USA/Canada.

```
! U1 setvar "wlan.country_code" "usa/canada"
```

wlan.current_tx_rate

This command retrieves the transmission rate of the wireless print server.

Getvar

To retrieve the current transmit rate of the wireless print server:

```
! U1 getvar "wlan.current_tx_rate"
```

Example

In this example, the `getvar` retrieves the transmission rate of the wireless print server.

```
! U1 getvar "wlan.current_tx_rate"
```

wlan.enable

This parameter can be used to enable or disable the Wireless LAN functionality of the printer.

Setvar

To enable or disable printer's Wireless LAN functionality:

```
! U1 setvar "wlan.enable" "value"
```

Values

- `"on"` Wireless LAN functionality is enabled
- `"off"` Wireless LAN functionality is disabled

Default

`"off"`

Getvar

To return the current setting value:

```
! U1 getvar "wlan.enable"
```

wlan.encryption_index

This parameter refers to the WEP (Wired Equivalent Privacy) encryption key index. It determines which one of the four encryption keys is to be used by the client (printer).

Setvar

To set the encryption key index to the specified value:

```
! U1 setvar "wlan.encryption_index" "value"
```

Values

- `"1"`
- `"2"`
- `"3"`
- `"4"`

Default

`"1"`

Getvar

To retrieve the current encryption key setting:

```
! U1 getvar "wlan.encryption_index"
```

Example

In this example, the `setvar` command instructs the printer to set the encryption key index to 1.

```
! U1 setvar "encryption_index" "1"
```

wlan.encryption_key

These parameters refer to the first, second, third, and fourth indexed WEP encryption keys. The WEP encryption keys are hexadecimal strings that are either 10 or 26 characters long depending on the encryption method (40-bit or 128 bit). The keys should match the wireless network WEP encryption keys.

Setvar

To set the encryption key value:

```
! U1 setvar "wlan.encryption_key[1|2|3|4]" "value"
```

Values

10 hexadecimal characters for 40-bit encryption and 26 hexadecimal characters for 128-bit encryption.

Default

All zeroes (10 or 26, depending on encryption setting)

Getvar

To instruct the printer respond with the encryption key value:

```
! U1 getvar "wlan.encryption_key1"
```

Example 1

In these examples, the `getvar` results assume that the printer is using 40-bit encryption with the default settings.

```
! U1 getvar "wlan.encryption_key1"
```

```
! U1 getvar "wlan.encryption_key2"
```

```
! U1 getvar "wlan.encryption_key3"
```

```
! U1 getvar "wlan.encryption_key4"
```

Results for each key: `"0000000000"`

Example 2

In these examples, the `setvar` command instructs the printer to set the encryption key value. This example assumes that the printer is using 40-bit encryption.

```
! U1 setvar "wlan.encryption_key1" "A1B2C3D4F5"
```

```
! U1 setvar "wlan.encryption_key2" "G1H2J3K4L5"
```

```
! U1 setvar "wlan.encryption_key3" "M1N2P3Q4R5"
```

```
! U1 setvar "wlan.encryption_key4" "S1T2V3W4X5"
```

wlan.encryption_mode

This parameter refers to WEP (Wired Equivalent Privacy) encryption. This parameter enables and disables the printer's WEP encryption. When using WEP encryption make sure that the encryption key matches the wireless network WEP encryption key.

Setvar

To instruct the printer to turn the LEAP mode `"on"` or `"off":`

```
! U1 setvar "wlan.encryption_mode" "value"
```

Values

- `"off"`
- `"40-bit"`
- `"128-bit"`

Default

`"off"`

Getvar

To return the type of encryption that is currently being used by the printer:

```
! U1 getvar "wlan.encryption_mode"
```

Example 1

This example instructs the printer to set encryption to 40-bit.

```
! U1 setvar "wlan.encryption_mode" "40-bit"
```

Example 2

This example instructs the printer to turn encryption off.

```
! U1 setvar "wlan.encryption_mode" "off"
```

wlan.essid

This printer setting refers to the printer's stored ESSID.

Setvar

To instruct the printer to change the ESSID:

```
! U1 setvar "wlan.essid" "value"
```

Values

32 character alphanumeric string

Default

"125" (Printers running Link-OS v5.3 or earlier versions)

"" (For Printers running Link-OS 6 or later versions)

Getvar

To instruct the printer to respond with the stored ESSID value:

```
! U1 getvar "wlan.essid"
```

Example

This `setvar` example shows the value set to "125".

```
! U1 setvar "wlan.essid" "125"
```

When the `setvar` value is set to "125", the `getvar` result is "125".

NOTE: For Link-OS Firmware earlier than 6.0, setting ESSID to "" allows the printer to attempt to connect to an AP with any ESSID. For Link-OS Firmware of 6.0 or later, setting ESSID to "" means that the printer will not attempt a Wi-Fi connection.

wlan.firmware_version

This command refers to the firmware version of the wireless radio card.

Getvar

To instruct the printer to respond with the current version of the wireless radio card firmware:

```
! U1 getvar "wlan.firmware_version"
```

Example

In this example, the `getvar` result is the version of Symbol 4137 card (for example, `"F3.91-69"`).

```
! U1 getvar "wlan.firmware_version"
```

wlan.ip.addr

This command allows you to get or set the wireless print servers IP address.

IMPORTANT: For a set IP address to take affect, the IP protocol must be set to permanent and the print server must be reset.

Setvar

To instruct the printer to change its current wireless print server IP address upon powering the printer on:

```
! U1 setvar "wlan.ip.addr" "value"
```

Values

Any valid IP address

Default

`"0.0.0.0"`

Getvar

To respond with the current wireless print server IP address:

```
! U1 getvar "wlan.ip.addr"
```

Example

This `setvar` example shows the value set to `"10.14.4.235"`.

```
! U1 setvar "wlan.ip.addr" "10.14.4.235"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"10.14.4.235"`.

wlan.ip.arp_interval

This print server setting allows you to specify the ARP (Address Resolution Protocol) interval or the ARP cache time out for the wireless print server.

Setvar

To instruct the printer to respond with the ARP interval or the ARP cache time out value for the wireless print server:

```
! U1 setvar "wlan.ip.arp_interval" "value"
```

Values
"0" to "30" seconds

Default
"0"

Getvar

To respond with the ARP interval or the ARP cache time out value (in seconds) for the wireless print server:

```
! U1 getvar "wlan.ip.arp_interval"
```

Example

This `setvar` example shows the value set to "0".

```
! U1 setvar "wlan.ip.arp_interval" "0"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is "0".

wlan.ip.default_addr_enable

This command allows you to default the wireless print server's IP address.

IMPORTANT: For a set IP address to take affect, the IP protocol must be set to permanent and the print server must be reset.

Setvar

To enable or disable te printer to use its default IP address, if no address is provided through DHCP or BOOTP:

```
! U1 setvar "wlan.ip.default_addr_enable" "value"
```

If you do not assign an IP address after 2 minutes, the 10/100 Internal PS defaults to IP address 192.168.254.254.

Values

- `"on"` enabled
- `"off"` disabled

Default

`"on"`

Getvar

To instruct the printer to show the status of the setting of the wireless print server's default IP address feature:

```
! U1 getvar "wlan.ip.default_addr_enable"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "wlan.ip.default_addr_enable" "on"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"on"`.

wlan.ip.dhcp.arp_verify

Specifies whether an ARP broadcast will be sent out to verify that the IP address received from the DHCP server is not already in use.

Setvar

To set the ARP broadcast feature:

```
! U1 setvar "wlan.ip.dhcp.arp_verify" "value"
```

Values

- `"on"` means an ARP broadcast will be sent out
- `"off"` means an ARP broadcast will not be sent out

Getvar

To return the current setting value:

```
! U1 getvar "wlan.ip.dhcp.arp_verify"
```

wlan.ip.dhcp.cache_ip

This command enables or disables the IP cache of the wireless print server.

Setvar

To set the status of the IP cache:

```
! U1 setvar "ip.dhcp.cache_ip" "value"
```

Values

- `"on"` means enabled
- `"off"` means disabled

Default

`"off"`

Getvar

To retrieve the status of the IP cache:

```
! U1 getvar "wlan.ip.dhcp.cache_ip"
```

Example

This `setvar` example shows the value set to `"off"`.

```
! U1 setvar "wlan.ip.dhcp.cache_ip" "off"
```

When the `setvar` value is set to `"off"`, the `getvar` result is `"off"`.

wlan.ip.dhcp.cid_all

This printer setting defines the entire client identifier (DHCP option 61) if DHCP is enabled on the wireless print server and "wlan.ip.dhcp.cid_type" is set to "0", or "2". The MAC address is used if the type is set to "1".

Setvar

To change the client identifer prefix and suffix of the wireless print server:

```
! U1 setvar "wlan.ip.dhcp.cid_all" "value"
```

Values

A maximum length of 60 characters if the CID type is ASCII, or 120 characters if the CID type is hexadecimal.

Default

""

Getvar

To instruct the printer to respond with the client identifier prefix and suffix of the wireless print server:

```
! U1 getvar "wlan.ip.dhcp.cid_all"
```

Example

This setvar example shows the value set to "printer".

```
! U1 setvar "wlan.ip.dhcp.cid_all" "printer"
```

What the setvar value is set to is the getvar result. In this example, the getvar result is "printer".

wlan.ip.dhcp.cid_enable

This command determines if DHCP (option 61) is turned on or off of the wireless print server.

Setvar

To instruct the printer to set the status of the client identifier of the wireless print server:

```
! U1 setvar "wlan.ip.dhcp.cid_enable" "value"
```

Values

- `"off"` means the client identifier is turned off
- `"on"` means the client identifier is turned on

Default

`"off"`

Getvar

To instruct the printer to respond with the status of the client identifier of the wireless print server:

```
! U1 getvar "wlan.ip.dhcp.cid_enable"
```

Example

This `setvar` example shows the value set to `"off"`.

```
! U1 setvar "wlan.ip.dhcp.cid_enable" "off"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"off"`.

wlan.ip.dhcp.cid_prefix

This printer setting defines the prefix to be prepended to the DHCP client identifier (option 61) when DHCP is enabled on the wireless print server and `"wlan.ip.dhcp.cid_type"` is set to `"0"` or `"2"`.

Setvar

To instruct the printer to change the CID prefix of the wireless print server:

```
! U1 setvar "wlan.ip.dhcp.cid_prefix" "value"
```

Values

Any text string up to 10 characters if the CID type is ASCII, or 20 characters if the CID type is hexadecimal.

Default

`""`

Getvar

To instruct the printer to respond with the client identifier prefix of the wireless print server:

```
! U1 getvar "wlan.ip.dhcp.cid_prefix"
```

Example

This `setvar` example shows the value set to `"PRT001"`.

```
! U1 setvar "wlan.ip.dhcp.cid_prefix" "PRT001"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"PRT001"`.

wlan.ip.dhcp.cid_suffix

This printer setting defines the unique suffix to be used as the client identifier (DHCP option 61) if DHCP is enabled on the wireless print server and `"wlan.ip.dhcp.cid_type"` is set to `"0"` or `"2"`.

Setvar

To instruct the printer to change the client identifier suffix value on the wireless print server:

```
! U1 setvar "wlan.ip.dhcp.cid_suffix" "value"
```

Values

The maximum length of a value allowed is 60 ASCII characters when the CID type is ASCII, or 120 hexadecimal values when the CID type is hexadecimal.

Default

`""`

Getvar

To instruct the printer to respond with the client identifier suffix on the wireless print server:

```
! U1 getvar "wlan.ip.dhcp.cid_suffix"
```

Example

This `setvar` example shows the value set to `"printer"`.

```
! U1 setvar "wlan.ip.dhcp.cid_suffix" "printer"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"printer"`.

wlan.ip.dhcp.cid_type

This printer setting defines the type of client identifier (DHCP option 61) that will be sent if DHCP is enabled on the wireless print server. A value of "1" means the type of Ethernet and the printer's MAC address will be used. A value of "0" or "2" means the client identifier sent will be "wlan.ip.dhcp.cid_prefix" concatenated with "wlan.ip.dhcp.cid_suffix".

Setvar

To instruct the printer to enable "synthetic" client identifier for the wireless print server:

```
! U1 setvar "wlan.ip.dhcp.cid_type" "value"
```

Values

- "0" means ASCII string
- "1" means MAC address of the wireless radio card
- "2" means HEX value

Default

"1"

Getvar

To instruct the printer to respond with the client identifier type for the wireless print server:

```
! U1 getvar "wlan.ip.dhcp.cid_type"
```

Example

This setvar example shows the value set to "1".

```
! U1 setvar "wlan.ip.dhcp.cid_type" "1"
```

What the setvar value is set to is the getvar result. In this example, the getvar result is "1".

wlan.ip.dhcp.lease.last_attempt

This command retrieves the last time a DHCP request was sent from the wireless print server.

Getvar

To retrieve the last time a DHCP request was sent from the wireless print server:

```
! U1 getvar "wlan.ip.dhcp.lease.last_attempt"
```

Example

In this example, the `getvar` retrieves the last time a DHCP request was sent from the wireless print server.

```
! U1 getvar "wlan.ip.dhcp.lease.last_attempt"
```

wlan.ip.dhcp.lease.length

This command retrieves the original length (in seconds) of the DHCP lease on the wireless print server.

Getvar

To retrieve the original length (in seconds) of the DHCP lease on the wireless print server:

```
! U1 getvar "wlan.ip.dhcp.lease.length"
```

Example

This command retrieves the original length of the DHCP lease on the wireless print server.

```
! U1 getvar "wlan.ip.dhcp.lease.length"
```

wlan.ip.dhcp.lease.server

This command retrieves the address of the print server that provided the DHCP lease on the wireless print server.

Getvar

To retrieve the address of the print server that provided the DHCP lease on the wireless print server:

```
! U1 getvar "wlan.ip.dhcp.lease.server"
```

Example

In this example, the `getvar` retrieves the server that provided the DHCP lease on the wireless print server.

```
! U1 getvar "wlan.ip.dhcp.lease.server"
```

wlan.ip.dhcp.lease.time_left

This command retrieves the time (in seconds) left in the current DHCP lease on the wireless print server.

Getvar

To retrieve the time (in seconds) left in the current DHCP lease on the wireless print server:

```
! U1 getvar "wlan.ip.dhcp.lease.time_left"
```

Example

In this example, the `getvar` retrieves the time left in the current DHCP lease on the wireless print server.

```
! U1 getvar "wlan.ip.dhcp.lease.time_left"
```

wlan.ip.dhcp.option12

This command specifies if the DHCP option 12 (host name) is on or off in the discovery packet that is sent from the wireless print server.

Setvar

To enable or disable the DHCP option 12:

```
! U1 setvar "wlan.ip.dhcp.option12" "value"
```

Values

- `"on"` turns on option 12
- `"off"` turns off option 12

Default

`"on"`

Getvar

To return the current setting value:

```
! U1 getvar "wlan.ip.dhcp.option12"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "wlan.ip.dhcp.option12" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`.

wlan.ip.dhcp.option12_format

This command specifies the format of the option 12 value to be used in the discovery packet of the wireless print server.

Setvar

To set the format of option 12 value to be used in the discovery packet of the wireless print server:

```
! U1 setvar "wlan.ip.dhcp.option12_format" "value"
```

Values

String

Default

""

Getvar

To retrieve the format of option 12 value to be used in the discovery packet of the wireless print server:

```
! U1 getvar "wlan.ip.dhcp.option12_format"
```

Example

This `setvar` example shows configuring the `wlan.ip.dhcp.option12_format` to the value contained in the `device.friendly_name`.

It is necessary to surround the SGD entry to be used as source for the data with the < and > characters.

```
! U1 setvar "wlan.ip.dhcp.option12_format" "<device.friendly_name>"
```

To further explain, if the above command was issued and the value currently stored in the `device.friendly_name` parameter was `"ShipPrinter"`, then the response to the following command would be `"ShipPrinter"`:

```
! U1 getvar "wlan.ip.dhcp.option12_value"
```

wlan.ip.dhcp.option12_value

This command retrieves the actual value which will be used in the discovery packet of the wireless print server.

Getvar

To return the actual value which will be used in the discovery packet of the wireless print server:

```
! U1 getvar "wlan.ip.dhcp.option12_value"
```

Example

This `setvar` example shows configuring the `wlan.ip.dhcp.option12_format` to the value contained in the `device.friendly_name`.

It is necessary to surround the SGD entry to be used as source for the data with the < and > characters.

```
! U1 setvar "wlan.ip.dhcp.option12_format" "<device.friendly_name>"
```

To further explain, if the above command was issued and the value currently stored in the `device.friendly_name` parameter was `"ShipPrinter`, then the response to following command would be `"ShipPrinter`:

```
! U1 getvar "wlan.ip.dhcp.option12_value"
```

wlan.ip.dhcp.request_timeout

This command sets the maximum time (in seconds) to wait for a response to a DHCP discovery request on the wireless print server.

Setvar

To set the maximum time (in seconds) to wait for a response to a DHCP discovery request on the wireless print server:

```
! U1 setvar "wlan.ip.dhcp.request_timeout" "value"
```

Values

"2" through "30"

Default

"2"

Getvar

To retrieve the maximum time (in seconds) to wait for a response to a DHCP discovery request on the wireless print server:

```
! U1 getvar "wlan.ip.dhcp.request_timeout"
```

Example

This `setvar` example shows the value set to "2".

```
! U1 setvar "wlan.ip.dhcp.request_timeout" "2"
```

When the `setvar` value is set to "2", the `getvar` result is "2".

wlan.ip.dhcp.requests_per_session

This command retrieves the maximum amount of DHCP discover requests for a single DHCP session on the wireless print server.

Setvar

To set the maximum amount of DHCP discover requests for a single DHCP session on the wireless print server:

```
! U1 setvar "wlan.ip.dhcp.requests_per_session" "value"
```

Values

"1" through "10"

Default

"2"

Getvar

To retrieve the currently set maximum amount of DHCP discover requests for a single DHCP session on the wireless print server:

```
! U1 getvar "wlan.ip.dhcp.requests_per_session"
```

Example

This `setvar` example shows the value set to "2".

```
! U1 setvar "wlan.ip.dhcp.requests_per_session" "2"
```

When the `setvar` value is set to "2", the `getvar` result is "2".

wlan.ip.dhcp.session_interval

This command retrieves how long it will take for a DHCP session to time out before a new DHCP session begins on the wireless print server.

Setvar

To set the DHCP session time out:

```
! U1 setvar "wlan.ip.dhcp.session_interval" "value"
```

Values

`"0"` through `"60"`

Default

`"10"`

Getvar

To return the current setting value:

```
! U1 getvar "wlan.ip.dhcp.session_interval"
```

Example

This `setvar` example shows the value set to `"10"`.

```
! U1 setvar "wlan.ip.dhcp.session_interval" "10"
```

When the `setvar` value is set to `"10"`, the `getvar` result is `"10"`.

wlan.ip.dns.domain

This command sets the value for DNS domains in a WLAN network. Up to five domains are allowed to be set.

Setvar

To set values to be used by the device as DNS domains when `wlan.ipv6.address_type` is `"static"` (IPv6) or `wlan.ip.protocol` is `"permanent"` (IPv4):

```
! U1 setvar "wlan.ip.dns.domain" "zebra.com zebra-lab.lan.com"
```

Getvar

To retrieve a space-delimited list of domains currently in use by the device, up to five domains. The list will contain a mixture of domains received from DHCPv4, DHCPv6, and user-set values depending upon address acquisition settings, using the following rules:

- IPv4 sources are the priority.
- If IPv4 the source provides five domains, there will only be IPv4-sourced domains.
- If there are slots to fit more domains, they will be filled with IPv6 sources.

```
! U1 getvar "wlan.ip.dns.domain"
```

NOTE: Retrieved values are always the values currently in use, which may not match values that were just set, depending on if you are using a static or auto address type.

Example

The following example sets `wlan.ip.dns.domain` to the domains `zebra.com` and `zebra-lab.lan.com`.

These values will be retrieved immediately if `internal_wired.ipv6.address_type` is `"static"` (IPv6) or `wlan.ip.protocol` is `"permanent"` (IPv4).

```
! U1 setvar "wlan.ip.dns.domain" "zebra.com zebra-lab.lan.com"
```

```
! U1 getvar "wlan.ip.dns.domain"
```

Result

```
"zebra.com zebra-lab.lan.com"
```

wlan.ip.dns.domain_user_value

This command displays the value that the user has set for `wlan.ip.dns.domain` with static (IPv6)/ permanent (IPv4) acquisition mode with a WLAN connection.

Getvar

To retrieve the value:

```
! U1 getvar "wlan.ip.dns.domain_user_value"
```

wlan.ip.dns.servers

This command sets a space-delimited list of the domain name servers from a WLAN print server. Up to three addresses may be set. Both IPv4 and IPv6 are supported.

Setvar

To set the list of DNS WLAN print servers, specify a list of space-delimited IP addresses, separated by spaces, to be used as DNS servers when `wlan.ipv6.address_type` is `"static"` (IPv6) or `wlan.ip.protocol` is `"permanent"` (IPv4).

```
! U1 setvar "wlan.ip.dns.servers" "value"
```

Getvar

To retrieve a space-delimited list IP address(es) of DNS server(s), up to three, that are currently in use by the device. The values to be used are a combination of addresses received from DHCPv4, DHCPv6, and user-set values using the following rules:

- At least one spot will be allocated to any enabled IP version.
- IPv4 addresses will take at least two slots if at least two IPv4 addresses are provided.
- User-set values will be included if static/permanent addresses are used.

NOTE: Retrieved values are always the values currently in use, which may not match values that were just set, depending on if you are using a static or auto address type.

```
! U1 getvar "wlan.ip.dns.servers"
```

Example

The following example sets `wlan.dns.servers` to `2001::123:4567:89ab:0:cdef` .

This value will be retrieved immediately if `wlan.ipv6.address_type` is `"static"` (IPv6) or `wlan.ip.protocol` is `"permanent"` (IPv4).

```
! U1 setvar "wlan.ip.dns.servers" "2001::123:4567:89ab:0:cdef"
```

```
! U1 getvar "wlan.ip.dns.servers"
```

Result

```
"2001::123:4567:89ab:0:cdef"
```

wlan.ip.dns.servers_user_value

This command displays the value that the user has set for wlan.ip.dns.servers with static (IPv6)/permanent (IPv4) acquisition mode with a WLAN connection.

Getvar

To retrieve the value:

```
! U1 getvar "wlan.ip.dns.servers_user_value"
```

wlan.ip.gateway

This command instructs the printer to change the wireless print server's gateway address.

IMPORTANT: This setting refers to the gateway address. A set value is ignored if the IP protocol is not set to permanent.

Setvar

To change the wireless printer server's gateway address:

```
! U1 setvar "wlan.ip.gateway" "value"
```

Values

Any valid gateway address

Default

`"0.0.0.0"`

Getvar

To respond with the wireless printer server's gateway address:

```
! U1 getvar "wlan.ip.gateway"
```

Example

This `setvar` example shows the value set to `"10.3.5.1"`.

```
! U1 setvar "wlan.ip.gateway" "10.3.5.1"
```

When the `setvar` value is set to `"10.3.5.1"`, the `getvar` result is `"10.3.5.1"`.

wlan.ip.netmask

This setting refers to the wireless print server's subnet mask address. This value is ignored if the IP protocol is not set to permanent.

Setvar

To change the wireless print servers subnet mask:

```
! U1 setvar "wlan.ip.netmask" "value"
```

Values

Any valid subnet mask.

Default

`"255.255.255.0"`

Getvar

To return the current setting value:

```
! U1 getvar "wlan.ip.netmask"
```

Example

This `setvar` example shows the value set to `"255.255.255.0"`.

```
! U1 setvar "wlan.ip.netmask" "255.255.255.0"
```

When the `setvar` value is set to `"255.255.255.0"`, the `getvar` result is `"255.255.255.0"`

wlan.ip.port

This printer setting refers to the wireless print server's port number that the TCP print service is listening on. Normal TCP communications from the host should be directed to this port.

Setvar

To set the wireless print server's TCP/UDP port number:

```
! U1 setvar "wlan.ip.port" "value"
```

Values

"1" through "65535" (excluding any ports currently used by other services, such as 21, 23, 80, and 515).

Default

"9100"

Getvar

To respond with the wireless printer server's TCP/UDP port number:

```
! U1 getvar "wlan.ip.port"
```

Example

This `setvar` example shows the value set to "9100".

```
! U1 setvar "wlan.ip.port" "9100"
```

When the `setvar` value is set to "9100", the `getvar` result is "9100".

wlan.ip.port_alternate

This command sets the port number of the alternate wlan port.

NOTE: Print servers supporting this command will monitor both the primary port and the alternate port for connections at the same time.

Setvar

To set the alternate wlan port for the print server:

```
! U1 setvar "wlan.ip.port_alternate" "6101"
```

Values

Any valid TCP port address

Default

`"9100"`

Getvar

To return the current alternate wlan port setting:

```
! U1 getvar "wlan.ip.port_alternate"
```

Values

The current port setting.

Example 1

This `setvar` example shows the value set to `"6101"`.

```
U1 setvar "wlan.ip.port_alternate" "6101"
```

Example 2

This setvar example sets the channel mask to use only channels 1,6,11.

```
! U1 setvar "wlan.channel_mask" "0x421"
```

Only channels 1, 6, and 11 will be used by the radio.

wlan.ip.port_json_config

This command determines the TCP port number to listen on for JSON configuration packets.

Setvar

To determine the TCP port number on which to listen for JSON configuration packets:

```
! U1 setvar "wlan.ip.port_json_config" "value"
```

Values

`"0"` disables the port

`"1"` through `"65535"` for port number to listen on

Default

`"9200"`

Getvar

To retrieve the TCP port number which is listening for JSON configuration packets:

```
! U1 getvar "wlan.ip.port_json_config"
```

Example

In this example, the `getvar` command causes the printer to get the TCP port number which is listening for JSON configuration packets.

```
! U1 getvar "wlan.ip.port_json_config"
```

In this example, the `getvar` command causes the printer to get the TCP port number which is listening for JSON configuration packets.

```
! U1 getvar "wlan.ip.port_json_config"
```

wlan.ip.protocol

This command configures the IP addressing method used by the wireless print server.

Setvar

To configure the IP addressing method used by the wireless print server:

```
! U1 setvar "wlan.ip.protocol" "value"
```

Values

- `"bootp"` uses the standard BOOTP addressing method to obtain an IP address and configuration
- `"dhcp"` uses the standard DHCP addressing method to obtain an IP address and configuration for a server specified period of time
- `"rarp"` uses the standard RARP addressing method to obtain an IP address
- `"glean only"` uses the IP address from a PING packet that is sent to its hardware address (unicast address)
- `“permanent"` uses static values assigned through other commands
- `"all"` tries all of the dynamic addressing methods, not permanent, to obtain an IP address

Default

`"all"`

Getvar

To return the value of the currently selected IP protocol used by the wireless print server:

```
! U1 getvar "wlan.ip.protocol"
```

Example

In this example, the `setvar` result is the current programming language that the printer is using.

```
! U1 setvar "wlan.ip.protocol" "bootp"
```

What the `setvar` value is set to is the `getvar` result. In this example, the `getvar` result is `"bootp"`.

wlan.ip.timeout.enable

Use this command to enable the connection timeout on the wireless print server. For this to take effect, the print server must be reset.

Setvar

To enable or disable the timeout checking on the wireless print server:

```
! U1 setvar "wlan.ip.timeout.enable" "value"
```

Values

`"off"` turns off the connection checking

`"on"` turns on the connection checking

Default

`"on"`

Getvar

To return whether the timeout checking is enabled on the wireless print server:

```
! U1 getvar "wlan.ip.timeout.enable"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "wlan.ip.timeout.enable" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`.

wlan.ip.timeout.value

This network setting refers to the number of seconds before the connection times out for the wireless print server. For this to take effect, the print server must be reset.

Setvar

To set the time value of the wireless print server:

```
! U1 setvar "wlan.ip.timeout.value" "value"
```

Values

`"1"` through `"3600"`

Default

`"300"`

Getvar

To respond with the current setting value:

```
! U1 getvar "wlan.ip.timeout.value"
```

Example

This `setvar` example shows the value set to `"300"`.

```
! U1 setvar "wlan.ip.timeout.value" "300"
```

When the `setvar` value is set to `"300"`, the `getvar` result is `"300"`.

wlan.ip.wins.addr

Specifies the WINS server address. If WINS Addressing is done via DHCP, then this value will automatically be filled by the DHCP server.

Setvar

To set the WINS server address:

```
! U1 setvar "wlan.ip.wins.addr" "value"
```

Values

A valid IP address

Default

`"0.0.0.0"`

Getvar

To return the current setting value:

```
! U1 getvar "wlan.ip.wins.addr"
```

wlan.ip.wins.permanent_source

Specifies if the source of the WINS address is DHCP or if it is set manually.

Setvar

To set the source of the WINS address:

```
! U1 setvar "wlan.ip.wins.permanent_source" "value"
```

Values

`"off"`: uses DHCP assigned WINS address

`"on"`: manually sets WINS address (set via `wlan.ip.wins.addr`)

Default

`"off"`

Getvar

To return the current setting value:

```
! U1 getvar "wlan.ip.wins.permanent_source"
```

wlan.ipv6.addresses

This command returns a list of up to ten IPv6 addresses in use by the printer with a WLAN connection when `wlan.ipv6.address_type` is set to `auto`. The command returns`"::"`when no address has been set.

Getvar

To retrieve the IPv6 addresses:

```
! U1 getvar "wlan.ipv6.addresses"
```

wlan.ipv6.address_type

This command controls how the interface gets IPv6 address(es). When set to `"auto"`, the Router Advertisement packets determine the usage of SLAAC and/or DHCP.

Setvar

To set the active network address type:

```
! U1 setvar "wlan.ipv6.address_type" "value"
```

Values

`"static"` or `"auto"`

Default

`"auto"`

Getvar

To retrieve the address type:

```
! U1 getvar "wlan.ipv6.address_type"
```

Example

This example changes the address type to `"static"`.

```
! U1 setvar "wlan.ipv6.address_type" "static"
```

```
! U1 getvar "wlan.ipv6.address_type"
```

Result

```
"static"
```

wlan.ipv6.dhcp.lease.last_attempt

This command retrieves the value of the Unix timestamp of the last attempt to obtain a DHCPv6 lease with a WLAN connection. The value is a whole number up to 4,294,967,295 in seconds.

Getvar

To retrieve the time of the last attempt:

```
! U1 getvar "wlan.ipv6.dhcp.lease.last_attempt"
```

wlan.ipv6.dhcp.lease.length

This command retrieves the value of the lease duration for an IP address provided by the DHCPv6 in seconds with a WLAN connection. The value is a whole number up to 4,294,967,295 in seconds.

Getvar

To retrieve the lease length:

```
! U1 getvar "wlan.ipv6.dhcp.lease.length"
```

wlan.ipv6.dhcp.lease.time_left

This command retrieves the value of the remaining lease duration for an IP address provided by the DHCPv6 in seconds with a WLAN connection. The value is a whole number up to 4,294,967,295 in seconds.

Getvar

To retrieve the lease time left:

```
! U1 getvar "wlan.ipv6.dhcp.lease.time_left"
```

wlan.ipv6.dhcp.option39_enable

This command enables option 39 (to configure the Fully Qualified Domain Name or FQDN) in DHCPv6 communication with a WLAN connection.

Setvar

To enable or disable option 39:

```
! U1 setvar "wlan.ipv6.dhcp.option39_enable" "value"
```

Values

`"on"` or `"off"`

Default

`"on"`

Getvar

To retrieve the option 39 status:

```
! U1 getvar "wlan.ipv6.dhcp.option39_enable"
```

Example

```
! U1 setvar "wlan.ipv6.dhcp.option39_enable" "off"
```

```
! U1 getvar "wlan.ipv6.dhcp.option39_enable"
```

Result

```
"off"
```

wlan.ipv6.dhcp.option39_format

This command sets the value of the format field for option 39 (to configure the Fully Qualified Domain Name or FQDN) in DHCPv6 with a WLAN connection. The value is a string up to 127 characters, and the default is the value set by `device.friendly_name`.

Setvar

To set the format string:

```
! U1 setvar "wlan.ipv6.dhcp.option39_format" "value"
```

where `"value"` is a string of up to 127 characters.

If the value is a source SGD command, it must be bracketed with `"<"and">"`, such as `"<device.friendly_name>"`.

Getvar

To retrieve the string:

```
! U1 getvar "wlan.ipv6.dhcp.option39_format"
```

Example

This command sets the value to `"printer2"`.

```
! U1 setvar "wlan.ipv6.dhcp.option39_format" "printer2"
```

```
! U1 getvar "wlan.ipv6.dhcp.option39_format"
```

Result

```
"printer2"
```

wlan.ipv6.dhcp.option39_fqdn

This command retrieves the fully qualified domain name (FQDN) of the printer as given by the DHCPv6 server for option 39 with a WLAN connection. The value is a string up to 127 characters.

Getvar

To retrieve the FQDN:

```
! U1 getvar "wlan.ipv6.dhcp.option39_fqdn"
```

wlan.ipv6.dhcp.option39_value

This command retrieves the value to be used for option 39 in DHCPv6 after processing the `internal_wired.ipv6.dhcp.option39_format` command with a WLAN connection. The value is a string of up to 127 characters.

Getvar

To retrieve the option 39 value:

```
! U1 getvar "wlan.ipv6.dhcp.option39_value"
```

wlan.ipv6.gateways

This command retrieves the IPv6 gateway(s) for WLAN devices in use by the printer (maximum list of 10).

Getvar

To retrieve the gateway(s):

```
! U1 getvar "wlan.ipv6.gateways"
```

wlan.ipv6.static.addresses

This command specifies the IPv6 address(es) to be used when `wlan.ipv6.address_type` is set to `"static"`. Up to three addresses may be set, separated by commas. The command returns `"::"` when no static address has been set.

Setvar

To set the IPv6 address(es):

```
! U1 setvar "wlan.ipv6.static.addresses" "value"
```

where `"value"` is a string of up to 152 characters.

Getvar

To retrieve the list of IPv6 address(es):

```
! U1 getvar "wlan.ipv6.static.addresses"
```

Example

To set three addresses:

```
! U1 setvar "wlan.ipv6.static.addresses"
 "fc04:1795::fe94:1704/32,fd04:1795::207:4dff:fe94:1704/64,fd04:1796::e0b/64"
```

```
! U1 getvar "wlan.ipv6.static.addresses"
```

Result

```
"fc04:1795::fe94:1704/32,fd04:1795::207:4dff:fe94:1704/64,fd04:1796::e0b/64"
```

wlan.ipv6.static.gateways

This command sets the IPv6 gateway to be used when `wlan.ipv6.address_type` is set to `"static"`. Only one gateway is supported. The command returns `"::"` when no static address has been set.

Setvar

To set the gateway:

```
! U1 setvar "wlan.ipv6.static.gateways" "value"
```

where `"value"` is a string of up to 50 characters. To clear a previously set gateway, use `""` or `"::"`.

Getvar

To retrieve the active network address type:

```
! U1 getvar "wlan.ipv6.static.gateways"
```

Example

To set the gateway to `"fe80::202:b3ff:febf:9d18"`:

```
! U1 setvar "wlan.ipv6.static.gateways" "fe80::202:b3ff:febf:9d18"
```

```
! U1 getvar "wlan.ipv6.static.gateways"
```

Result

```
"fe80::202:b3ff:febf:9d18"
```

wlan.keep_alive.enable

This setting controls the printers ability to send a LSAP (link service access point) packet to the access point on an user controllable interval. This feature is included to accommodate access points that require a regular confirmation that wireless clients are still active.

Setvar

To enable or disable the keep alive printer setting:

```
! U1 setvar "wlan.keep_alive.enable" "value"
```

Values

`"on"` turns on keep_alive

`"off"` turns off keep_alive

Default

`"on"`

Getvar

To return with the current value of the `wlan.keep_alive.enable` setting:

```
! U1 getvar "wlan.keep_alive.enable"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "wlan.keep_alive.enable" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`

wlan.keep_alive.timeout

This printer setting manages the interval at which the LSAP (link service access point) packet is sent.

Setvar

To configure the frequency at which the printer sends the `wlan.keep_alive packet`:

```
! U1 setvar "wlan.keep_alive.timeout" "value"
```

Values
`"5"` to `"300"` seconds

Default

`"15"`

Getvar

To respond with the `wlan.keep_alive.timeout` interval value:

```
! U1 getvar "wlan.keep_alive.timeout"
```

Example

This `setvar` example shows the value set to `"15"`.

```
! U1 setvar "wlan.keep_alive.timeout" "15"
```

When the `setvar` value is set to `"15"`, the `getvar` result is `"15"`.

wlan.kerberos.kdc

This printer setting refers to the Kerberos Key Distribution Center (KDC). The KDC is a trusted server which maintains a database with account information for all security principals (users) for a particular site or administrative domain (realm).

Setvar

To change the Kerberos KDC:

```
! U1 setvar "wlan.kerberos.kdc" "value"
```

Values

"0" to "32" ASCII characters

Default

"krbtgt"

Getvar

To respond with the current Kerberos KDC:

```
! U1 getvar "wlan.kerberos.kdc"
```

Example

This `setvar` example shows the value set to `"krbtgt"`.

```
! U1 setvar "wlan.kerberos.kdc" "krbtgt"
```

When the `setvar` value is set to `"krbtgt"`, the `getvar` result is `"krbtgt"`

wlan.kerberos.mode

This printer setting refers to the Kerberos network authentication protocol. Kerberos provides secure mutual authentication for a wireless client through a Symbol Access Point, based on user information stored on a Kerberos KDC (Key Distribution Center) server.

Setvar

To enable or disable the Kerberos mode:

```
! U1 setvar "wlan.kerberos.mode" "values"
```

Values

`"off"` disables Kerberos mode

`"on"` enables Kerberos mode

Default

`"off"`

Getvar

To respond with the current Kerberos mode:

```
! U1 getvar "wlan.kerberos.mode"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "wlan.kerberos.mode" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`.

wlan.kerberos.password

This printer setting refers to the Kerberos password. The password must correspond to a user profile established on the Kerberos KDC server in use.

Setvar

To set the Kerberos password:

```
! U1 setvar "wlan.kerberos.password" "value"
```

Values

`"0"` through `"32"` alphanumeric characters

Default

`"password"`

Getvar

To respond with the current Kerberos password:

```
! U1 getvar "wlan.kerberos.password"
```

For protection a single `"*"` prints.

Example

This `setvar` example shows the value set to `"password"`.

```
! U1 setvar "wlan.kerberos.password" "password"
```

When the `setvar` value is set to `"password"`, the `getvar` result is `"*"`.

wlan.kerberos.realm

This printer setting refers to the Kerberos realm, an administrative domain with its own Kerberos server (KDC).

IMPORTANT: If you are using a Windows 2000 Server the realm must be all upper-case. For details, see example below.

Setvar

To change the Kerberos realm:

```
! U1 setvar "wlan.kerberos.realm" "value"
```

Values

`"0"` through `"64"` alphanumeric characters

Default

`"kerberos"`

Getvar

To respond with the current Kerberos realm:

```
! U1 getvar "wlan.kerberos.realm"
```

Example

This `setvar` example shows the value set to `"zebra"`.

```
! U1 setvar "wlan.kerberos.realm" "zebra"
```

When the `setvar` value is set to `"zebra"`, the `getvar` result is `"zebra"`.

This `setvar` example shows the value set to "ZEBRA" on a Windows 2000 server.

```
! U1 setvar "wlan.kerberos.realm" "ZEBRA"
```

When the `setvar` value is set to "ZEBRA", the `getvar` result is "ZEBRA".

wlan.kerberos.username

This printer setting refers to the Kerberos user name. The user name must correspond to a user profile established on the Kerberos KDC server in use.

Setvar

To change the Kerberos user name:

```
! U1 setvar "wlan.kerberos.username" "value"
```

Values

`"0"` to `"32"` alphanumeric characters

Default

`"user"`

Getvar

To respond with the current Kerberos user name:

```
! U1 getvar "wlan.kerberos.username"
```

Example

This `setvar` example shows the value set to `"user"`.

```
! U1 setvar "wlan.kerberos.username" "user"
```

When the `setvar` value is set to `"user"`, the `getvar` result is `"user"`.

wlan.leap_mode

This printer setting refers to Cisco LEAP (Lightweight Extensible Authentication Protocol). LEAP provides secure mutual authentication for a wireless client through a Cisco Aironet Access Point, based on user information stored on a backend RADIUS (Remote Authentication in Dial-Up User Service) /AAA (Authentication, Authorization, and Accounting) server.

NOTE:

- This command is only supported on printers using firmware Vxx.18.xx or earlier.
- This command is not supported on units with a Frequency Hopping Spread Spectrum (FHSS) radio.

Setvar

To turn the LEAP mode `"on"` or `"off"`:

```
! U1 setvar "wlan.leap_mode" "values"
```

Values

`"off"` disables LEAP mode

`"on"` enables LEAP mode

Default

`"off"`

Getvar

To respond with the LEAP mode:

```
! U1 getvar "wlan.leap_mode"
```

Example

This `setvar` example shows the value set to `"on"`.

```
! U1 setvar "wlan.leap_mode" "on"
```

When the `setvar` value is set to `"on"`, the `getvar` result is `"on"`.

wlan.leap_password

This printer setting refers to the LEAP password. The password must correspond to a user profile established on the RADIUS/AAA server in use.

NOTE: This parameter is not supported on units with a Frequency Hopping Spread Spectrum (FHSS) radio.

Setvar

This command instructs the printer to change the LEAP password.

```
! U1 setvar "wlan.leap_password" "values"
```

Values
0 to 32 ASCII characters

Default
`"password"`

Getvar

This command instructs the printer to respond with the LEAP password.

```
! U1 getvar "wlan.leap_password"
```

For protection, a single `"*"`prints.

Example

This setvar example shows the value set to `"password"`.

```
! U1 setvar "wlan.leap_password" "password"
```

When the `setvar` value is set to `"password"`, the `getvar` result is `"*"`.

wlan.leap_username

This printer setting refers to the LEAP user name. The user name must correspond to a user profile established on the RADIUS/AAA server in use.

NOTE: This parameter is not supported on units with a Frequency Hopping Spread Spectrum (FHSS) radio.

Setvar

To change the LEAP user name:

```
! U1 setvar "wlan.leap_username" "values"
```

Values

0 to 32 alphanumeric ASCII characters.

Default

`"user"`

Getvar

To respond with the LEAP user name:

```
! U1 getvar "wlan.leap_username"
```

Example

This setvar example shows the value set to `"user"`.

```
! U1 setvar "wlan.leap_username" "user"
```

When the `setvar` value is set to `"user"`, the `getvar` result is `"user"`.

wlan.mac_addr

This command retrieves the MAC address of the wireless print server.

Getvar

To respond with the MAC address of the wireless print server:

```
! U1 getvar "wlan.mac_addr"
```

Example

In this example, the `getvar` result is the MAC address for the wireless print server.

```
! U1 getvar "wlan.mac_addr"
```

wlan.mac_raw

This command specifies the RAW MAC address of the wireless print server. The raw mac address is the mac address without the colons (`":"`).

Getvar

To retrieve the RAW MAC address of the wireless print server:

```
! U1 getvar "wlan.mac_raw"
```

Example

In this example, the `getvar` retrieves the RAW MAC address of the wireless print server.

```
! U1 getvar "wlan.mac_raw"
```

wlan.operating_mode

This printer setting refers to the network operating mode. Infrastructure mode means that the printer will try to associate with an access point. Ad hoc mode means that the printer will try to associate with a device other than an access point and join a standalone network.

To use `"ad hoc"` mode, configure the printer as follows:

- Set the ESSID to the new network's ESSID.
- Turn off the DHCP and assign an IP Address to the printer.
- Set the subnet mask on the printer to the new network's subnet mask.
- Change the operating mode on the printer to `"ad hoc"`.

Setvar

To set the network operating mode:

```
! U1 setvar "wlan.operating_mode" "value"
```

Values

- `"adhoc"` means the printer will try to associate with a network device

`"infrastructure"` means the printer will try to associate with an access point

Getvar

To respond with the network-mode value:

```
! U1 getvar "wlan.operating_mode"
```

Example

This `setvar` example shows the value set to `"infrastructure"`.

```
! U1 setvar "wlan.operating_mode" "infrastructure"
```

When the `setvar` value is set to `"infrastructure"`, the `getvar` result is `"infrastructure"`.

wlan.password

This printer setting refers to the generic password that is used by the wireless securities that need a password.

IMPORTANT: Kerberos has its own password field.

Setvar

To set a generic password for the wireless securities that need a password:

```
! U1 setvar "wlan.password" "value"
```

Values

A maximum of 32 alphanumeric characters.

Default

`"password"`

Getvar

To respond with a generic password for wireless securities:

```
! U1 getvar "wlan.password"
```

For protection a single `"*"` prints.

Example

This `setvar` example shows the value set to `"password"`.

```
! U1 setvar "wlan.password" "password"
```

When the `setvar` value is set to `"password"`, the `getvar` result is `"*"`.

wlan.permitted_channels

This command returns the list of permitted channels.

Getvar

To return the list of permitted channels within the defined country or region:

```
! U1 getvar "wlan.permitted_channels"
```

Result

One or more of the following channels:
`1,2,3,4,5,6,7,8,9,10,11,36,40,44,48,52,56,60,64,100,104,108,112,116,132,136,140`

wlan.pmf

This command allows the user to configure the 802.11 Protected Management Frame settings. This is only applicable for printers having the 802.11ac radio feature.

Setvar

To configure the protected management feature setting:

```
! U1 setvar "wlan.pmf" "value"
```

Values

`"disabled"` means Protected Management Frame is disabled

`"enabled"` means Protected Management Frame is enabled

`"required"` means Printer must support PMF.

Default

`"enabled"`

Getvar

To return the current setting value:

```
! U1 getvar "wlan.pmf"
```

wlan.poor_signal_threshold

Sets or returns the poor signal threshold value. The poor signal threshold is a percentage of the signal strength.

Setvar

```
! U1 setvar "wlan.poor_signal_threshold" "value"
```

Values
"0" to "100"

Default
"0"

Getvar

```
! U1 getvar "wlan.poor_signal_threshold"
```

wlan.preamble

This printer setting selects the radio preamble length to be used.

Setvar

This command instructs the printer to set the preamble length.

```
! U1 setvar "wlan.preamble" "value"
```

Values

`"long"` enables long preamble

`"short"` enables short preamble

Default

`"long"`

Getvar

This command instructs the printer to respond with the current preamble length.

```
! U1 getvar "wlan.preamble"
```

Example

This `setvar` example shows the value set to `"long"`.

```
! U1 setvar "wlan.preamble" "long"
```

When the `setvar` value is set to `"long"`, the `getvar` result is `"long"`.

wlan.private_key_password

This printer setting allows the setting of the optional private key password.

Setvar

This command instructs the printer to set the private key password.

```
! U1 setvar "wlan.private_key_password" "value"
```

Values
A maximum of 32 alphanumeric characters

Default
""

Getvar

This command instructs the printer to respond with the value of the private key password.

```
! U1 getvar "wlan.private_key_password"
```

For protection a single "*"prints.

Example

This `setvar` example shows the value set to `"password"`.

```
! U1 setvar "wlan.private_key_password" "password"
```

When the `setvar` value is set to `"password"`, the `getvar` result is `"*"`.

wlan.region_code

This command defines the regulatory country for which the radio is currently configured.

Setvar

This command sets the region code for which the radio is to be configured.

```
! U1 setvar "wlan.region_code" "value"
```

Values

- "not available"
- "usa/canada"
- "japan"
- "rest of world"#

Getvar

This command retrieves the region code for which the radio is currently configured.

```
! U1 getvar "wlan.country_code"
```

wlan.roam.interchannel_delay

This command sets how long of a delay before scanning the next channel when roaming.

Setvar

This command sets how long of a delay before scanning the next channel when roaming. The values are in milliseconds.

```
! U1 setvar "wlan.roam.interchannel_delay" "value"
```

Values
"0" to "30000"

Default
"400"

Getvar

This command retrieves the current set delay time before scanning the next channel when roaming.

```
! U1 getvar "wlan.roam.interchannel_delay"
```

Example

This `setvar` example shows the value set to "400".

```
! U1 setvar "wlan.roam.interchannel_delay" "400"
```

The `getvar` result returns the current `setvar` value. In this example, the `getvar` result is "400".

wlan.roam.interval

This printer setting refers to specifying the wireless roam interval.

Setvar

This command instructs the printer to set the wireless roam interval.

```
! U1 setvar "wlan.roam.interval"
```

Values
Decimal values between `"5"` and `"255"` inclusive

Default
`"20"`

Getvar

This command instructs the printer to respond with the specified roam interval.

```
! U1 getvar "wlan.roam.interval"
```

Example

This `setvar` example shows the value set to `"20"`.

```
! U1 setvar "wlan.roam.interval" "20"
```

When the `setvar` value is set to `"20"`, the `getvar` result is `"20"`.

wlan.roam.max_chan_scan_time

This command sets how long the radio waits on a channel looking for probe responses.

Setvar

This command sets how long the radio waits on a channel looking for probe responses. The values are in milliseconds.

```
! U1 setvar "wlan.roam.max_chan_scan_time" "value"
```

Values
"10" to "500"

Default
"100"

Getvar

This command retrieves the current setting for how long the radio waits on a channel looking for probe responses.

```
! U1 getvar "wlan.roam.max_chan_scan_time"
```

Example

This `setvar` example shows the value set to `"100"`.

```
! U1 setvar "wlan.roam.max_chan_scan_time" "100"
```

The `getvar` result returns the current `setvar` value. In this example, the `getvar` result is `"100"`.

wlan.roam.max_fail

This command determines teh number of consecutive tx packet failures, at which point th radio should start its roaming algorithm.

Setvar

This command sets the max_fail threshold value.

```
! U1 setvar "wlan.roam.max_fail" "value"
```

Values
"2" to "75" inclusive

Default
"10"

Getvar

This command returns the number for the max_fail threshold.

```
! U1 getvar "wlan.roam.max_fail"
```

Example

In this example, the `setvar` sets the max_fail threshold value to 30 packets.

```
! U1 setvar "wlan.roam.max_fail" "30"
```

wlan.roam.monitor

This command sets the wireless LAN roam monitoring event messages.

Setvar

To instruct the printer to turn off or print the roam event messages:

```
! U1 setvar "wlan.roam.monitor" "value"
```

Values

`"off"`	roam monitor event messages are turned off
`"print"`	roam monitor event messages are printed.
`"serial"`	roam monitor event messages are output to the serial port.
`"file"`	roam monitor event messages are stored in the roam.log file on the E: drive.

Default

`"off"`

Getvar

To return the current setting value:

```
! U1 getvar "wlan.roam.monitor"
```

wlan.roam.neighbor_assist

This command supports 802.11k roaming. When enabled and connected to a network, the printer will query the neighbor list from the Access Point (AP) and use the neighbor AP channel list to reduce the number of channels needed to scan during roaming attempts.

Setvar

To enable `wlan.roam.neighbor_assist`:

```
! U1 setvar "wlan.roam.neighbor_assist" "on"
```

Values

- `on` enables the neighbor query.
- `off` disables the neighbor query.

Default

`off`

Getvar

To return the value of `wlan.roam.neighbor_assist`:

```
! U1 getvar "wlan.roam.neighbor_assist " "on"
```

Values

- `on` the neighbor query is enabled.
- `off` the neighbor query is disabled.

wlan.roam.rssi

This command allows you to specify the absolute value of the negative dBm for the RSSI threshold, which is the point at which the radio will start the roaming algorithm.

Setvar

This command sets the RSSI threshold value.

```
! U1 setvar "wlan.roam.rssi" "value"
```

Values
"60" to "125"

Default
"74"

Getvar

This command retrieves the absolute value of the negative dBM for the RSSI threshold.

```
! U1 getvar "wlan.roam.rssi"
```

Example

In this example, the `setvar` sets the RSSI threshold value to -80 dBm.

```
! U1 setvar "wlan.roam.rssid" "80"
```

wlan.roam.signal

This printer setting refers to specifying the wireless roam signal.

Setvar

To set the wireless roam signal:

```
! U1 setvar "wlan.roam.signal" "value"
```

Values

Decimal values between 1 and 75 inclusive.

Default

`"50"`

Getvar

To respond with the specified wireless roam signal:

```
! U1 getvar "wlan.roam.signal"
```

Example

This `setvar` example shows the value set to `"50"`.

```
! U1 setvar "wlan.roam.signal" "50"
```

When the `setvar` value is set to `"50"`, the `getvar` result is `"50"`.

wlan.rts_cts_enabled

Enables the RTS/CTS HT protection frames when configuring a WLAN connection, preventing interference with other nearby 802.11 signals. If the protection frames are not enabled, the WLAN radio will use CTS-to-self.

Setvar

```
! U1 setvar "wlan.rts_cts_enabled" "value"
```

Values

- `"on"`
- `"off"`

Default

`"off"`

Getvar

```
! U1 getvar "wlan.rts_cts_enabled"
```

Values

- `"on"`
- `"off"`

wlan.security

This printer setting allows you to specify both the wireless encryption type and authentication type in one command.

NOTE: The supporting parameters that are required vary based on the security type that you select. See Supporting SGDs for Different Security Types on page 3 for instructions for each security type.

When using certificate files, Zebra printers support:

- using Privacy Enhanced Mail (PEM) formatted certificate files.
- using the client certificate and private key as two files, each downloaded separately.
- using exportable PAC files for EAP-FAST.

These certificate files can only be sent using ZPL, not SGD. The ZPL command to use when sending these certificate files is the ~DY command.

Configuring the printer for WPA also allows the printer to be used in WPA2 environments.

NOTE: When using certificate files, the time on the printer must be set correctly for the WebSocket connection to succeed, as the time is used in the certificate validation.

Setvar

To set the wireless security value:

```
! U1 setvar "wlan.security" "value"
```

Values

- "1" No wireless security or "none"
- "2" WEP 40-bit or "wep 40-bit"
- "3" WEP 128-bit or "wep 128-bit"
- "4" EAP-TLS or "eap-tls"
- "5" EAP-TTLS or "eap-ttls"
- "6" EAP-FAST or "eap-fast"
- "7" PEAP or "peap"
- "8" LEAP or "leap"
- "9" WPA PSK or "wpa psk" (Key rotation for WPA2 PSK is supported in firmware versions V53.15.8Z , V60.15.8Z, and later.)
- "10" WPA EAP-TLS or "wpa eap-tls"
- "11" WPA EAP-TTLS or "wpa eap-ttls"
- "12" WPA EAP-FAST or "wpa eap-fast"
- "13" WPA PEAP or "wpa peap"
- "14" WPA LEAP or "wpa leap"
- "15" Kerberos or "kerberos"
- "17" WPA SAE or "wpa sae"

Default

"1"

Getvar

To return the name and not the type:

```
! U1 getvar "wlan.security" "value"
```

If an invalid security mode is entered the printer returns "Invalid Mode".

Examples

This setvar example shows the value set to "1".

```
! U1 setvar "wlan.security" "1"
```

When the setvar value is set to "1", the getvar result is "none".

Supporting SGDs for Different Security Types

The supporting SGD commands required for `wlan.security` vary based on the security type that you select. You must send the additional commands for your printer to be able to work on your wireless network. Follow the example and format for your specific security type in this section, substituting your own wireless network data.

Security Type 1: No Wireless Security Active

Additional parameters that need to be set: none

Example

This example turns off all wireless securities controlled under this command, but it does not reset the printer's wireless settings to their defaults.

```
! U1 setvar "wlan.security" "1"
```

Security Type 2: WEP 40-Bit

Additional parameters that need to be set and the SGD commands to use:

- WEP encryption index (see `wlan.wep.index`)
- WEP authentication type (see `wlan.wep.auth_type`)
- WEP key type (see `wlan.wep.key_format`)
- the actual values of any WEP encryption keys to be used (see `wlan.wep.key1` , `wlan.wep.key2`, `wlan.wep.key3`, or `wlan.wep.key4`)

Example

This example configures the printer for WEP 40-bit encryption using index key 1, open authentication, and a hexadecimal WEP key with a value of "A1B2C3D4F5".

```
! U1 setvar "wlan.security" "2"
```

```
! U1 setvar "wlan.wep.index" "1"
```

```
! U1 setvar "wlan.wep.auth_type" "open"
```

```
! U1 setvar "wlan.wep.key_format" "hex"
```

```
! U1 setvar "wlan.wep.key1" "A1B2C3D4F5"
```

Security Type 3: WEP 128-Bit

Additional parameters that need to be set and the SGD commands to use:

- WEP encryption index (see `wlan.wep.index`)

- WEP authentication type (see `wlan.wep.auth_type`)
- WEP key type (see `wlan.wep.key_format`)
- the actual values of any WEP encryption keys to be used (see `wlan.wep.key1`, `wlan.wep.key2`, `wlan.wep.key3`, or `wlan.wep.key4`)

Example

This example configures the printer for WEP 128-bit encryption using index key 2, open authentication, and four hexadecimal WEP keys.

```
! U1 setvar "wlan.security" "3"
```

```
! U1 setvar "wlan.wep.index" "2"
```

```
! U1 setvar "wlan.wep.auth_type" "open"
```

```
! U1 setvar "wlan.wep.key_format" "hex"
```

```
! U1 setvar "wlan.wep.key1" "001122334455667788"
```

```
! U1 setvar "wlan.wep.key2" "112233445566778899"
```

```
! U1 setvar "wlan.wep.key3" "223344556677889900"
```

```
! U1 setvar "wlan.wep.key4" "334455667788990011"
```

Security Type 4: EAP-TLS

Additional parameters that need to be set and the SGD commands to use:

- optional private key password (see `wlan.private_key_password`)

Example

This example configures the printer for EAP-TLS authentication with an optional private key password with a value of "`private`."

```
! U1 setvar "wlan.security" "4"
```

```
! U1 setvar "wlan.private_key_password" "private"
```

Security Type 5: EAP-TTLS

Additional parameters that need to be set and the SGD commands to use:

- user ID (see `wlan.username`)
- password (see `wlan.password`)

Example

With a value of `"user"` and a password with a value of `"password"`.

```
! U1 setvar "wlan.security" "5"
```

```
! U1 setvar "wlan.username" "user"
```

```
! U1 setvar "wlan.password" "password"
```

Security Type 6: EAP-FAST

Additional parameters that need to be set and the SGD commands to use:

- user ID (see `wlan.username`)
- password (see `wlan.password`)
- optional private key password (see `wlan.private_key_password`)

Example

This example configures the printer for EAP-FAST authentication, including a user ID of `"user"`, a password of `"password"`, and an optional private key of `"private"`.

```
! U1 setvar "wlan.security" "6"
```

```
! U1 setvar "wlan.username" "user"
```

```
! U1 setvar "wlan.password" "password"
```

```
! U1 setvar "wlan.private_key_password" "private"
```

Security Type 7: PEAP

Additional parameters that need to be set and the SGD commands to use:

- user ID (see `wlan.username`)
- password (see `wlan.password`)

Example

This example configures the printer for PEAP authentication, including a user ID with a value of `"user"` and a password with a value of `"password"`.

```
! U1 setvar "wlan.security" "7"
```

```
! U1 setvar "wlan.username" "user"
```

```
! U1 setvar "wlan.password" "password"
```

Security Type 8: LEAP

Additional parameters that need to be set and the SGD commands to use:

- user ID (see `wlan.username`)
- password (see `wlan.password`)

Example

This example configures the printer for LEAP authentication, including a user ID with a value of `"user"` and a password with a value of `"password"`.

```
! U1 setvar "wlan.security" "8"
```

```
! U1 setvar "wlan.username" "user"
```

```
! U1 setvar "wlan.password" "password"
```

Security Type 9: WPA PSK

NOTE: Configuring the printer for WPA also allows the printer to be used in WPA2 environments. Key rotation for WPA2 PSK is supported in firmware version 60.15.8Z and later and in firmware version 53.15.8Z and later.

Additional parameters that need to be set and the SGD commands to use:

- Pre-Shared Key (PSK) value (see `wlan.wpa.psk`)

Example

This example configures the printer for WPA PSK authentication with a PSK value of all zeroes (64 hexadecimal digits).

```
! U1 setvar "wlan.security" "9"
```

```
! U1 setvar "wlan.wpa.psk" "00000000..."
```

Security Type 10: WPA EAP-TLS

NOTE: Configuring the printer for WPA also allows the printer to be used in WPA2 environments.

Additional parameters that need to be set and the SGD commands to use:

- optional private key password (see `wlan.private_key_password`)

Example

This example configures the printer for WPA EAP-TLS authentication with an optional private key password with a value of `"private"`.

```
! U1 setvar "wlan.security" "10"
```

```
! U1 setvar "wlan.private_key_password" "private"
```

Security Type 11: WPA EAP-TTLS

NOTE: Configuring the printer for WPA also allows the printer to be used in WPA2 environments.

Additional parameters that need to be set and the SGD commands to use:

- user ID (see `wlan.username`)
- password (see `wlan.password`)

Example

This example configures the printer for WPA EAP-TTLS authentication, including a user ID with a value of `"user"`" and a password with a value of `"password"`.

```
! U1 setvar "wlan.security" "11"
```

```
! U1 setvar "wlan.username" "user"
```

```
! U1 setvar "wlan.password" "password"
```

Security Type 12: WPA EAP-FAST

NOTE: Configuring the printer for WPA also allows the printer to be used in WPA2 environments.

Additional parameters that need to be set and the SGD commands to use:

- user ID (see `wlan.username`)
- password (see `wlan.password`)
- optional private key password (see `wlan.private_key_password`)

Example

This example configures the printer for WPA EAP-FAST authentication, including a user ID of "user", a password of "password", and an optional private key of "private".

```
! U1 setvar "wlan.security" "12"
```

```
! U1 setvar "wlan.username" "user"
```

```
! U1 setvar "wlan.password" "password"
```

```
! U1 setvar "wlan.private_key_password" "private"
```

Security Type 13: WPA PEAP

NOTE: Configuring the printer for WPA also allows the printer to be used in WPA2 environments.

Additional parameters that need to be set and the SGD commands to use:

- user ID (see wlan.username)
- password (see wlan.password)

Example

This example configures the printer for WPA PEAP authentication, including a user ID with a value of "user" and a password with a value of "password".

```
! U1 setvar "wlan.security" "13"
```

```
! U1 setvar "wlan.username" "user"
```

```
! U1 setvar "wlan.password" "password"
```

Security Type 14: WPA LEAP

NOTE: Configuring the printer for WPA also allows the printer to be used in WPA2 environments.

Additional parameters that need to be set and the SGD commands to use:

- user ID (see wlan.username)
- password (see wlan.password)

Example

This example configures the printer for WPA LEAP authentication, including a user ID with a value of "user" and a password with a value of "password".

```
! U1 setvar "wlan.security" "14"
```

```
! U1 setvar "wlan.username" "user"
```

```
! U1 setvar "wlan.password" "password"
```

Security Type 15: Kerberos

Additional parameters that need to be set and the SGD commands to use:

- Kerberos user ID (see `wlan.kerberos.username`)
- Kerberos password (see `wlan.kerberos.password`)
- realm (see `wlan.kerberos.realm`)
- Key Distribution Center (KDC) (see `wlan.kerberos.kdc`)

Example

This example configures the printer for Kerberos encryption, including a Kerberos user ID with a value of `"user"`, a Kerberos password with a value of `"password"`, a realm of `"zebra"`, and a KDC of `"krbtgt"`.

```
! U1 setvar "wlan.security" "15"
```

```
! U1 setvar "wlan.kerberos.username" "user"
```

```
! U1 setvar "wlan.kerberos.password" "password"
```

```
! U1 setvar "wlan.kerberos.realm" "zebra"
```

```
! U1 setvar "wlan.kerberos.kdc" "krbtgt"
```

wlan.signal_noise

This command returns the signal noise on the wireless network. Values above 40% represent a very significant noise, and radio communication is not reliable.

Getvar

To return the current signal noise on the wireless network:

```
! U1 getvar "wlan.signal_noise"
```

Example

In this example, the `getvar` result is the current signal_noise value.

```
! U1 getvar "wlan.signal_noise"
```

wlan.signal_quality

This command instructs the printer to return the current signal quality of the wireless network. Values below 40% represent a very poor signal quality, and radio communication is not reliable.

Getvar

To return the current signal quality of the wireless network:

```
! U1 getvar "wlan.signal_quality"
```

Example

In this example, the `getvar` result is the current signal_quality value.

```
! U1 getvar "wlan.signal_quality"
```

wlan.signal_strength

This command returns the signal strength of the connection to the access point as a percentage value between zero (not connected) and 100 (strongest signal). Values below 40% represent a very poor signal and radio communication is not reliable.

Getvar

To respond with the current signal strength:

```
! U1 getvar "wlan.signal_strength"
```

Example

In this example, the `getvar` result is `"93"`.

```
! U1 getvar "wlan.signal_strength"
```

wlan.station_name

This printer setting refers to the station name.

Setvar

To set the station name:

```
! U1 setvar "wlan.station_name" "value"
```

Values

A maximum of 32 alphanumeric characters

Default

"ZEBRA"

Getvar

To respond with the station name value:

```
! U1 getvar "wlan.station_name"
```

Example

This `setvar` example shows the value set to "ZEBRA".

```
! U1 setvar "wlan.station_name" "ZEBRA"
```

When the `setvar` value is set to "ZEBRA", the `getvar` result is "ZEBRA".

wlan.translation_disable_clear

This command clears the Transition Disable setting value saved in the printer.

Setvar

To clear the Transition Disable setting in the printer:

```
! U1 setvar "wlan.transition_disable_clear" ""
```

wlan.tx_power

Use this command to specify the wireless transmit power.

Setvar

To set the wireless transmit power:

```
! U1 setvar "wlan.tx_power" "value"
```

Values

Decimal values of `"1"`, `"5"`, `"20"`, `"30"`, `"50"`, `"100"`

Default

`"100"`

Getvar

To return with the wireless transmit power value:

```
! U1 getvar "wlan.tx_power"
```

Example

This `setvar` example shows the value set to `"100"`.

```
! U1 setvar "wlan.tx_power" "100"
```

When the `setvar` value is set to `"100"`, the `getvar` result is `"100"`.

wlan.tx_rate

Use this command to specify the wireless transmit rate.

Setvar

To set the wireless transmit rate:

```
! U1 setvar "wlan.tx_rate" "value"
```

Values

`"1"`, `"2"`, `"5.5"`, `"11"`, `"all"`

Default

`"all"`

Getvar

To respond with the wireless transmit rate:

```
! U1 getvar "wlan.tx_rate"
```

Example

This `setvar` example shows the value set to `"all"`.

```
! U1 setvar "wlan.tx_rate" "all"
```

When the `setvar` value is set to `"all"`, the `getvar` result is `"all"`.

wlan.user_channel_list

This command sets the list of available channels.

Setvar

To set the list of available channels:

```
! U1 setvar "wlan.user_channel_list" "value"
```

Values

One or more of the following:

```
"1,2,3,4,5,6,7,8,9,10,11,36,40,44,48,52,56,60,64,100,104,108,112,116,132,136,140
"all"
```

Default

```
"all"
```

Getvar

To retrieve the currently set list of available channels:

```
! U1 getvar wlan.user_channel_list"
```

Example

This example sets the available channel list to channels 1-9.

```
! U1 setvar "wlan.user_channel_list" "1,2,3,4,5,6,7,8,9"
```

wlan.username

This printer setting refers to the generic user name that is used by the wireless securities that need a user name.

IMPORTANT: Kerberos has its own user name field.

Setvar

To set a generic user name for wireless securities that need a user name:

```
! U1 setvar "wlan.username" "value"
```

Values

A maximum of 32 alphanumeric characters

Default

`"user"`

Getvar

To respond with a generic user name for the wireless securities that need a user name:

```
! U1 getvar "wlan.username"
```

Example

This `setvar` example shows the value set to `"user"`.

```
! U1 setvar "wlan.username" "user"
```

When the `setvar` value is set to `"user"`, the `getvar` result is `"user"`.

wlan.wep.auth_type

For the WEP security type, this printer setting selects the authentication type to be used between the printer and the access point. The authentication types are open system and shared key.

NOTE: This command is not supported for printers running Link-OS 6 or later versions.

Setvar

To set the WEP authentication type:

```
! U1 setvar "wlan.wep.auth_type" "value"
```

Values

`"open"` enables the open authentication type

`"shared"` enables the shared authentication type

Getvar

To retrieve the current WEP authentication type:

```
! U1 getvar "wlan.wep.auth_type"
```

Example

This `setvar` example shows the value set to `"open"`.

```
! U1 setvar "wlan.wep.auth_type" "open"
```

When the `setvar` value is set to `"open"`, the `getvar` result is `"open"`.

wlan.wep.index

This command sets the WEP (Wired Equivalent Privacy) encryption key index. This printer setting determines which one of the four encryption keys is to be used by the client (printer).

NOTE: This command is not supported for printers running Link-OS 6 or later versions.

Setvar

To set the encryption key index:

```
! U1 setvar "wlan.wep.index" "value"
```

Values

- `"1"` enables encryption key 1
- `"2"` enables encryption key 2
- `"3"` enables encryption key 3
- `"4"` enables encryption key 4

Default

`"1"`

Getvar

To respond with the encryption key index:

```
! U1 getvar "wlan.wep.index"
```

Example

This `setvar` example shows the value set to `"1"`.

```
! U1 setvar "wlan.wep.index" "1"
```

When the `setvar` value is set to `"1"`, the `getvar` result is `"1"`.

wlan.wep.key1

Use this command to set the first indexed WEP encryption key. The WEP encryption key is a hexadecimal or string value. This key should match the wireless network WEP encryption key 1.

NOTE: This command is not supported for printers running Link-OS 6 or later versions.

Setvar

To set the encryption key:

```
! U1 setvar "wlan.wep.key1" "value"
```

Values

- 10 hexadecimal characters for 40-bit encryption
- 26 hexadecimal characters for 128-bit encryption

Default

All zeros

Getvar

To instruct the printer to respond with the encryption key:

```
! U1 getvar "wlan.wep.key1"
```

For protection a single `"*"` prints.

Example

This `setvar` example shows the value set to `"A1B2C3D4F5"`.

```
! U1 setvar "wlan.wep.key1" "A1B2C3D4F5"
```

When the `setvar` value is set to `"A1B2C3D4F5"`, the `getvar` result is `"*"`.

wlan.wep.key2

Use this command to set the second indexed WEP encryption key. The WEP encryption key is a hexadecimal string value. This key should match the wireless network WEP encryption key 2.

NOTE: This command is not supported for printers running Link-OS 6 or later versions.

Setvar

To set the encryption key:

```
! U1 setvar "wlan.wep.key2" "value"
```

Values

- 10 hexadecimal characters for 40-bit encryption
- 26 hexadecimal characters for 128-bit encryption

Default

All zeros

Getvar

To instruct the printer to respond with the encryption key:

```
! U1 getvar "wlan.wep.key2"
```

For protection a single `"*"` prints.

Example

This `setvar` example shows the value set to `"A1B2C3D4F5"`.

```
! U1 setvar "wlan.wep.key2" "A1B2C3D4F5"
```

When the `setvar` value is set to `"A1B2C3D4F5"`, the `getvar` result is `"*"`.

wlan.wep.key3

Use this command to set the third indexed WEP encryption key. The WEP encryption key is a hexadecimal string value. This key should match the wireless network WEP encryption key 3.

NOTE: This command is not supported for printers running Link-OS 6 or later versions.

Setvar

To set the encryption key:

```
! U1 setvar "wlan.wep.key3" "value"
```

Values

- 10 hexadecimal characters for 40-bit encryption
- 26 hexadecimal characters for 128-bit encryption

Default

All zeros

Getvar

To instruct the printer to respond with the encryption key:

```
! U1 getvar "wlan.wep.key3"
```

For protection a single `"*"` prints.

Example

This `setvar` example shows the value set to `"A1B2C3D4F5"`.

```
! U1 setvar "wlan.wep.key3" "A1B2C3D4F5"
```

When the `setvar` value is set to `"A1B2C3D4F5"`, the `getvar` result is `"*"`.

wlan.wep.key4

Use this command to set the fourth indexed WEP encryption key. The WEP encryption key is a hexadecimal string value. This key should match the wireless network WEP encryption key 4.

NOTE: This command is not supported for printers running Link OS 6 or later versions.

Setvar

To set the encryption key:

```
! U1 setvar "wlan.wep.key4" "value"
```

Values

10 hexadecimal characters for 40-bit encryption

26 hexadecimal characters for 128-bit encryption

Default

All zeros

Getvar

To respond with the encryption key:

```
! U1 getvar "wlan.wep.key4"
```

For protection a single `"*"` prints.

Example

This `setvar` example shows the value set to `"A1B2C3D4F5"`.

```
! U1 setvar "wlan.wep.key4" "A1B2C3D4F5"
```

When the `setvar` value is set to `"A1B2C3D4F5"`, the `getvar` result is `"*"`.

wlan.wpa.psk

This printer setting specifies the pre-shared key (PSK) value to use when the WPA authentication is set to PSK.

Setvar

To set the pre-shared key:

```
! U1 setvar "wlan.wpa.psk" "value"
```

Values

64 hexadecimal digits

Default

64 zeros (00000000...)

Getvar

To return the pre-shared key value:

```
! U1 getvar "wlan.wpa.psk"
```

For protection a single `"*"` prints.

Example

This `setvar` example shows the value set to `"00000000..."`.

```
! U1 setvar "wlan.wpa.psk" "00000000..."
```

When the `setvar` value is set to `"00000000..."`, the `getvar` result is `"*"`.

wlan.wep.key_format

This printer setting specifies the format for the WEP key.

This command is disabled for Link OS 6 printers and later versions.

NOTE: This printer setting should proceed any of the `wep.key` settings if you select a non-default value.

Setvar

To set the WEP key format:

```
! U1 setvar "wlan.wep.key_format" "value"
```

Values

- `"ascii"` the WEP key is set by ASCII string
- `"hex"` the WEP key is a Hex string

Default

`"hex"`

Getvar

To respond with the WEP key format:

```
! U1 getvar "wep.key_format"
```

Example

This `setvar` example shows the value set to `"ascii"`.

```
! U1 setvar "wlan.wep.key_format" "ascii"
```

When the `setvar` value is set to `"ascii"`, the `getvar` result is `"ascii"`.

wlan.wpa.groupkey_ciphersuite

This command returns the encryption method currently used for unicast packets.

Getvar

To return the current encryption method value:

```
! U1 getvar "wlan.wpa.groupkey_ciphersuite"
```

Result

`"NONE"`	No encryption being used.
`"WEP40"`	WEP40 encryption being used.
`"TKIP"`	TKIP encryption being used.
`"AES"`	AES being used.
`"WEP104"`	WEP104 encryption being used.
`"WPA2"`	WEP104 encryption being used.
`""`	Reported if printer is not yet associated with the wireless LAN.

wlan.wpa.pairwise_ciphersuite

This command returns the encryption method currently used for unicast packets.

Getvar

To return the value of the current encryption methods:

```
! U1 getvar "wlan.wpa.pairwise_ciphersuite"
```

Result

"NONE"	No encryption being used.
"WEP40"	WEP40 encryption being used.
"TKIP"	TKIP encryption being used
"AES"	AES being used.
"WEP104"	WEP104 encryption being used.
"WPA2"	WPA2 encryption being used.
""	Reported if printer is not yet associated with the wireless LAN.

wlan.wpa.timecheck

Allows the user to disable the certificate timestamp check that is performed during a WPA TLS handshake.

Setvar

To enable or disable the certificate timestamp check that is performed during a WPA TLS handshake:

```
! U1 setvar "wlan.wpa.timecheck" "value"
```

Values

- `"yes"` means the timecheck during the handshake will be performed
- `"no"` means the timecheck during the handshake will not be performed

Default

`"yes"`

Getvar

To return the current setting value:

```
! U1 getvar "wlan.wpa.timecheck"
```

wlan.wpa.wpa_version

Returns the currently active WPA version.

Getvar

To return the currently active WPA version:

```
! U1 getvar "wlan.wpa.wpa_version"
```

Values

"WPA" indicates WPA is being used.

"WPA2" indicates WPA2 is being used.

"WPA3"indicates that WPA3 is being used.

"" reported if the WLAN is not connected or WPA is not enabled.

SGD RFID Commands

This section contains the SGD commands for RFID-specific applications.

rfid.antenna_sweep

This command enables/disables the antenna sweep feature.

If the RFID media loaded in the printer is known to be in range of an antenna at the F0 programming position, you can avoid RFID calibration by using the RFID antenna sweep feature. With this feature enabled, when the first RFID format is sent after a printer powerup or printhead close, the printer scans through the antennas to find the optimal antenna element.

NOTE:

- This command is valid only on ZT400 and ZT600 series RFID printers.
- The label length must be 2 in. (51 mm) or greater, and the programming position must be F0.

Setvar

To enable or disable the antenna sweep feature:

```
! U1 setvar "rfid.antenna_sweep" "value"
```

Values

- "on"
- "off"

Default

"off"

Getvar

To retrieve the current antenna sweep setting:

```
! U1 getvar "rfid.adaptive_sweep"
```

rfid.country_code

This command sets or returns the RFID reader country code. The country code is restricted based on the region code assigned to the reader and, in some instances, cannot be modified. You can check the countries available for your region through the control panel menu items on your printer.

Setvar

To set the RFID reader's current country code

```
! U1 setvar "rfid.country_code" "value"
```

Values

The country code choices available vary depending on the region for which your printer is configured.

Getvar

To retrieve the RFID reader's current country code:

```
! U1 getvar "rfid.country_code"
```

Example

In this example, the `setvar` sets the country code to USA/Canada.

```
! U1 setvar "rfid.country_code" "usa/canada"
```

rfid.enable

This command instructs an RFID printer to enable or disable RFID functionality. You must restart the printer for the command to take effect.

NOTE: When this function is set to `"on"`, changes are made to normal printer functionality.
Loading printer defaults does NOT:

- Default the sensor select setting
- Default media tracking sensor settings
- Default label length
- Perform an auto calibration

Setvar

To enable or disable RFID functionality:

```
! U1 setvar "rfid.enable" "value"
```

Values

`"on"` enables RFID functionality

`"off"` disables RFID functionality

Default

`"on"`

rfid.error.response

This command can be used to retrieve the RFID status, including any error codes or messages.

Getvar

To retrieve any active RFID error messages:

```
! U1 getvar "rfid.error.response"
```

Example

This `getvar` example shows responses that you may get in different situations:

```
! U1 getvar "rfid.error.response"
```

If no RFID tag is present, you get the following response:

```
"NO TAG FOUND"
```

If an RFID tag is present and there are no errors, you get the following response:

```
"RFID OK"
```

rfid.hop_table_version

This command retrieves the RFID reader's hop table version.

Getvar

To retrieve the RFID reader's hop table version:

```
! U1 getvar "rfid.hop_table_version"
```

rfid.position.program

This command sets the read/write position of the RFID tag (programming position).

IMPORTANT: If this command is used to specify a value for the programming position, this value will be used for the programming position for all labels until a new position is specified or until the tag calibration procedure is run.

Setvar

This command instructs the printer to set the read/write position of the RFID tag.

```
! U1 setvar "rfid.position.program" "value"
```

Values

- "F0" to "Fxxx" (where xxx is the label length in millimeters or "999", whichever is less)

  The printer prints the first part of a label until it reaches the specified distance and then begins programming. After programming, the printer prints the remainder of the label.

- "B0" to "B30"

  The printer backfeeds the label for the specified distance and then begins programming. To account for the backfeed, allow empty media liner to extend out of the front of the printer when using a backward programming position.

- "up" move to the next value
- "down" move to the previous value

Absolute Mode (all firmware versions)

- xxxx=0 to label length (in dot rows). Move the media to the specified position xxxx on the label, measured in dot rows from the label top, before encoding. Set to 0 (no movement) if the tag is already in the effective area without moving the media.

Relative Mode (firmware versions V53.17.6 and later)

- "F0" to "Fxxx" (where xxx is the label length in millimeters or 999, whichever is less)

  The printer prints the first part of a label until it reaches the specified distance and then begins programming. After programming, the printer prints the remainder of the label.

- "B0" to "B30" (Does not apply to the RP4T printer.)

  The printer backfeeds the label for the specified distance and then begins programming. To account for the backfeed, allow empty media liner to extend out of the front of the printer when using a backward programming position.

Default

- For printers using V53.17.7 and later: "F0" (which moves the leading edge of the label to the print line)
- For the R2844-Z and RPAX: "0" (no movement)
- For all other printers or firmware: label length minus 1 mm (1/16 in.)

Getvar

This command instructs the printer to respond with the current programming position.

```
! U1 getvar "rfid.position.program"
```

Example

This example shows the programming position being set at 15 mm from the leading edge of the label.

```
! U1 setvar "rfid.position.program" "F15"
```

When the `setvar` value is set to `"F15"`, the `getvar` result is `"F15"`.

rfid.reader_1.antenna_port

This command specifies the RFID antenna to be used for RFID operation.

NOTE:

- This applies only to ZT400 and ZT600 series RFID printers, which have multiple antenna elements. Other printers, which only have one antenna element, always use an antenna element value of A1.
- Printers automatically select the best antenna element and read/write power levels for the media during RFID transponder calibration. The ZT400 and ZT600 series printers also may set the levels during an adaptive antenna sweep. Use ^HL or ~HL on page 412 to view the antenna element and power settings being used.

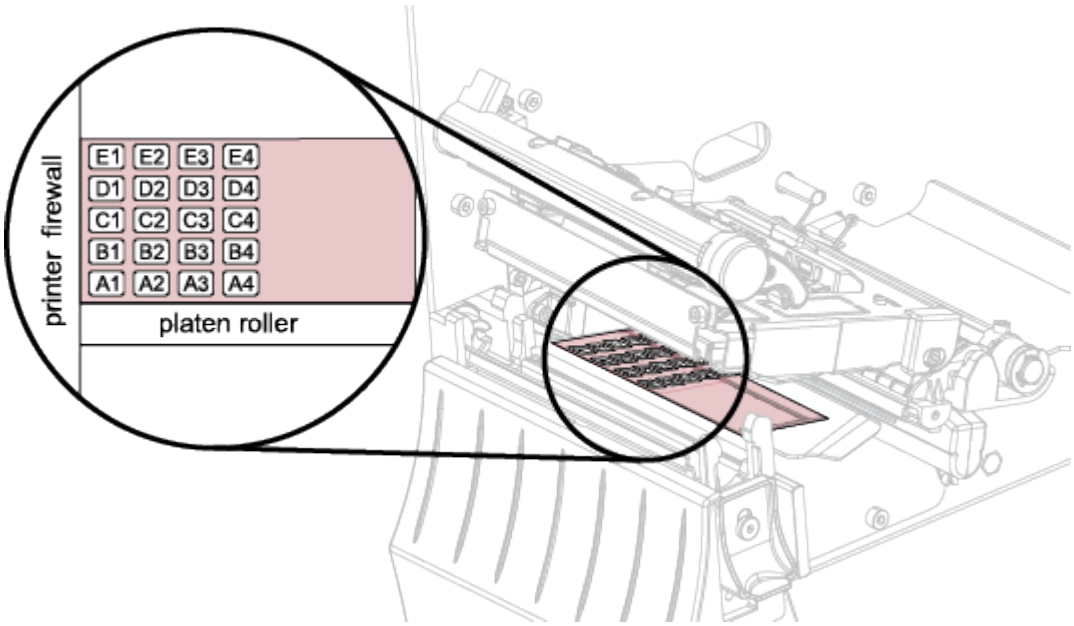
Setvar

Sets the antenna port.

```
! U1 setvar "rfid.reader_1.antenna_port" "value"
```

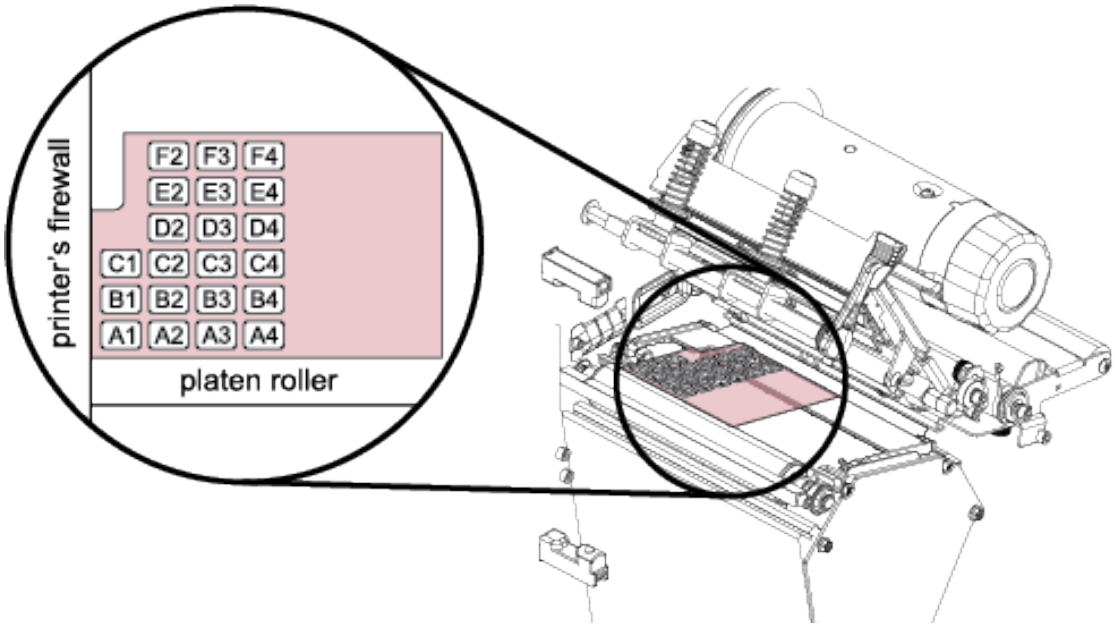
Values (ZT400 and ZT600 Series only)

E1	E2	E3	E4
D1	D2	D3	D4
C1	C2	C3	C4
B1	B2	B3	B4
A1	A2	A3	A4



Values (R110Xi4 only)

NA	F2	F3	F4
NA	E2	E3	E4
NA	D2	D3	D4
C1	C2	C3	C4
A1	A2	A3	A4



Default

"A4"

Getvar

Retrieves the current antenna port.

```
! U1 getvar "rfid.reader_1.antenna_port"
```

Example

This `setvar` example shows the selection of antenna port D3.

```
! U1 setvar "rfid.reader_1.antenna_port" "D3"
```

When the `setvar` value is set to `"D3"`, the `getvar` result is `"D3"`.

rfid.reader_1.power.read

This command sets the RFID reader power level for reading RFID tags.

NOTE: NOTE: Printers automatically select the best antenna element and read/write power levels for the media during RFID transponder calibration. The ZT400 and ZT600 series printers also may set the levels during an adaptive antenna sweep. Use ^HL or ~HL on page 412 to view the antenna element and power settings being used.

Setvar

Instructs the printer to set the read power level of the antenna.

```
! U1 setvar "rfid.reader_1.power.read" "value"
```

Values

RP4T (all firmware versions), R53.16.4Z, V53.17.7, V74.19.6Z, and all Link-OS printers:

- `"0"` to `"30"`
- `"up"` Increase the current value by 1
- `"down"` Decrease the current value by 1

R53.16.3Z
`"0"` to `"30"`

R60.16.x, R62.16.x, R63.16.x, R65.16.x, SP994Q, SP999G, SP1027G, SP1056F, SP1082G, and later

- `"0"` to `"30"`
- `"high"`
- `"medium"`
- `"low"`

Older firmware

- `"high"`
- `"medium"`
- `"low"`

Default

RP4T (all firmware versions), R53.16.4Z, V53.17.7, V74.19.6Z, and all Link-OS printers
`"16"`

R53.16.3Z
`"16"`

R60.16.x, R62.16.x, R63.16.x, R65.16.x, SP994Q, SP999G, SP1027G, SP1056F, SP1082G, and later
`"low"`

Older firmware
`"low"`

Getvar

To return the antenna's read power level:

```
! U1 getvar "rfid.reader_1.power.read"
```

Example

This `setvar` example sets the antenna to power setting 16 for reading RFID tags.

```
! U1 setvar "rfid.reader_1.power.read" "16"
```

When the `setvar` value is set to `"16"`, the `getvar` result is `16`.

rfid.reader_1.power.write

Use this command to set the RFID write power levels if the desired levels are not achieved through RFID tag calibration. If not enough power is applied, the tag may not have sufficient power for programming, and tag data will fail to encode. If too much power is applied, the extra power may cause data communication errors or may cause the wrong tag to be programmed.

NOTE: Printers automatically select the best antenna element and read/write power levels for the media during RFID transponder calibration. The ZT400 and ZT600 series printers also may set the levels during an adaptive antenna sweep.

NOTE: This parameter is ignored on the R110Xi HF printer because read and write powers cannot be specified separately. See to set the power level for the R110Xi HF printer.

Setvar

Instructs the printer to set the antenna's write power level.

```
! U1 setvar "rfid.reader_1.power.write" "value"
```

Values

RP4T (all firmware versions), R53.16.4Z, V53.17.7, V74.19.6Z, and all Link-OS printers

- `"0"` to `"30"`
- `"up"` Increase the current value by 1
- `"down"` Decrease the current value by 1

R53.16.3Z

`"0"` to `"30"`

R60.16.x, R62.16.x, R63.16.x, R65.16.x, SP994Q, SP999G, SP1027G, SP1056F, SP1082G, and later

- `"0"` to `"30"`
- `"high"`
- `"medium"`
- `"low"`

Older firmware

- `"high"`
- `"medium"`
- `"low"`

Default

RP4T (all firmware versions), R53.16.4Z, V53.17.7, V74.19.6Z, and all Link-OS printers
`"16"`

R53.16.3Z
`"16"`

R60.16.x, R62.16.x, R63.16.x, R65.16.x, SP994Q, SP999G, SP1027G, SP1056F, SP1082G, and later
`"low"`

Older firmware
`"low"`

Getvar

To return the antenna's write power level:

```
! U1 getvar "rfid.reader_1.power.write"
```

Example

This `setvar` example sets the antenna to power setting 16 for writing RFID tags.

```
! U1 setvar "rfid.reader_1.power.write" "16"
```

When the `setvar` value is set to `"16"`, the `getvar` result is `"16"`.

rfid.reader_1.power.single_power

This command sets the RFID reader power level for reading and writing to RFID tags for readers with a single power level.

NOTE: This command applies only to the R110Xi HF printer, firmware version R65.X.

Setvar

To set the power level for reading and writing:

```
! U1 setvar "rfid.reader_1.power.single_power" "value"
```

Values

- `"high"`
- `"medium"`
- `"low"`

Default

`"low"`

Getvar

To respond with the current power level:

```
! U1 getvar "rfid.reader_1.power.single_power"
```

This `setvar` example sets the antenna to high power for writing to RFID tags.

Example

```
! U1 setvar "rfid.reader_1.power.single_power" "high"
```

When the `setvar` value is set to `"high"`, the `getvar` result is `"high"`.

rfid.reader_1.firmware_version

This command returns the RFID reader firmware version.

Getvar

To return the RFID reader firmware version:

```
! U1 getvar "rfid.reader_1.firmware_version"
```

Example

This example shows the response you receive when an RFID reader is attached:

```
! U1 getvar "rfid.reader_1.firmware_version"
```

If an RFID reader is present and connected, you get the firmware version in the following format:

`"xx.xx.xx.xx"`

If there is no RFID reader or if the reader is not connected correctly, the response is blank.

rfid.reader_1.hardware_version

This command returns the RFID reader hardware version.

Getvar

To return the RFID reader hardware version:

```
! U1 getvar "rfid.reader_1.hardware_version"
```

Example

This example shows the response you receive when an RFID reader is attached:

```
! U1 getvar "rfid.reader_1.hardware_version"
```

If an RFID reader is present and connected, you get the hardware version in the following format:

`"xx.xx.xx.xx"`

If there is no RFID reader or if the reader is not connected correctly, the response is blank.

rfid.reader_1.model

This command returns the printer's RFID reader model number.

Getvar

To return the printer's RFID reader model number:

```
! U1 getvar "rfid.reader_1.model"
```

rfid.recipe_version

The RFID recipe file controls how the printer manages RFID tag encoding, according to the type of tag in use.

This command returns the version number of the RFID recipe file currently in use. The RFID recipe file is named RFIDRCPE.XML. The default location for this file is `Z:RFIDRCPE.XML`. If a file using the same name is stored in the E: memory location, it will be used instead of the file stored in the Z: memory location.

Getvar

To return the version number of the RFID recipe file currently in use:

```
! U1 getvar "rfid.recipe_version"
```

rfid.region_code

This command returns the region code assigned to the printer's RFID device.

Getvar

To retrieve the RFID region code:

```
! U1 getvar "rfid.region_code"
```

Values

- `"not available"`
- `"usa/canada"`
- `"japan"`
- `"rest of world"`

rfid.tag.calibrate

Use this command to initiate tag calibration for RFID media. During the process, the printer moves the media, calibrates the RFID tag position, and determines the optimal settings for the RFID media being used. Depending on the printer, these settings include the programming position, the antenna element to use, and the read/write power level to use.

NOTE: For more information about RFID tag calibration, refer to the RFID Programming Guide for your printer. A copy is available online at www.zebra.com/manuals.

Setvar

To initiate tag calibration for RFID media:

```
! U1 setvar "rfid.tag.calibrate" "value"
```

Values

`"restore"`

`"run"`

Example

This `setvar` example restores the programming position back to the printer's default value.

```
! U1 setvar "rfid.tag.calibrate" "restore"
```

This `setvar` example performs RFID tag calibration.

```
! U1 setvar "rfid.tag.calibrate" "run"
```

rfid.tag.data

This command tells the RFID reader to attempt to read a tag over the RFID antenna, even if the printhead is open. Results are returned to the host.

Before running this command, position an RFID label over the printer's RFID antenna.

Getvar

To return the current tag's data:

```
! U1 getvar "rfid.tag.data"
```

For more information about this option and for the location of the RFID antenna, refer to the RFID Programming Guide for your printer. A copy is available online at www.zebra.com/manuals.

Examples

This example gets data from the current tag, assuming that an RFID label with data `"0123456789ABCDEF12345678"` is in place over the antenna.

```
! U1 setvar "rfid.tag.data"
```

The printer responds with `"0123456789ABCDEF12345678"`.

This example gets data from the current tag, assuming that no tag data can be read or that no tag is present.

```
! U1 setvar "rfid.tag.data"
```

The printer responds with "`NO DATA`".

rfid.tag.read.content

This command instructs the printer which data to read from the tag with the `rfid.tag.read.execute` command.

See rfid.tag.read.execute on page 1527.

Setvar

To instruct the printer which data to read from the tag with the `rfid.tag.read.execute` command:

```
! U1 setvar "rfid.tag.read.content" "value"
```

Values

- `"epc"` reads the EPC data based on the EPC size specified in the RFID tag's protocol bits, up to 160 bits
- `"tid information"` reads the first 32 bits of the TID (Tag ID)
- `"password status"` reads the tag's access and kill passwords
- `"protocol bits"` reads the protocol bits from the EPC memory banks and converts that value to the EPC size
- `"memory bank sizes"` reads the EPC, TID, and user memory banks sizes
- `"up"` sets the command to the previous test
- `"down"` sets the command to the next test

Default

`"epc"`

Getvar

To retrieve the current setting:

```
! U1 getvar "rfid.tag.read.content"
```

rfid.tag.read.execute

This command reads the data specified by the `rfid.tag.read.content` command.

See rfid.tag.read.content on page 1526.

Setvar

To read the specified data:

```
! U1 setvar "rfid.tag.read.execute"
```

rfid.tag.read.result_line1

This command reports the results of the `rfid.tag.read.execute` command.

Getvar

To retrieve the results of the `rfid.tag.read.execute` command:

```
! U1 getvar "rfid.tag.read.result_line1"
```

rfid.tag.read.result_line2

This command reports the results of the `rfid.tag.read.execute` command.

Getvar

To retrieve the results of the `rfid.tag.read.execute` command:

```
! U1 setvar "rfid.tag.read.result_line2"
```

rfid.tag.read.result_line1_alternate

This command reports the results of the `rfid.tag.read.execute` command.

Getvar

To retrieve the results of the `rfid.tag.read.execute` command:

```
! U1 setvar "rfid.tag.read.result_line1_alternate"
```

rfid.tag.read.result_line2_alternate

This command reports the results of the `rfid.tag.read.execute` command.

Getvar

To retrieve the results of the `rfid.tag.read.execute` command:

```
! U1 getvar "rfid.tag.read.result_line2_alternate"
```

rfid.tag.test

This command performs an RFID test. In the RFID test, the printer attempts to read and write to a transponder that you place over the RFID antenna. Results are displayed on the printer's control panel display.

For more information about the RFID antenna location, refer to the RFID Programming Guide for your printer. A copy is available online at www.zebra.com/manuals.

In the slow version of the RFID test, the printer first displays the hardware version, the reader firmware version, and the program position.

NOTE: This command is valid only on RP4T printers.

Setvar

To set the programming position:

```
! U1 setvar "rfid.tag.test" "value"
```

Values

"quick"

"slow"

Example

This `setvar` example performs a quick RFID test, which shows a pass or fail message.

```
! U1 setvar "rfid.tag.test" "quick"
```

This `setvar` example performs a slow RFID test, which shows the success or failure of each read or write tag operation.

```
! U1 setvar "rfid.tag.test" "slow"
```

rfid.tag.test.content

This command instructs the printer which test to perform on the tag with the `rfid.tag.test.execute` command.

See rfid.tag.test.execute on page 1534.

Setvar

To instruct the printer which test to perform on the tag with the `rfid.tag.test.execute` command:

```
! U1 setvar "rfid.tag.test.content" "value"
```

Values

- `"quick"` performs a read EPC test and a write EPC test (using random data)
- `"read"` performs a read EPC test
- `"write"` performs a write EPC test (using random data)
- `"up"` sets the command to the previous test
- `"down"` sets the command to the next test

Default

`"quick"`

Getvar

To retrieve the current setting:

```
! U1 getvar "rfid.tag.test.content"
```

rfid.tag.test.execute

This command tests the data specified by the `rfid.tag.test.content` command.

See rfid.tag.test.content on page 1533.

Setvar

To test the specified data:

```
! U1 setvar "rfid.tag.test.execute"
```

rfid.tag.test.result_line1

This command reports the results of the `rfid.tag.test.execute` command.

Getvar

To retrieve the results of the `rfid.tag.test.execute` command:

```
! U1 setvar "rfid.tag.test.result_line1"
```

rfid.tag.test.result_line2

This command reports the results of the `rfid.tag.test.execute` command.

Getvar

To retrieve the results of the `rfid.tag.test.execute` command:

```
! U1 setvar "rfid.tag.test.result_line2"
```

rfid.tag.type

This command sets the reader's RFID tag type.

Setvar

To set the reader's tag type:

```
! U1 setvar "rfid.tag.type" "value"
```

Values

UHF Printers

Value	Definition
none	None
class0	EPC Class 0
class0+	EPC Class 0 Plus
class1_64bit	EPC Class 1 64-bit
class1_96bit	EPC Class 1 96-bit
ucode_epc_1_19	UCODE EPC 1.19
class0+_impinj	mpinj Class 0 Plus
ISO18000A	ISO 18000-06A
gen2	EPC Class 1, Generation 2 (Gen 2)
ISO18000B	ISO18000B

HF Printers

Value	Definition
none	None
detect	Auto detect (query tag to determine)
tagit	Tag*It (Texas Instruments Tagit tags)
icode	I*code (Phillips Icode tags)
pico	Pico Tag (Inside Technology's)
ISO15693	ISO 15693
EPC	EPC tag (13.56 MHz)
UIC	UID Tag
mifare_ultralight	Mifare UltraLight

Getvar

To respond with the reader's current tag type:

```
! U1 getvar "rfid.tag.type"
```

Example

This `setvar` example shows the reader's tag type being set to Gen 2.

```
! U1 setvar "rfid.tag.type" "gen2"
```

For tag types supported by older printers, refer to the original RFID Programming Guide, part number 58978L-xxx).

rfid.log.enabled

This command enables or disables the RFID host log.

Setvar

To set the command:

```
! U1 setvar "rfid.log.enabled" "value"
```

Values

`"yes"` Enables the RFID host log

`"no"` Disables the RFID host log

Default

`"no"`

Example

In this example, the setvar enables the RFID host log.

```
! U1 setvar "rfid.log.enabled" "yes"
```

Getvar

To view the current setting value:

```
! U1 getvar "rfid.log.enabled"
```

rfid.log.entries

This command returns the RFID host log. This command is equivalent to the ^HL and ~HL command. Host logs are not displayed during an ALLCV.

Getvar

To get the RFID host logs:

```
! U1 getvar "rfid.log.entries"
```

Result

```
[0x02]<start>
Nov-13-2017 23:31:30,R,F0,A1,16,00000000,E2009059621800752090 38CD
...
<end>[0x03]
```

In this example, "..." can be more entries.

`[0x02]` and `[0x03]` are the STX and ETX binary characters.

rfid.log.clear

This command clears the RFID host log.

Setvar

To set the command:

```
! U1 setvar "rfid.log.clear" ""
```

Values
NA

Do

To clear the RFID host logs:

```
! U1 do "rfid.log.clear" ""
```

Values
NA

Zebra Code Pages

This section lists the various character code pages.

Zebra Code Page 850 — Latin Character Set

This is the Zebra Code Page 850:

NOTE: For hex 5C, a cent sign prints for all printer resident fonts. A backslash prints for downloaded fonts.

CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC
	20	32	0	30	48	@	40	64	P	50	80	‘	60	96	p	70	112	Ç	80	128
!	21	33	1	31	49	A	41	65	Q	51	81	a	61	97	q	71	113	ü	81	129
"	22	34	2	32	50	B	42	66	R	52	82	b	62	98	r	72	114	é	82	130
#	23	35	3	33	51	C	43	67	S	53	83	c	63	99	s	73	115	â	83	131
$	24	36	4	34	52	D	44	68	T	54	84	d	64	100	t	74	116	ä	84	132
%	25	37	5	35	53	E	45	69	U	55	85	e	65	101	u	75	117	à	85	133
&	26	38	6	36	54	F	46	70	V	56	86	f	66	102	v	76	118	å	86	134
’	27	39	7	37	55	G	47	71	W	57	87	g	67	103	w	77	119	ç	87	135
(	28	40	8	38	56	H	48	72	X	58	88	h	68	104	x	78	120	ê	88	136
)	29	41	9	39	57	I	49	73	Y	59	89	i	69	105	y	79	121	ë	89	137
*	2a	42	:	3a	58	J	4a	74	Z	5a	90	j	6a	106	z	7a	122	è	8a	138
+	2b	43	;	3b	59	K	4b	75	[	5b	91	k	6b	107	{	7b	123	ï	8b	139
,	2c	44	<	3c	60	L	4c	76	¢ \	5c	92	l	6c	108	|	7c	124	î	8c	140
–	2d	45	=	3d	61	M	4d	77	]	5d	93	m	6d	109	}	7d	125	ì	8d	141
.	2e	46	>	3e	62	N	4e	78	^	5e	94	n	6e	110	~	7e	126	Ä	8e	142
/	2f	47	?	3f	63	O	4f	79	_	5f	95	o	6f	111	⌂	7f	127	Å	8f	143

CHR	HEX	DEC
É	90	144
æ	91	145
Æ	92	146
ô	93	147
ö	94	148
ò	95	149
û	96	150
ù	97	151
ÿ	98	152
Ö	99	153
Ü	9a	154
ø	9b	155
£	9c	156
Ø	9d	157
×	9e	158
ƒ	9f	159
á	a0	160
í	a1	161
ó	a2	162
ú	a3	163
ñ	a4	164
Ñ	a5	165
ª	a6	166
º	a7	167
¿	a8	168
®	a9	169
¬	aa	170
½	ab	171
¼	ac	172
¡	ad	173
«	ae	174
»	af	175
▒	b0	176
▒	b1	177
▓	b2	178
│	b3	179
┤	b4	180
Á	b5	181
Â	b6	182
À	b7	183
©	b8	184
╣	b9	185
║	ba	186
╗	bb	187
╝	bc	188
¢	bd	189
¥	be	190
┐	bf	191
└	c0	192
┴	c1	193
┬	c2	194
├	c3	195
─	c4	196
┼	c5	197
ã	c6	198
Ã	c7	199
╚	c8	200
╔	c9	201
╩	ca	202
╦	cb	203
╠	cc	204
═	cd	205
╬	ce	206
¤	cf	207
ð	d0	208
Ð	d1	209
Ê	d2	210
Ë	d3	211
È	d4	212
ı	d5	213
Í	d6	214
Î	d7	215
Ï	d8	216
┘	d9	217
┌	da	218
█	db	219
▄	dc	220
¦	dd	221
Ì	de	222
▀	df	223
Ó	e0	224
ß	e1	225
Ô	e2	226
Ò	e3	227
õ	e4	228
Õ	e5	229
µ	e6	230
þ	e7	231
Þ	e8	232
Ú	e9	233
Û	ea	234
Ù	eb	235
ý	ec	236
Ý	ed	237
¯	ee	238
´	ef	239
	f0	240
±	f1	241
‗	f2	242
¾	f3	243
¶	f4	244
§	f5	245
÷	f6	246
¸	f7	247
°	f8	248
¨	f9	249
·	fa	250
¹	fb	251
³	fc	252
²	fd	253
■	fe	254
	ff	255

Zebra Code Page 1250 — Central and Eastern European Latin Character Set

This is the Zebra Code Page 1250 that supports scalable/downloaded TTF fonts:

NOTE: Font 0 (zero) was used to display this chart.

CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC
	20	32	0	30	48	@	40	64	P	50	80	‘	60	96	p	70	112	€	80	128
!	21	33	1	31	49	A	41	65	Q	51	81	a	61	97	q	71	113		81	129
”	22	34	2	32	50	B	42	66	R	52	82	b	62	98	r	72	114	‚	82	130
#	23	35	3	33	51	C	43	67	S	53	83	c	63	99	s	73	115		83	131
$	24	36	4	34	52	D	44	68	T	54	84	d	64	100	t	74	116	„	84	132
%	25	37	5	35	53	E	45	69	U	55	85	e	65	101	u	75	117	…	85	133
&	26	38	6	36	54	F	46	70	V	56	86	f	66	102	v	76	118	†	86	134
’	27	39	7	37	55	G	47	71	W	57	87	g	67	103	w	77	119	‡	87	135
(	28	40	8	38	56	H	48	72	X	58	88	h	68	104	x	78	120		88	136
)	29	41	9	39	57	I	49	73	Y	59	89	i	69	105	y	79	121	‰	89	137
*	2a	42	:	3a	58	J	4a	74	Z	5a	90	j	6a	106	z	7a	122	Š	8a	138
+	2b	43	;	3b	59	K	4b	75	[	5b	91	k	6b	107	{	7b	123	‹	8b	139
,	2c	44	<	3c	60	L	4c	76	\	5c	92	l	6c	108	|	7c	124	Ś	8c	140
–	2d	45	=	3d	61	M	4d	77	]	5d	93	m	6d	109	}	7d	125	Ť	8d	141
.	2e	46	>	3e	62	N	4e	78	^	5e	94	n	6e	110	~	7e	126	Ž	8e	142
/	2f	47	?	3f	63	O	4f	79	_	5f	95	o	6f	111		7f	127	Ź	8f	143

CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC
	90	144		a0	160	°	b0	176	Ŕ	c0	192	Đ	d0	208	ŕ	e0	224	đ	f0	240
‘	91	145	˘	a1	161	±	b1	177	Á	c1	193	Ń	d1	209	á	e1	225	ń	f1	241
’	92	146	ˇ	a2	162	˛	b2	178	Â	c2	194	Ň	d2	210	â	e2	226	ň	f2	242
“	93	147	Ł	a3	163	ł	b3	179	Ă	c3	195	Ó	d3	211	ă	e3	227	ó	f3	243
”	94	148	¤	a4	164	´	b4	180	Ä	c4	196	Ô	d4	212	ä	e4	228	ô	f4	244
•	95	149	Ą	a5	165	µ	b5	181	Ĺ	c5	197	Ő	d5	213	Í	e5	229	ő	f5	245
–	96	150	¦	a6	166	¶	b6	182	Ć	c6	198	Ö	d6	214	ć	e6	230	ö	f6	246
—	97	151	§	a7	167	·	b7	183	Ç	c7	199	×	d7	215	ç	e7	231	÷	f7	247
	98	152	¨	a8	168	¸	b8	184	Č	c8	200	Ř	d8	216	č	e8	232	ř	f8	248
™	99	153	©	a9	169	ą	b9	185	É	c9	201	Ů	d9	217	é	e9	233	ů	f9	249
š	9a	154	Ş	aa	170	ş	ba	186	Ę	ca	202	Ú	da	218	ę	ea	234	ú	fa	250
›	9b	155	«	ab	171	»	bb	187	Ë	cb	203	Ű	db	219	ë	eb	235	ű	fb	251
ś	9c	156	¬	ac	172	Ľ	bc	188	Ě	cc	204	Ü	dc	220	ě	ec	236	ü	fc	252
ť	9d	157	-	ad	173	˝	bd	189	Í	cd	205	Ý	dd	221	í	ed	237	ý	fd	253
ž	9e	158	®	ae	174	ľ	be	190	Î	ce	206	Ţ	de	222	î	ee	238	ţ	fe	254
ź	9f	159	Ż	af	175	ż	bf	191	Ď	cf	207	ß	df	223	ď	ef	239	˙	ff	255

Zebra Code Page 1252— Latin Character Set

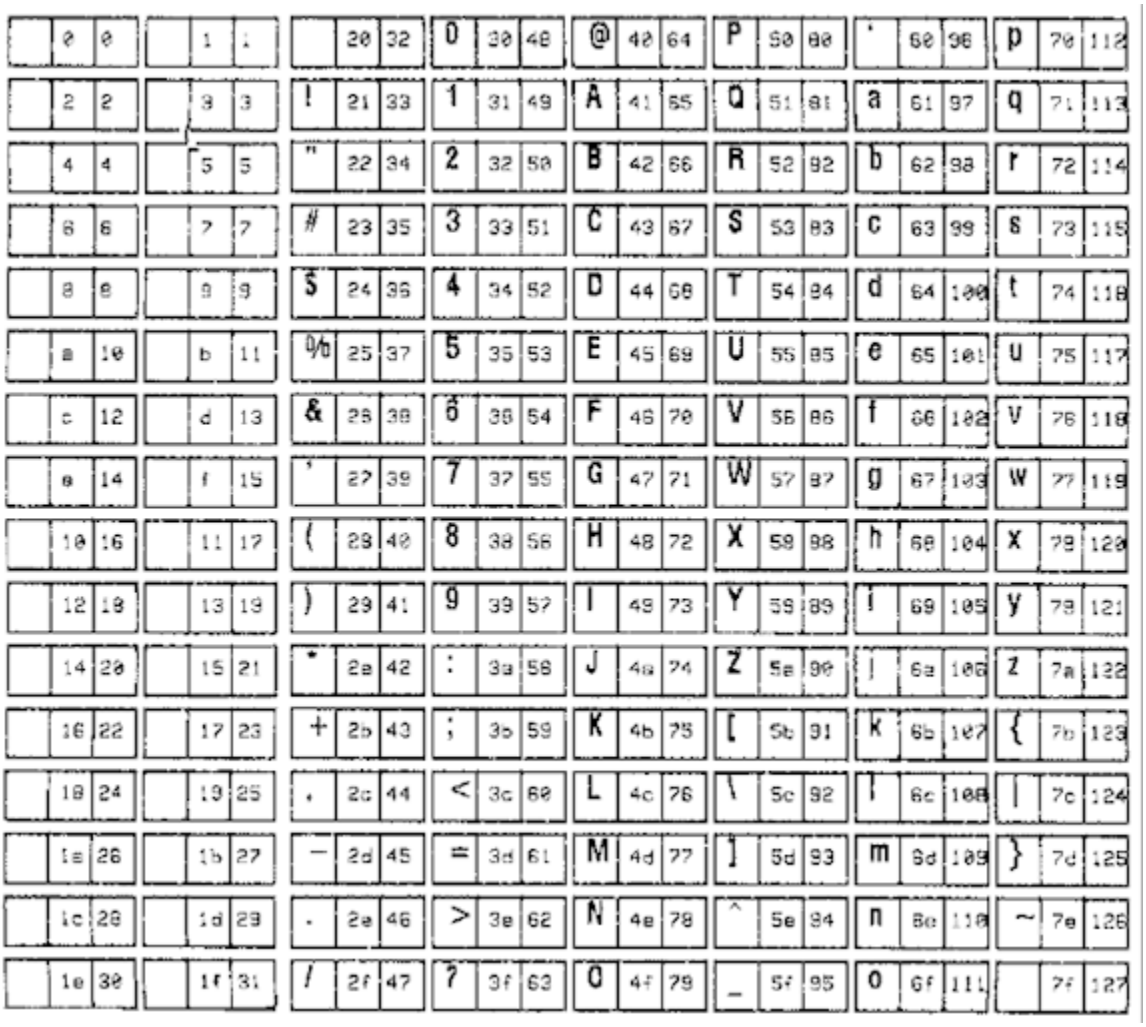
This is the Zebra Code Page 1252:

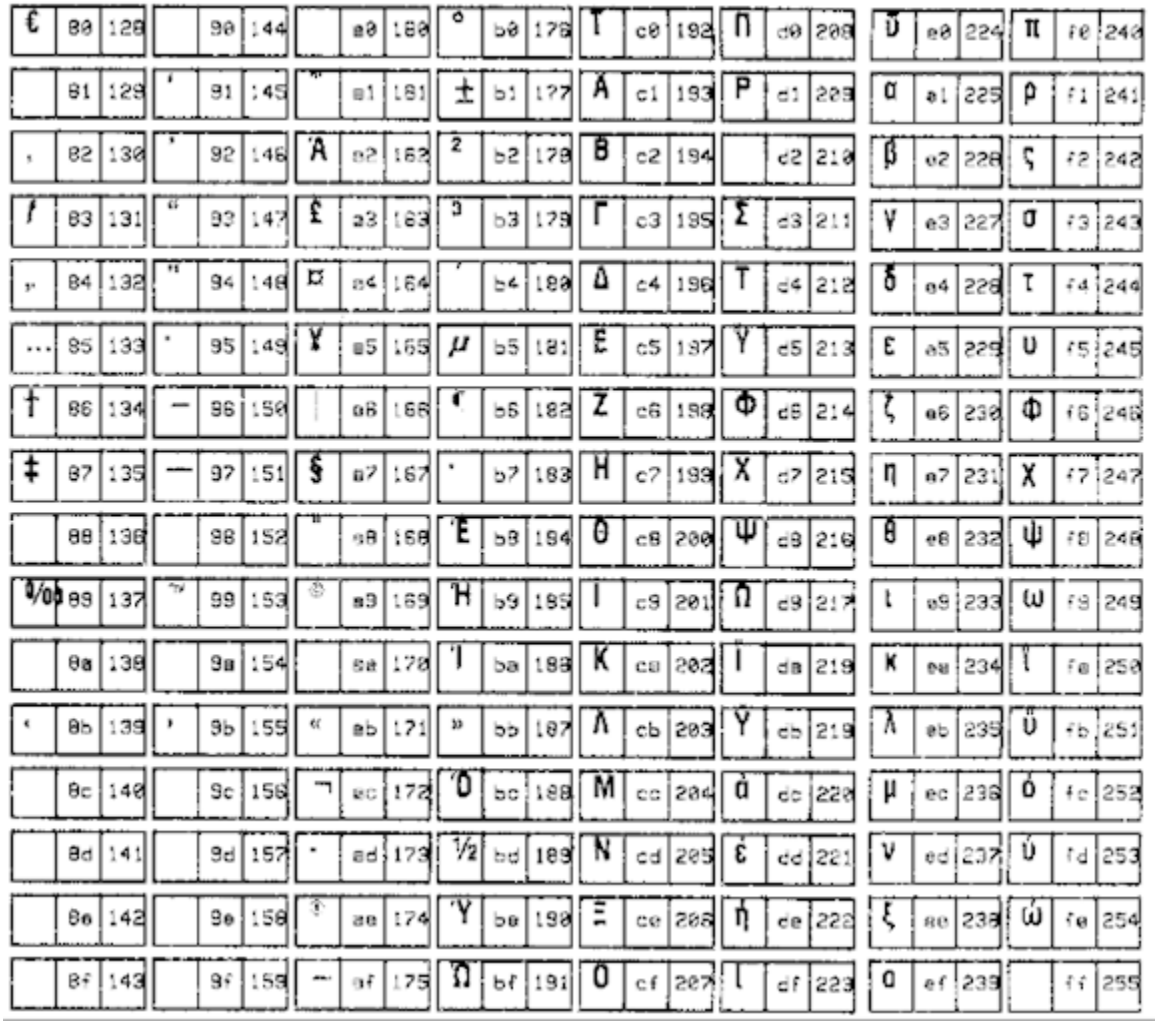
CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC
	20	32	0	30	48	@	40	64	P	50	80	`	60	96	p	70	112	€	80	128
!	21	33	1	31	49	A	41	65	Q	51	81	a	61	97	q	71	113		81	129
"	22	34	2	32	50	B	42	66	R	52	82	b	62	98	r	72	114	‚	82	130
#	23	35	3	33	51	C	43	67	S	53	83	c	63	99	s	73	115	*ƒ*	83	131
$	24	36	4	34	52	D	44	68	T	54	84	d	64	100	t	74	116	„	84	132
%	25	37	5	35	53	E	45	69	U	55	85	e	65	101	u	75	117	…	85	133
&	26	38	6	36	54	F	46	70	V	56	86	f	66	102	v	76	118	†	86	134
'	27	39	7	37	55	G	47	71	W	57	87	g	67	103	w	77	119	‡	87	135
(	28	40	8	38	56	H	48	72	X	58	88	h	68	104	x	78	120	ˆ	88	136
)	29	41	9	39	57	I	49	73	Y	59	89	i	69	105	y	79	121	‰	89	137
*	2a	42	:	3a	58	J	4a	74	Z	5a	90	j	6a	106	z	7a	122	Š	8a	138
+	2b	43	;	3b	59	K	4b	75	[	5b	91	k	6b	107	{	7b	123	‹	8b	139
,	2c	44	<	3c	60	L	4c	76	\	5c	92	l	6c	108	|	7c	124	Œ	8c	140
-	2d	45	=	3d	61	M	4d	77	]	5d	93	m	6d	109	}	7d	125		8d	141
.	2e	46	>	3e	62	N	4e	78	^	5e	94	n	6e	110	~	7e	126	Ž	8e	142
/	2f	47	?	3f	63	O	4f	79	_	5f	95	o	6f	111		7f	127		8f	143

CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC
	90	144		a0	160	°	b0	176	À	c0	192	Ð	d0	208	à	e0	224	ð	f0	240
‘	91	145	¡	a1	161	±	b1	177	Á	c1	193	Ñ	d1	209	á	e1	225	ñ	f1	241
’	92	146	¢	a2	162	2	b2	178	Â	c2	194	Ò	d2	210	â	e2	226	ò	f2	242
“	93	147	£	a3	163	3	b3	179	Ã	c3	195	Ó	d3	211	ã	e3	227	ó	f3	243
”	94	148	¤	a4	164	´	b4	180	Ä	c4	196	Ô	d4	212	ä	e4	228	ô	f4	244
•	95	149	¥	a5	165	µ	b5	181	Å	c5	197	Õ	d5	213	å	e5	229	õ	f5	245
–	96	150	¦	a6	166	¶	b6	182	Æ	c6	198	Ö	d6	214	æ	e6	230	ö	f6	246
—	97	151	§	a7	167	·	b7	183	Ç	c7	199	×	d7	215	ç	e7	231	÷	f7	247
˜	98	152	¨	a8	168	¸	b8	184	È	c8	200	Ø	d8	216	è	e8	232	ø	f8	248
™	99	153	©	a9	169	1	b9	185	É	c9	201	Ù	d9	217	é	e9	233	ù	f9	249
š	9a	154	ª	aa	170	º	ba	186	Ê	ca	202	Ú	da	218	ê	ea	234	ú	fa	250
›	9b	155	«	ab	171	»	bb	187	Ë	cb	203	Û	db	219	ë	eb	235	û	fb	251
œ	9c	156	¬	ac	172	¼	bc	188	Ì	cc	204	Ü	dc	220	ì	ec	236	ü	fc	252
	9d	157	-	ad	173	½	bd	189	Í	cd	205	Ý	dd	221	í	ed	237	ý	fd	253
ž	9e	158	®	ae	174	¾	be	190	Î	ce	206	Þ	de	222	î	ee	238	þ	fe	254
Ÿ	9f	159	¯	af	175	¿	bf	191	Ï	cf	207	ß	df	223	ï	ef	239	ÿ	ff	255

Zebra Code Page 1253 — Modern Greek Character Set

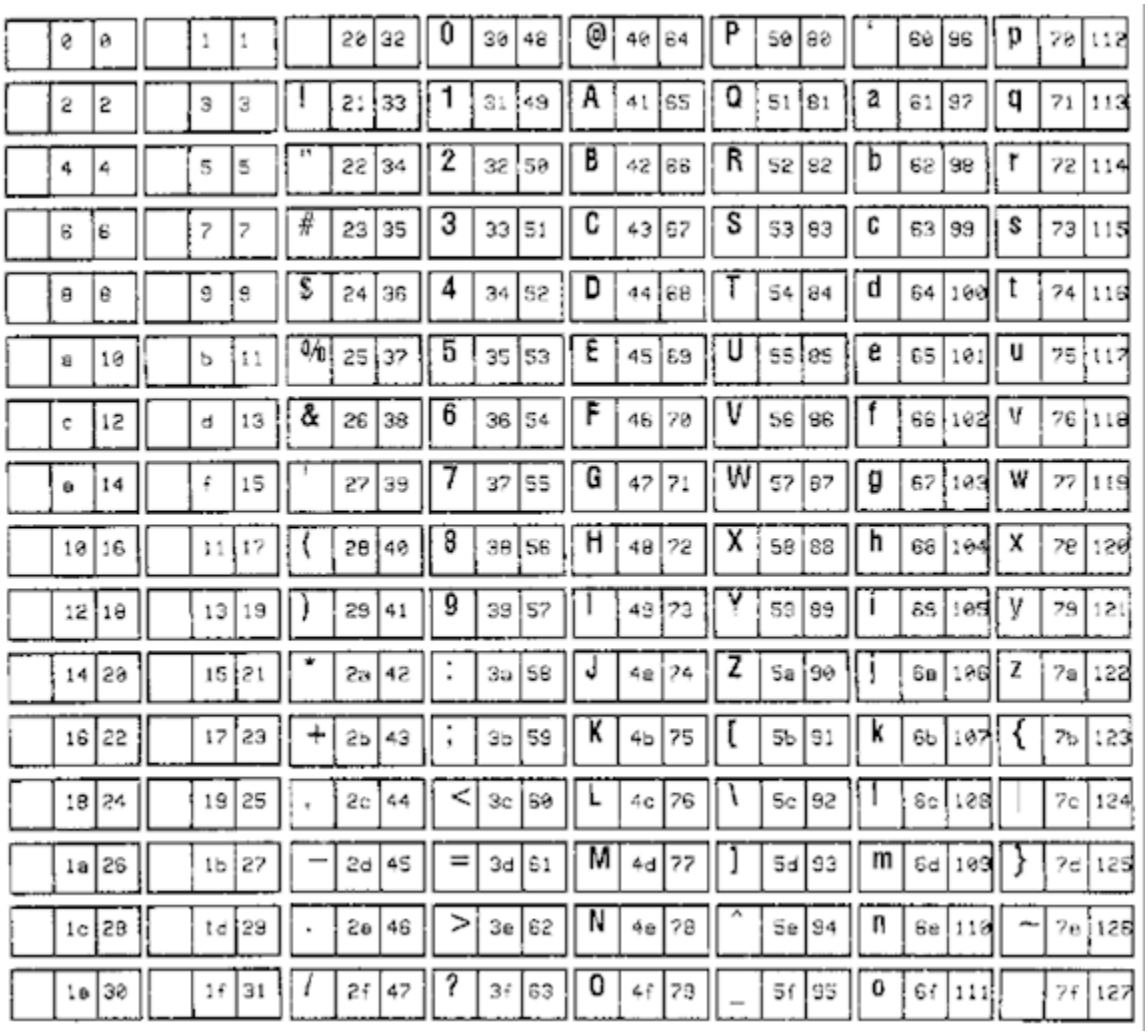
This is the Zebra Code Page 1253:

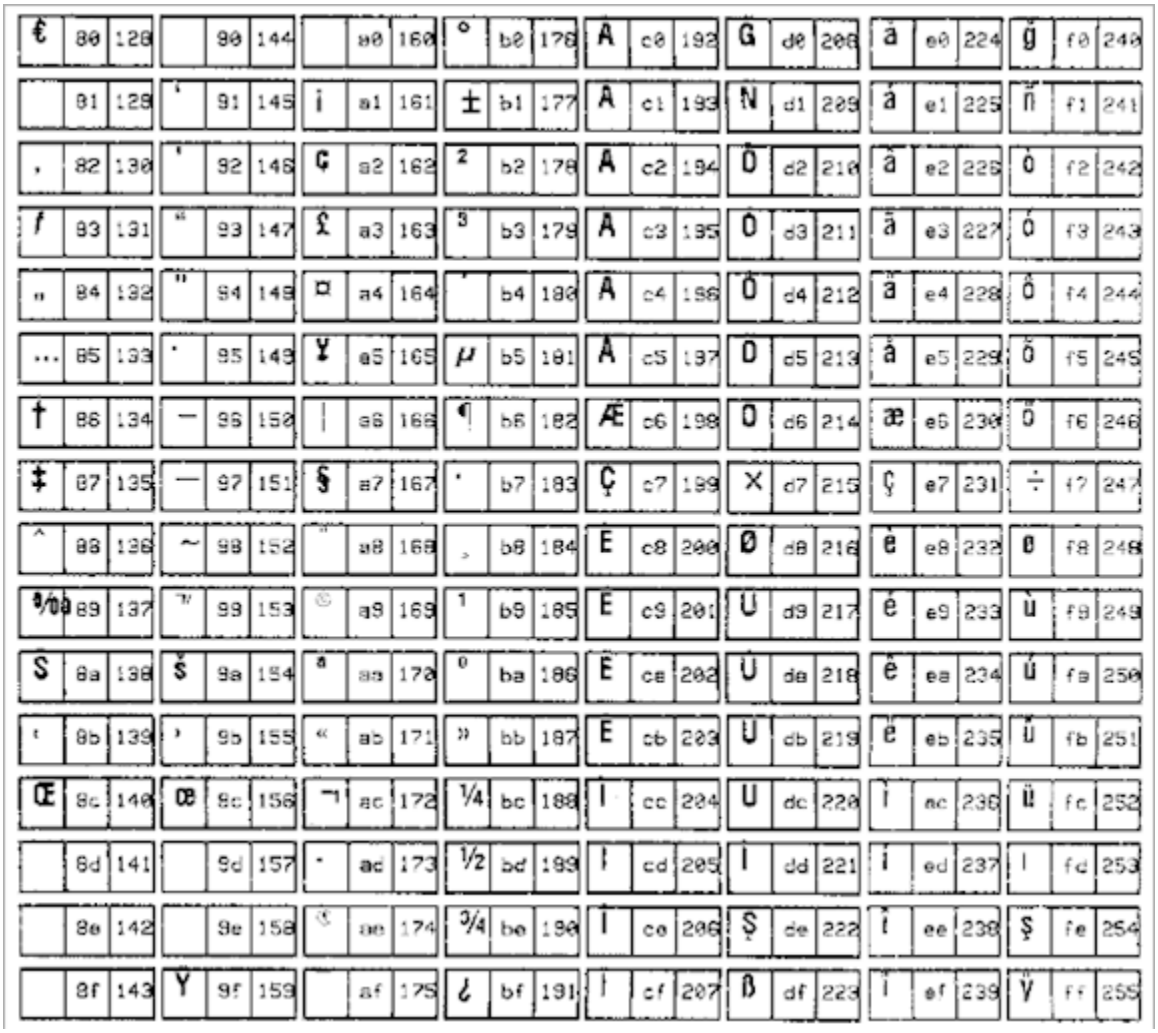




Zebra Code Page 1254 — Turkish Character Set

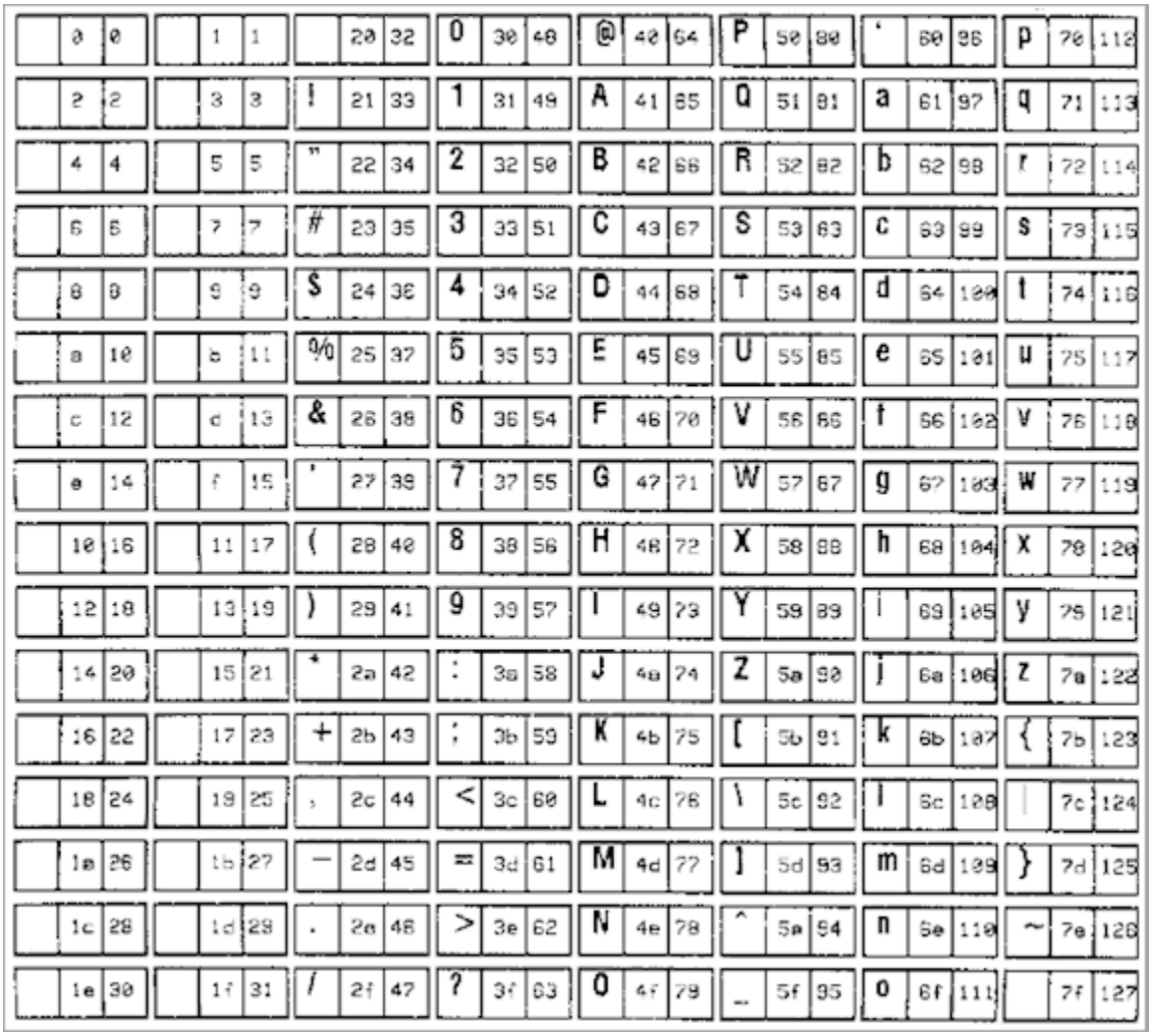
This is the Zebra Code Page 1254.

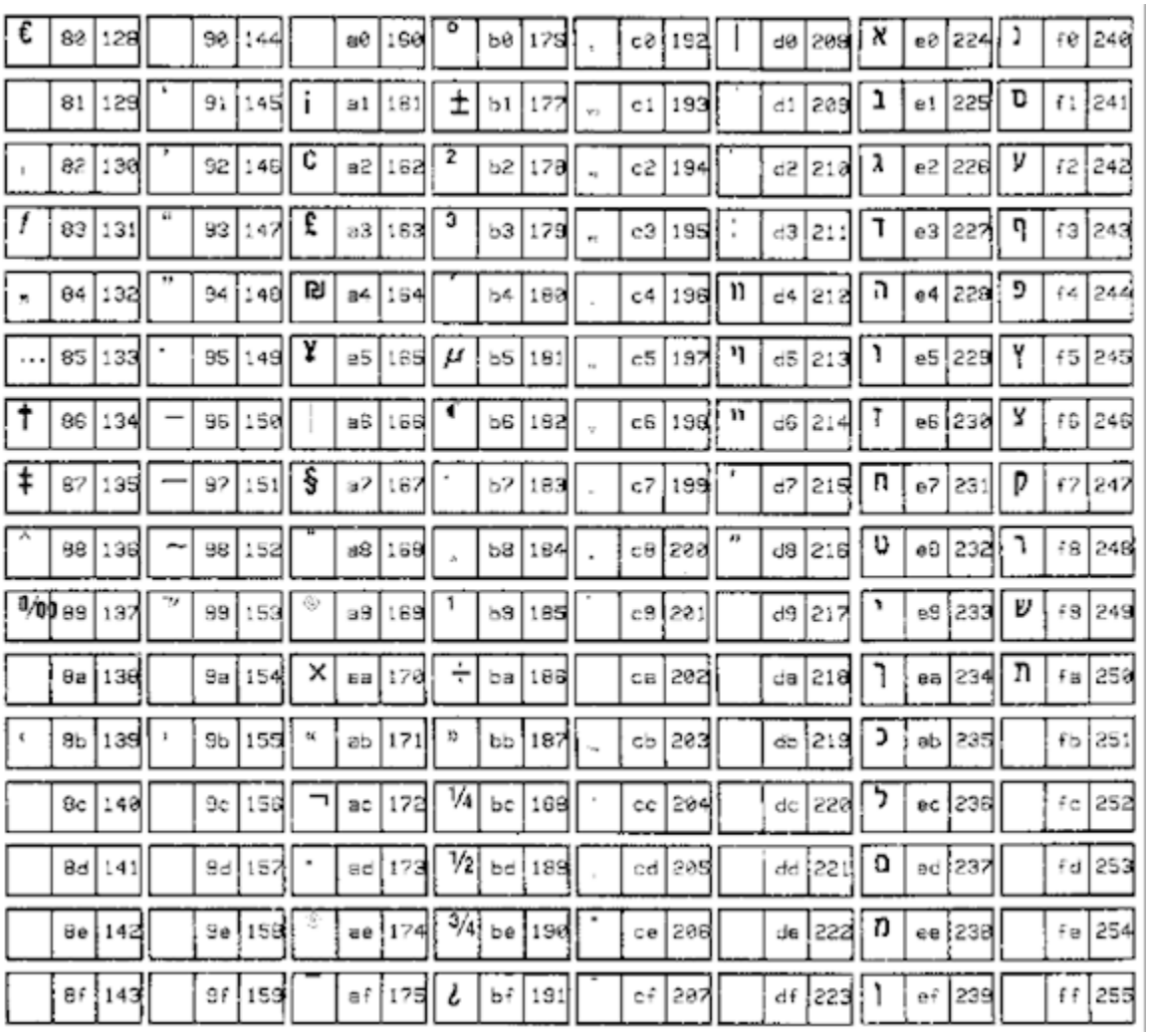




Zebra Code Page 1255 — Hebrew Character Set

This is the Zebra Code Page 1255:





ASCII

This section shows the American Standard Code for Information Interchange (ASCII) code used by Zebra printers.

Shaded areas in the table below indicate characters not recommended for command prefix, format prefix, or delimiter characters.

Table 28 ASCII Code Chart

HEX	Character	HEX	Character	HEX	Character	HEX	Character
00	NUL	20	Space	40	@	60	‘
01	SOH	21	!	41	A	61	a
02	STX	22	“	42	B	62	b
03	ETX	23	#	43	C	63	c
04	EOT	24	$	44	D	64	d
05	ENQ	25	%	45	E	65	e
06	ACK	26	&	46	F	66	f
07	BEL	27	‘	47	G	67	g
08	BS	28	(	48	H	68	h
09	HT	29	)	49	I	69	i
0A	LF	2A	*	4A	J	6A	j
0B	VT	2B	+	4B	K	6B	k
0C	FF	2C	,	4C	L	6C	l
0D	CR	2D	-	4D	M	6D	m
0E	SO	2E	.	4E	N	6E	n
0F	SI	2F	/	4F	O	6F	o
10	DLE	30	0	50	P	70	p
11	DC1	31	1	51	Q	71	q
12	DC2	32	2	52	R	72	r
13	DC3	33	3	53	S	73	s
14	DC4	34	4	54	T	74	t

Table 28 ASCII Code Chart (Continued)

HEX	Character	HEX	Character	HEX	Character	HEX	Character
15	NAK	35	5	55	U	75	u
16	SYN	36	6	56	V	76	v
17	ETB	37	7	57	W	77	w
18	CAN	38	8	58	X	78	x
19	EM	39	9	59	Y	79	y
1A	SUB	3A	:	5A	Z	7A	z
1B	ESC	3B	;	5B	[	7B	{
1C	FS	3C	<	5C	\	7C	|
1D	GS	3D	=	5D	]	7D	}
1E	RS	3E	>	5E	^	7E	~
1F	US	3F	?	5F	_	7F	DEL

Fonts and Barcodes

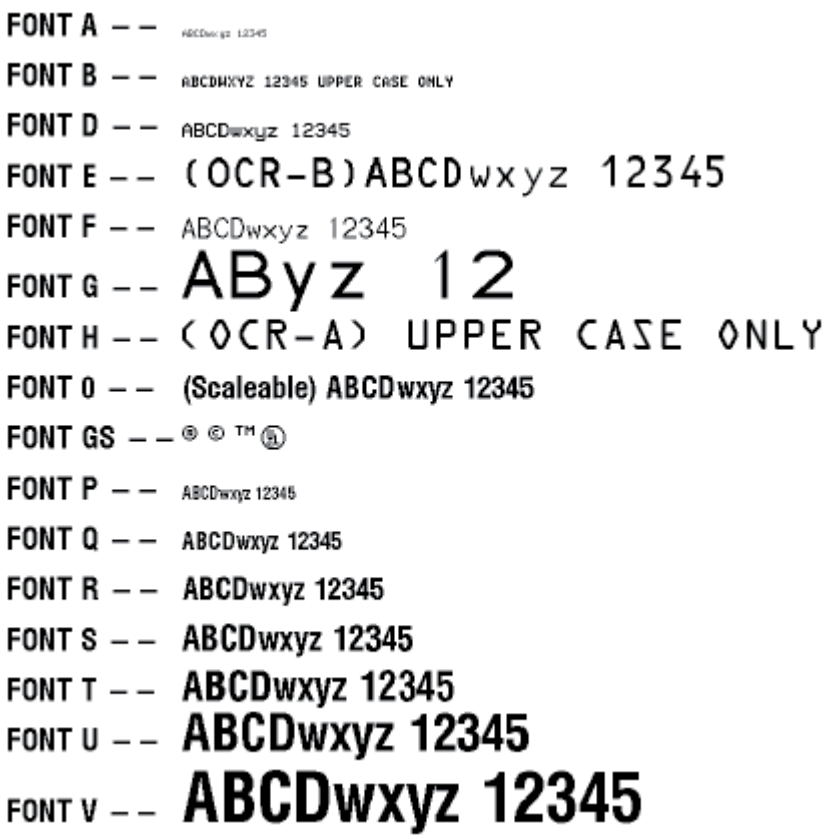
This section provides information about different fonts (type faces) and barcodes that can be used with the printer.

Standard Printer Fonts

Most Zebra printers come standard with 15 bitmapped fonts and one scalable font.

Additional downloadable bitmapped and scalable fonts are also available. Character size and density (how dark it appears) depend on the density of the printhead and the media used.

Figure 22 Examples of the Standard Printer Fonts

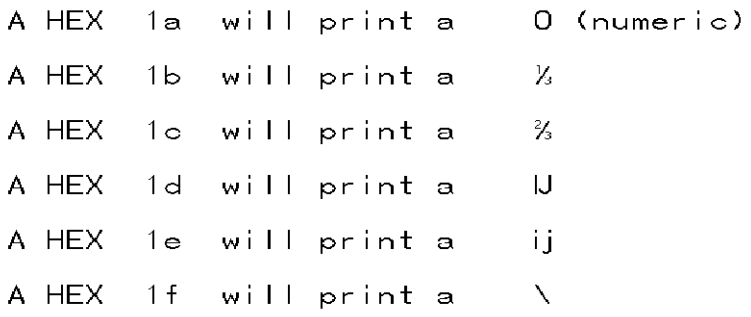


To use one of these fonts, you must either use the change alphanumeric default font command (^CF) or specify an alphanumeric field command (^A).

The standard Zebra character set is Code 850 for character values greater than 20 HEX. There are six HEX character values below 20 HEX that are also recognized. The figure below shows how these character values are printed.

NOTE: Unidentified characters should default to a space.

Figure 23 Recognized HEX Values below 20 HEX

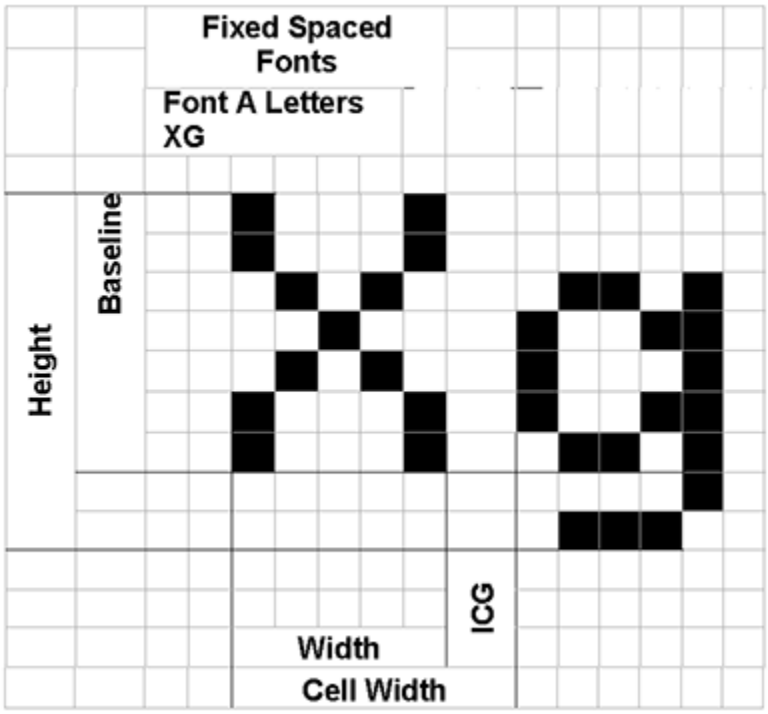


Proportional and Fixed Spacing

Proportional spacing is different than fixed spacing. In Table 29 Intercharacter Gap and Baseline Parameters on page 1559, the intercharacter gap (ICG), the space between characters, is constant for

fonts A through H, which means that the spacing between all characters is the same. For example, the spacing between the letters `MW` is the same as between the letters `IE`.

Figure 24 Fixed Space Fonts Proportion Example



The baseline is the imaginary line on which the bottom (base) of all characters (except any descenders) rest. The area between the baseline and the bottom of the matrix is used for any character “descenders.” Baseline numbers define where the baseline is located in relationship to the top of the matrix. For example, the baseline for font “E” is 23 dots down from the top of the matrix.

Table 29 Intercharacter Gap and Baseline Parameters

Font	H x W (in dots)	Type	Intercharacter Gap (in dots)	Baseline(in dots)
A	9 x 5	U-L-D	1	7
B	11 x 7	U	2	11
C,D	18 x 10	U-L-D	2	14
E	28 x 15	OCR-B	5	23
F	26 x 13	U-L-D	3	21
G	60 x 40	U-L-D	8	48
H	21 x 13	OCR-A	6	21
GS	24 x 24	SYMBOL	PROPORTIONAL	3 x HEIGHT/4
0	DEFAULT: 15 x 12		PROPORTIONAL	3 x HEIGHT/4

Scalable Versus Bitmapped Fonts

For scalable fonts, setting the height and width equally produces characters that appear the most balanced. Balanced characters are pleasing to the eye because actual height and width are approximately equal to each other. This is achieved through the use of a smooth-scaling algorithm in the printer.

For bitmapped fonts, this balancing is built into the font. In actuality, the height of a bitmap font is slightly larger than the width. Bitmap fonts are always at the maximum size of the character's cell.

Font Matrices

This section lists the font matrices.

Type Key

U = Uppercase, L = Lowercase, D = Descenders

Table 30 6 dot/mm Printhead

Font	Matrix	Type	Character Size			
	HxW (in dots)		HxW (in in.)	Char./in.	HxW (in mm)	Char. /mm
A	9 x 5	U-L-D	0.059 x 0.039	25.4	1.50 x 0.99	1.01
B	11 x 7	U	0.072 x 0.059	16.9	1.82 x 1.50	0.066
C, D	18 x 10	U-L-D	0.118 x 0.079	12.7	2.99 x 2.00	0.05
E	21 x 10	OCR-B	0.138 x 0.085	11.7	3.50 x 2.16	0.46
F	26 x 13	U-L-D	0.170 x 0.105	9.53	4.32 x 2.67	0.37
G	60 x 40	U-L-D	0.394 x 0.315	3.18	10.0 x 8.00	0.125
H	17 x 11	OCR-A	0.111 x 0.098	10.2	2.81 x 2.48	0.40
GS	24 x 24	SYMBOL	0.157 x 0.157	6.35	3.98 x 3.98	0.251
0	Default: 15 x 12					

Table 31 8 dot/mm (203 dpi) Printhead

Font	Matrix	Type	Character Size			
	HxW (in dots)		HxW (in in.)	Char./in.	HxW (in mm)	Char. /mm
A	9 X 5	U-L-D	0.044 x 0.030	33.3	1.12 x 0.76	1.31
B	11 X 7	U	0.054 x 0.044	22.7	1.37 x 1.12	0.89
C, D	18 X 10	U-L-D	0.089 x 0.059	16.9	2.26 x 1.12	0.66
E	28 x 15	OCR-B	0.138 x 0.098	10.2	3.50 x 2.49	0.40
F	26 x 13	U-L-D	0.128 x 0.079	12.7	3.25 x 2.00	0.50
G	60 x 40	U-L-D	0.295 x 0.197	4.2	7.49 x 5.00	0.167
H	21 x 13	OCR-A	0.103 x 0.093	10.8	2.61 x 2.36	0.423
GS	24 x 24	SYMBOL	0.118 x 0.118	8.5	2.99 x 2.99	0.334

Table 31 8 dot/mm (203 dpi) Printhead (Continued)

Font	Matrix	Type	Character Size			
	HxW (in dots)		HxW (in in.)	Char./in.	HxW (in mm)	Char. /mm
P	20 x 18	U-L-D	0.098 x 0.089	N/A	2.50 x 2.25	N/A
Q	28 x 24	U-L-D	0.138 x 0.118	N/A	3.50 x 3.00	N/A
R	35 x 31	U-L-D	0.172 x 0.153	N/A	4.38 x 3.88	N/A
S	40 x 35	U-L-D	0.197 x 0.172	N/A	5.00 x 4.38	N/A
T	48 x 42	U-L-D	0.236 x 0.207	N/A	6.00 x 5.25	N/A
U	59 x 53	U-L-D	0.290 x 0.261	N/A	7.38 x 6.63	N/A
V	80 x 71	U-L-D	0.394 x 0.349	N/A	10.00 x 8.88	N/A
0	Default: 15 x 12	U-L-D	Scalable		Scalable	

Table 32 12 dot/mm (300 dpi) Printhead

Font	Matrix	Type	Character Size			
	HxW (in dots)		HxW (in in.)	Char./in.	HxW (in mm)	Char. /mm
A	9 X 5	U-L-D	0.030 x 0.020	50.8	0.75 x 0.50	2.02
B	11 X 7	U	0.036 x 0.030	33.8	0.91 x 0.75	1.32
C, D	18 X 10	U-L-D	0.059 x 0.040	25.4	1.50 x 1.00	1.00
E	42 x 20	OCR-B	0.138 x 0.085	23.4	1.75 x 1.08	0.92
F	26 x 13	U-L-D	0.085 x 0.053	19.06	2.16 x 1.34	0.74
G	60 x 40	U-L-D	0.197 x 0.158	6.36	5.00 x 4.00	0.25
H	34 x 22	OCR-A	0.111 x 0.098	10.20	2.81 x 2.48	0.40
GS	24 x 24	SYMBOL	0.079 x 0.079	12.70	1.99 x 1.99	0.52
P	20 x 18	U-L-D	0.067 x 0.060	N/A	1.69 x 1.52	N/A
Q	28 x 24	U-L-D	0.093 x 0.080	N/A	2.37 x 2.03	N/A
R	35 x 31	U-L-D	0.117 x 0.103	N/A	2.96 x 2.62	N/A
S	40 x 35	U-L-D	0.133 x 0.177	N/A	3.39 x 2.96	N/A
T	48 x 42	U-L-D	0.160 x 0.140	N/A	4.06 x 3.56	N/A
U	59 x 53	U-L-D	0.197 x 0.177	N/A	5.00 x 4.49	N/A
V	80 x 71	U-L-D	0.267 x 0.237	N/A	6.77 x 6.01	N/A
0	Default: 15 x 12	U-L-D	Scalable		Scalable	

Table 33 24 dot/mm (600 dpi) Printhead

Font	Matrix	Type	Character Size			
	HxW (in dots)		HxW (in in.)	Char./in.	HxW (in mm)	Char. /mm
A	9 X 5	U-L-D	0.015 x 0.010	100.00	0.38 x 0.25	4.00
B	11 X 7	U	0.018 x 0.015	66.66	0.46 x 0.38	2.60
C, D	18 X 10	U-L-D	0.030 x 0.020	50.00	0.77 x 0.51	2.0
E	42 x 20	OCR-B	0.137 x 0.087	11.54	3.47 x 2.20	0.45
F	26 x 13	U-L-D	0.043 x 0.027	37.5	1.10 x 0.68	1.50
G	60 x 40	U-L-D	0.100 x 0.080	12.50	2.54 x 2.04	0.50
H	34 x 22	OCR-A	0.100 x 0.093	10.71	2.54 x 2.37	0.42
GS	24 x 24	SYMBOL	0.040 x 0.040	25.00	1.02 x 1.02	1.00
P	20 x 18	U-L-D	0.067 x 0.060	N/A	1.69 x 1.52	N/A
Q	28 x 24	U-L-D	0.093 x 0.080	N/A	2.37 x 2.03	N/A
R	35 x 31	U-L-D	0.117 x 0.103	N/A	2.96 x 2.62	N/A
S	40 x 35	U-L-D	0.133 x 0.117	N/A	3.39 x 2.96	N/A
T	48 x 42	U-L-D	0.160 x 0.140	N/A	4.06 x 3.56	N/A
U	59 x 53	U-L-D	0.197 x 0.177	N/A	5.00 x 4.49	N/A
V	80 x 71	U-L-D	0.267 x 0.237	N/A	6.77 x 6.01	N/A
0	Default: 15 x 12	U-L-D	Scalable		Scalable	

Barcodes

Every barcode contains data made up of a sequence of light spaces and dark bars that represent letters, numbers, or other graphic characters. -

The usable characters differ among the various kinds of bar codes. Each barcode section in the ZPL Commands provides a table of applicable characters. Start and stop characters and check digits are used by many, but not all, barcodes. These will be indicated in the specific barcode explanations.

Zebra printers can print the following kinds of barcodes:

Barcode modulus “X” dimensions
• Picket fence (non-rotated) orientation:
  • 203 dpi = 0.0049 in. mil to 0.049 in.
  • 300 dpi = 0.0033 in. mil to 0.033 in.
• Ladder (rotated) orientation:
  • 203 dpi = 0.0049 in. mil to 0.049 in.
  • 300 dpi = 0.0039 in. mil to 0.039 in.
	Linear barcodes
	• Codabar
	• Code 11
	• Code 39
	• Code 93
	• Code 128 with subsets A/B C and UCC Case Codes
	• ISBT-128
	• UPC-A
	• UPC-E
	• EAN-8
	• EAN-13
	• UPC and EAN 2 or 5 digit extensions
	• Planet Code
	• Plessey
	• Postnet
	• Standard 2 of 5
	• Industrial 2 of 5
	• Interleaved 2 of 5
	• LOGMARS
	• MSI
	• GS1 DataBar Omnidirectional

Two-dimensional barcodes • Aztec • Code 49 • Maxi Code • TLC39 • PDF-417 • QR Code • Codablock • DataMatrix • Micro-PDF417	
Barcode ratios • 2:1 • 7:3 • 5:2 • 3:1	

Basic Format for Bar Codes

The basic format for bar codes is quiet zone, start character, data, check digit, stop character, and quiet zone. Not all bar codes require each of these elements.

Every bar code requires a quiet zone. A quiet zone (sometimes called a “clear area”) is an area adjacent to the machine-readable symbols that ensure proper reading (decoding) of the symbols. No printing is permissible within this area. Preprinted characters, borders, and background color are acceptable if they are invisible to the reading device; these are used in some applications but restrict the type of reading device that can be used. The size of the quiet zone depends on the size of bar widths (usually 10 times the width of the narrow bar).

Figure 25 Quiet Zone in a Bar Code

Barcode Field Instructions

To create a barcode, a barcode field command must be contained in the label format.

Table 34 Barcode Field Commands on page 1564 shows the barcode field commands. The number in brackets denotes the print ratio. Each command produces a unique barcode.

IMPORTANT: (*) for Fixed Printing Ratio means that the ratio between the width of the bars in the code is a fixed standard and cannot be changed.

As another reference to the barcode field commands ratio, see Table 6 Module Width Ratios in Dots on page 148.

Table 34 Barcode Field Commands

ZPL Command	Command Description	Ratio
^B0	Aztec Barcode Parameters	[Fixed]
^B1	Code 11 (USD-8)	[2.0 - 3.0]
^B2	Interleaved 2 of 5	[2.0 - 3.0]
^B3	Code 39 (USD-3 and 3 of 9)	[2.0 - 3.0]
^B4	Code 49 (*)	[Fixed]
^B5	Planet Code Barcode	[Fixed]
^B7	PDF417 (*)	[Fixed]
^B8	EAN-8 (*)	[Fixed]
^B9	UPC-E	[Fixed]
^BA	Code 93 (USS-93)(*)	[Fixed]
^BB	CODABLOCK A, E, F (*)	[Fixed]
^BC	Code 128 (USD-6) (*)	[Fixed]
^BD	UPS MaxiCode (*)	[Fixed]
^BE	EAN-13	[Fixed]
^BF	Micro-PDF417	[Fixed]

Table 34 Barcode Field Commands (Continued)

ZPL Command	Command Description	Ratio
^BI	Industrial 2 of 5	[2.0 - 3.0]
^BJ	Standard 2 of 5	[2.0 - 3.0]
^BK	ANSI Codabar (USD-4 and 2 of 7)	[2.0 - 3.0]
^BL	LOGMARS	[2.0 - 3.0]
^BM	MSI	[2.0 - 3.0]
^B0	Aztec Barcode Parameters	[Fixed]
^BP	Plessey	[2.0 - 3.0]
^BQ	QR Code (*)	[Fixed]
^BR	GS1 Databar (formerly RSS)	[Fixed]
^BS	UPC/EAN Extensions (*)	[Fixed]
^BU	UPC-A (*)	[Fixed]
^BX	Data Matrix (*)	[Fixed]
^BZ	PostNet (*), USPS Intelligent Mail, and Planet barcodes	[Fixed]

Additionally, each barcode field command can be issued with a definition parameter string. The parameter string defines field rotation, height, and interpretation line status for all barcodes. For some barcodes, the parameter string also sets a check digit, start character, and/or stop character. Use the definition parameter string to command the printer to print barcodes of appropriate heights and densities that conform to the specifications of the application.

The use of the parameter string is optional because all parameters have default values. If the default values for all of the barcode parameters suit the application, then only the barcode command needs to be entered.

Parameters in barcode field commands are “position specific.” If a value (other than the default value) is manually entered for one parameter the ZPL II delimiter character (a comma) must be used to mark the position of the preceding parameters in the string.

To change just the third parameter, enter two commas and then the value for the third parameter. The default values will be automatically used for the first and second parameters.

Bar Code Command Groups

Bar code commands are organized into four groups.

Each group represents a type of bar code. The following tables identify the groups and the bar codes they contain:

Table 35 Numeric Only Bar Codes

ZPL Command	Command Description
^B0	Aztec Bar Code Parameters
^B1	Code 11

Table 35 Numeric Only Bar Codes (Continued)

ZPL Command	Command Description
^B5	Planet Code Bar Code
^BI	Industrial 2 of 5
^BJ	Standard 2 of 5
^BK	ANSI Codabar (or NW-7)
^BM	MSI
^BO	Aztec Bar Code Parameters
^BP	Plessey
^BZ	PostNet (*), USPS Intelligent Mail, and Planet bar codes

Table 36 Retail Labeling Bar Codes

ZPL Command	Command Description
^B0	Aztec Bar Code Parameters
^B8	EAN-8
^B9	UPC-E
^BE	EAN-13
^BO	Aztec Bar Code Parameters
^BS	UPC/EAN extensions
^BU	UPC-A

Table 37 Alphanumeric Bar Codes

ZPL Command	Command Description
^B0	Aztec Bar Code Parameters
^B3	Code 39
^BA	Code 93
^BC	Code 128
^BL	LOGMARS
^BO	Aztec Bar Code Parameters

Table 38 Two-Dimensional Bar Codes

ZPL Command	Command Description
^B0	Aztec Bar Code Parameters
^B4	Code 49
^B7	PDF417

Table 38 Two-Dimensional Bar Codes (Continued)

ZPL Command	Command Description
^BB	CODABLOCK
^BD	UPS MaxiCode
^BF	MicroPDF417
^BQ	QR Code
^B0	Aztec Bar Code Parameters
^BR	GS1 Databar (formerly RSS)
^BT	TLC39
^BX	Data Matrix

Mod 10 and Mod 43 Check Digits

This section provides information about Mod 10 and Mod 43 check digits.

Mod 10 Check Digit

This section lists the calculations for determining the Mod 10 Check Digit character.

The calculations for determining the Mod 10 Check Digit character are as follows:

1. Start at the first position and add the value of every other position together.

   0 + 2 + 4 + 6 + 8 + 0 = 20

2. The result of Step 1 is multiplied by 3.

   20 x 3 = 60

3. Start at the second position and add the value of every other position together.

   1 + 3 + 5 + 7 + 9 = 25

4. The results of steps 2 and 3 are added together.

   60 + 25 = 85

5. The check character (12th character) is the smallest number which, when added to the result in step 4, produces a multiple of 10.

   85 + X = 90 (next higher multiple of 10)

   X = 5 Check Character

   This bar code illustrates the above example. The digit on the right (5) is the check digit.

Mod 43 Check Digit

This section lists the calculations for determining the Mod 43 check Digit character.

The calculations for determining the Mod 43 check Digit character are as follows:

Each character in the Code 39 character set has a specific value, as follows:

0=0	B=11	X=33
1=1	C=12	Y=34
2=2	D=13	Z=35
3=3	E=14	- =36
4=4	F=15	. =37
5=5	G=16	Space = 38
6=6	H=17	$=39
7=7	I=18	/=40
8=8	J=19	+=41
9=9	K=20	%=42
A=10	L=21	

Example

Data string 2345ABCDE/

1. Add the sum of all the character values in the data string. Using the chart above, the sum of the character values is as follows:

   1 + 2 + 3 + 4 + 5 + 10 + 11 + 12 + 13 + 14 + 40 =115

2. Divide the total by 43. Keep track of the remainder.

   115/43 = 2 Remainder is 29

3. The “check digit” is the character that corresponds to the value of the remainder.

   Remainder = 29

   29 is the value for the letter T.

   T is the check digit.

   Below is a bar code that illustrates the example. The character on the right, T, is the check digit.

```
^F0125,100^B3N,Y,150,Y,N^FD12345ABCDE/^FS
```

Error Detection Protocol

This section explains the Zebra protocol that has been supplanted in TCP/IP based applications because of the error detection compatibility inherent in the TCP/IP protocol.

Introduction

There are many instances when it is vitally important that the information sent to the Zebra printer is received completely Error-Free. ZPL II supports an error detection protocol called Zebra Packet Response Protocol to meet this need.

NOTE: This protocol only works when using serial interface. It does not function when using parallel interface.

What is a Protocol?

A protocol is a precisely defined set of rules. In the case of data communications, a Protocol defines how data is transmitted, received, and acknowledged between two devices.

The sole purpose of the Packet Response Protocol is to ensure that the information sent from a Host computer to the Zebra printer is received accurately. Remember, the protocol cannot insure the accuracy of the data that is actually sent from the Host computer. The commands and data needed to make a label (ZPL II Format) are encapsulated within the information sent from the Host computer.

How Protocols Work

The basic unit of data transfer in the Packet Response Protocol is called a “Transaction.” A Transaction is a two-way communication procedure that consists of information being sent from the Host computer to the Zebra printer, and the printer sending back a response to the Host computer. This response is an indication that the Zebra printer has either accepted or rejected the information sent from the Host computer.

Information is sent in the form of “Packets.” Packets sent from the Host computer are called Request Packets.

When a Request Packet is received, the Zebra printer analyzes the information in the Packet. If the Request Packet is accepted, the Zebra printer will send a positive response back to the Host computer. The Host computer can then send the next Request Packet. If the information is rejected, the Zebra printer will send a negative response back to the Host computer. The Host computer then sends the same Request Packet again.

The Zebra Packet Response Protocol can be used in both single-printer applications, where there is only one Zebra printer connected to the Host computer, and multi-drop systems in which several Zebra printers are connected to the same Host computer.

Request Packet Formats from the Host Computer

The first part of each data transfer Transaction is the sending of a Request Packet by the Host computer. The Request Packet contains a fixed length “Header” block and a variable length “Data” block. Each Packet sent from the Host computer to the Zebra printer must always use the following format.

The Request Packet Header Block is comprised of five fixed-length fields. The Request Packet Data Block is comprised of four fixed-length fields and one variable-length field. These fields are defined as follows.

Header Block					Data Block				
SOH	DST. Z-ID	SRC. Z-ID	TYPE	SEQ. #	STX	FORMAT	EXT	CRC	EOT
1	3	3	1	1	1	≤1024	1	2	1

Header Block Fields

This section provides a list of header block fields.

- SOH (start of header character)

  The Zebra printer interprets this character as the beginning of a new Request Packet. The ASCII Control Code character SOH (01H) is used as the Start of Header Character.

- DST. Z-ID (destination Zebra-ID)

  This is the three-digit ASCII I.D. number used to identify which Zebra printer is to receive the Request Packet. The Zebra printer compares this number to the Network ID number assigned to it during Printer Configuration. The Zebra printer will act on the Request Packet only if these numbers match.

- SRC. Z-ID (source Zebra-ID)

  This is a three-digit ASCII number used to identify the Host computer. This number is determined by the user.

- TYPE (packet type)

  This field is used to define the type of Request Packet being sent by the Host. Only two characters are valid in this field:

  - ‘P’ indicates a Print Request Packet
  - ‘I’ indicates an Initialize Request Packet

  Most of the Packets sent by the Host to the Zebra printer will be of the ‘P’ variety, requesting a label to be printed.

  The ‘I’ character tells the Zebra printer to initialize the packet sequence numbering. It is required in the first packet of a new printing session, after starting up the Host computer or the Zebra printer.

- SEQ. # (the sequence number of the request packet)

  This block contains a single digit number used to denote the current Transaction Number. The Host computer must increment this number by “1" for each new Request/Response Transaction pair, i.e. 0, 1, 2,..., 9. The numbers repeat after every 10 Transactions.

Data Block Fields

This section provides a list of data block fields.

- STX (Start of Text)

  The Zebra printer interprets this character as the beginning of the variable-length Data Format portion of the Request Packet. The ASCII Control Code character STX (02H) is used as the Start of Text Character.

- DATA FORMAT (Label Information)

  A variable-length portion of the Request Packet that contains the complete or partial ZPL II label format, or partial data string (such as a downloaded graphic).

  This field can contain from 0 to 1024 characters. If the Format of a label is longer than 1024 characters, the Data Format fields from consecutive packets will be concatenated together in the printer's Receive Data Buffer as if they were sent as one long direct transmission.

  Special consideration has been given to the possible requirement to include ASCII Control Characters (values less than 20H) in the Data Format portion of a Request Packet. Characters such as EOT (04H), STX (02H), SOH (01H), and ETX (03H), are part of the Error Detection Protocol and could interrupt normal communication procedures if received at the wrong time.

- ETX (End of Text)

  The Zebra printer interprets this character as the end of the variable length Data Format portion of the Request Packet. The ASCII Control Code character ETX (03H) is used as the End of Text Character.

- CRC (Cyclic Redundancy Check)

  The CRC is a 2 character field. A Cyclic Redundancy Check is a type of error checking used to maintain the validity and integrity of the information transmitted between the Host computer and the Zebra printer. This Protocol uses the 16-bit CCITT method of producing a CRC.

  The CRC is a two-byte value derived from the contents of the packet between, but not including, the SOH character and the CRC code itself. The Zebra printer will calculate a CRC of the Request Packet received and compare the value with the CRC Value in this field. The CRC of the Request Packet must match the CRC calculated by the Zebra printer in order for the Request Packet to be valid.

- EOT (End of Transmission)

  The Zebra printer interprets this character as the end of the Request Packet. The ASCII Control Code character EOT (04H) is used as the End of Transmission Character.

Response From the Zebra Printer

When the Zebra printer receives the EOT character, it will begin acting on the Request Packet received. The printer will compare certain characters and numeric values within the received Request Packet and send a response back to the Host computer.

Zebra Packet Response

The Packet Response protocol provides the highest degree of error checking and is well suited to the Host-Multiple Printer application. The Response Packet from the Zebra printer will always use the following format.

The Request Packet Header Block is comprised of five fixed-length fields. The Request Packet Data Block is comprised of four fixed-length fields and one variable-length field. These fields are defined as follows.

Header Block					Data Block				
SOH	DST. Z-ID	SRC. Z-ID	TYPE	SEQ. #	STX	FORMAT	EXT	CRC	EOT
1	3	3	1	1	1	≤1024	1	2	1

Header Block Fields

This section provides descriptions for the header block fields.

- SOH (Start of Header Character)

  The Zebra printer sends this character as the beginning of a new Response Packet. The ASCII Control Code character SOH (01H) is used as the Start of Header Character.

- DST. Z-ID (Destination Zebra-ID)

  This is the same three-digit ASCII number used to identify the Host Computer that was contained in the SRC. Z-ID field of the Request Packet that initiated this Response Packet. The Host compares this number to its known value to insure it is the proper destination.

- SRC. Z-ID (Source Zebra-ID)

  This is the three character ASCII Network I.D. of the Zebra printer that is sending the Response Packet.

- TYPE (Packet Type)

  This block is used to define the type of Response Packet being sent to the Host. Only three characters are valid in this field.

  - ‘A’ This is a Positive Acknowledgment to the Host computer. It indicates that the Request Packet was received without a CRC error. The Host computer may send the next Request Packet.
  - ‘N’ This is the Negative Acknowledgment to the Host computer. It indicates that an error was detected in the packet sent from the Host computer. The Host computer must retransmit the same Request Packet again.
  - ‘S’ This character indicates that the Response Packet contains the Zebra Printer Status requested by a ~HS (Host Status) command received from the Host.

- SEQ. # (Used to denote the current message sequence number)

  This number is identical to the message sequence number in the Request Packet. It denotes the message sequence number to which the Response Packet is replying.

Data Block Fields

This section provides descriptions for the data block fields.

- STX (Start of Text)

  The Zebra printer sends this character as the beginning of the variable length Data Format portion of the Response Packet. The ASCII Control Code character STX (02H) is used as the Start of Text Character.

- DATA FORMAT (Label Information)

  The ‘variable length’ portion of the Response Packet. If the Packet Type field in the Response Header contains an ‘A’ or an ‘N’, no data will appear in this field. If the Packet Type field contains an ‘S‘, this field will contain the Printer Status Message.

- ETX (End of Text)

  The Zebra printer sends this character as the end of the variable length Data Format portion of the Request Packet. The ASCII Control Code character ETX (03H) is used as the End of Text Character.

- CRC (Cyclic Redundancy Check)

  This is the CRC of the Response Packet as calculated by the Zebra printer. This Cyclic Redundancy Check maintains the validity and integrity of the information transmitted between the Zebra printer and the Host computer.

  This CRC is a two Byte value derived from the contents of the packet between, but not including, the SOH character and the CRC code itself. The Host computer will calculate a CRC of the received Response Packet and compare it to the CRC value in this field. The CRC of the Response Packet must match the CRC calculated by the Host computer in order for the Response Packet to be valid.

- EOT (End of Transmission)

  The Zebra printer sends this character as the end of the Response Packet. The ASCII Control Code character EOT (04H) is used as the End of Transmission Character.

Disguising Control Code Characters

There may be occasions when ASCII Control Codes (00H - 19H) must be included as part of the Data Format block of a Request Packet. To eliminate any problems, these characters must be disguised so that the communication protocol does not act on them.

This procedure must be used to disguise each Control Code.

- A SUB (1AH) character must precede each Control Code placed in the Data Format block.
- The value of 40H must be added to the Hex value of the Control Code.
- The ASCII Character corresponding to the total value produced in step 2 must be entered in the Data Format right after the SUB character.

The Zebra printer automatically converts the modified control character back to its correct value by discarding the SUB (1AH) character and subtracting 40H from the next character.

To include a DLE (10H) character in the Data Format block:

1. Enter a SUB (1AH) character into the Data Format.

2. Add 40H to the DLE value of 10H for a resulting value of 50H.

3. Enter the ASCII character “P” (50H) in the Data Format after the SUB character.

NOTE: This technique is counted as two characters of the 1024 allowed in the Data Format block.

Rules for Transactions

This section lists the rules for transactions.

- Every Transaction is independent of every other Transaction and can only be initiated by the Host computer.
- A valid Response Packet must be received by the Host computer to complete a Transaction before the next Request Packet is sent.
- If an error is encountered during a Transaction, the entire Transaction (i.e., Request Packet and Response Packet) must be repeated.

- The Zebra printer does not provide for system time-outs and has no responsibility for insuring that its Response Packets are received by the Host computer.
- The Host computer must provide time-outs for all of the Transactions and insure that communication continues.
- If any part of a Transaction is lost or received incorrectly, it is the responsibility of the Host computer to retry the whole Transaction.

Error Detection Protocol Application

The following are the basic requirements for setting up the Zebra printer to use the Error Detection Protocol.

Activating the Protocol

Protocol is a front panel selection, or can be done with the ZPL command ^SC.

Setting Up Communications

Insure that the Host computer and the Zebra printer are characterized with the same communication parameters; i.e., Parity, Baud Rate, etc. The communications must be set up for 8 data bits.

Setting the Printer ID Number

The Protocol uses the printer's Network ID number to insure communication with the proper unit. The Network ID is programmed into the printer by sending the printer a ^NI (Network ID Number) command or done through the front panel.

If there is only one printer connected to the Host computer, the Network ID number should be set to all zeros (default).

If there is more than one printer, such as in a broadcast or multi-drop environment, each printer should be assigned its own unique ID number. Printers in this environment, with an ID of all zeros, will receive ALL label formats regardless of the actual printer ID number in the DST. Z-ID block of the Request Packet.

Error Conditions and System Faults

This section describes error conditions and system faults that might occur.

Restarting a Transmission

If a break in communication occurs, the Host must restart the transmission of the current label format with an Initialization Request Packet. The Zebra printer will not respond to Request Packets sent out of sequence. However, the Zebra printer will respond to an Initialization Request Packet and restart its internal counting with the sequence number of the Request Packet.

CRC Error Conditions and Responses

A CRC error condition can be detected when the printer receives a Request Packet or when the Host computer receives a Response Packet. The following list defines these errors and how the Host computer should respond to them.

Error	Response
The CRC calculated by the Zebra printer does not match the one received as part of the Request Packet.	The Zebra printer will return a Negative Acknowledgment Response Packet. The Host computer should retry the same Transaction with the same Sequence Number.

Error	Response
The CRC calculated by the Host computer does not match the one received as part of the Response Packet.	The Host computer should retry the same Transaction with the same Sequence Number.

Time-Out Error Conditions and Responses

There are certain conditions at the Zebra printer that might cause the Host computer to time-out while processing a Transaction. The following list illustrates these conditions and how the Host computer should respond to them.

Error	Response
A Request Packet from the Host computer is not received by the Zebra printer.	The Host computer times out and resends the Request Packet of the same Transaction with the same Sequence Number.
A Request Packet from the Host computer is partially received by the Zebra printer.	The Host computer times out and resends the Request Packet of the same Transaction with the same Sequence Number.
A Response Packet from the Zebra printer is not received by the Host computer.	The Host computer times out and resends the Request Packet of the same Transaction with the same Sequence Number.
A Response Packet from the Zebra printer is partially received by the Host computer.	The Host computer times out and resends the Request Packet of the same Transaction with the same Sequence Number.

How the Zebra Printer Processes a Request Packet

The following describes the steps taken at the Zebra printer to process a Request Packet.

1. The Zebra printer looks for a SOH (Start of Header) character. As soon as it finds one, it places the SOH and all the data after it into its Receive Data Buffer. This process continues until the printer receives an EOT (End of Transmission) character.

NOTE: If a second SOH is received before an EOT is detected, the contents of the Receive Buffer will be discarded. All of the data after the second SOH will be placed in the Receive Data Buffer.

2. After detecting the EOT, the printer checks for the following:

   * The DST. Z-ID matches the printer's Network I.D.

NOTE: If the Network ID at the printer is all zeros, the printer will accept all Request Packets regardless of the DST. Z-ID received. If a Request Packet is received with the DST. Z-ID all zeros, it is accepted by all printers regardless of their Network ID setting.

   *The Data Format begins with STX and ends with ETX.

   *The Sequence Number has not been used before.

   If the check is satisfactory, proceed to Step 3 on the following page.

   If any part of the check is unsatisfactory, the printer discards the data in its Receive Data Buffer and waits for another SOH. No response is sent to the computer.

Exceptions

It is possible that the printer will send a response to the host that the host does not receive. Therefore, the host will send the same request packet to the printer again. If this happens, the printer will not use the data if it already used it before. However, the printer will send a response back to the host.

The printer calculates the CRC and compares it with the one received in the Request Packet. If the CRC is valid, the printer sends a Positive Response Packet to the Host computer. It then transfers the 'Variable Length' data from the Receive Buffer to its memory for processing. If the CRC does not match, and the printer is set up to return a Negative Response Packet, the following will take place:

1. The printer assumes that the DST. Z-ID, SRC. Z-ID, and Sequence Number are correct and that the error was in the variable data.
2. The same DST. Z-ID, printers SRC. Z-ID, and Sequence Number will be returned back to the host in the Negative Response Packet.
3. If the assumption in (a) is incorrect, the Host computer can time-out and retransmit the original Request Packet.

How the Zebra Printer Responds to Host Status

The following describes how the Zebra printer to responds to host status command.

If a ~HS (Host Status) command is received by the Zebra printer, the printer will send back an acknowledgment for the receipt of the packet. It then sends an additional packet that includes the Host Status information in the Variable Length portion of the packet.

ZB64 Encoding and Compression

This section describes the Base 64 MIME (ZB64) encoding and compression.

This is the same type of MIME encoding that is used in e-mail.

For more information on ZB64 Encoding and Compression, contact your Reseller or Zebra Representative.

Introduction to B64 and Z64

The first encoding, known as B64, encodes the data using the MIME Base64 scheme. Base64 is used to encode e-mail attachments and is specifically designed to address communications path limitations, such as control characters and 7-bit data links.

It encodes the data using only the printable ASCII characters:

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

a b c d e f g h I j k l m n o p q r s t u v w x y z

0 1 2 3 4 5 6 7 8 9 + / =

With the use of ZPL, this has the added benefit of avoiding the caret (^) and tilde (~) characters. Base64 encodes six bits to the byte, for an expansion of 33 percent over the un-enclosed data. This is much better than the 100 percent expansion given by the existing ASCII hexadecimal encoding.

The second encoding, known as Z64, first compresses the data using the LZ77 algorithm to reduce its size. (This algorithm is used by the PKWARE® compression program PKZIP™ and is integral to the PNG graphics format.) The compressed data is then encoded using the MIME Base64 scheme as described above.

A CRC is calculated across the Base64-encoded data. If the CRC-check fails or the download is aborted, the object can be invalidated by the printer.

The robust encodings can be piggybacked on the existing download commands with full backward compatibility. This is done by prefacing the new encodings with a header that uniquely identifies them. The download routines in the printer firmware can key-off the header to determine whether the data is in the old ASCII hexadecimal encoding or one of the new encodings. This allows existing downloadable objects to be used in their present format, while new objects can be created using the same download commands with the new encodings for increased integrity and reduced download times.

For easy reference, B64 and Z64 are referred to as ZB64. In any reference to the ZB64 encoding, assume that both Base64-only (B64) and LZ77/Base64 (Z64) encodings are accepted.

The following is an example of an existing download command using the new encoding:

```
~DTARIAL,59494,:Z64:H4sICMB8+DMAC0FSSUFMLlRURgDsmnd8VEW7x5+ZOedsyibZNNJhly
WhbEJIwYSwJDGNkmwghJIgJYEEEhQIPSggKAjEAiIiVaSoIJYNBAkIGgGxUBVUUCGU0JQSC0WFnPvbE
+SF18+9H+
8f973X+3Jm93umzzNznvnNSSFGRJ6ARAVZvXK7XDaXLyTiR5B7ontuZPQ824I5RKIa6ew
+aba8+pU1rVDZiciv
```

[multiple lines deleted]

```
/O6DU5wZ7ie2+g4xzDPwCpwm3nqW2GAPcdclxF4fIP66jHjncmKvKzh/ZUNCxl9/QQx2HXHYB4m/
PkQcdCdx2G7OY
t+mszkMh4iZxoifvkh89BFipo87kwD/Bf/dOcycAAEA:a1b2
```

The parameters are identical to the existing ~DT command:

Parameter	Details
o = font name	Values: any valid TrueType name, up to 8 characters Default: if a name is not specified, UNKNOWN is used In this example, Arial is the specified font.
s = font size	Values: the number of memory bytes required to hold the Zebra-downloadable format of the font Default: if an incorrect value or no value is entered, the command is ignored In this example, 59494 is the size. To maintain compatibility with the existing ASCII hexadecimal encoding, this field must contain the size of the un-enclosed and uncompressed object — the number of bytes that are finally placed into the printer's memory, not the number of bytes downloaded.
data = data string	Values: a string of ASCII hexadecimal values (two hexadecimal digits/byte). The total number of two-digit values must match parameter s. Default: if no data is entered, the command is ignored Everything following the size field is data. The new encoding imposes a header with a unique signature. The new encoding must start with the characters :B64: (data encoded in Base-64 only) or :Z64: (data compressed with LZ77, then encoded in Base-64) followed by the encoded data. After the data is presented, another colon (:) and four hexadecimal digits comprise the CRC. The Base64 standard allows new-line characters (carriage returns and line feeds) to be inserted into the encoded data for clarity. These characters are ignored by the printer.

When downloading graphics, the colon is used in the current ASCII hexadecimal encoding indicate "repeat the previous dot row." Since this shorthand is invalid for the first character of data (no previous dot row has been downloaded), it is safe for the printer to detect the leading colon character as the lead-in for the new encodings.

B64 and Z64 Encoding

These download encodings, B64 and Z64, are created as drop-in replacements for the existing ASCII hexadecimal encoding.

B64 encoding do the following:

- Encode the compressed data using the MIME Base64 algorithm.
- Calculate a CRC across the encoded data.
- Add a unique header to differentiate the new format from the existing ASCII hex encoding.

Z64 encoding do the following:

- Compress the data using the LZ77 algorithm.
- Encode the compressed data using the MIME Base64 algorithm.
- Calculate a CRC across the encoded data.
- Add a unique header to differentiate the new format from the existing ASCII hexadecimal encoding.

The data field have this format:

```
:id:encoded_data:crc
```

Parameter	Details
`:id`	The identifying string B64 or Z64.
`:iencoded_data`	Data to download, compressed with LZ77 (if the id parameter is set to Z64) and encoded with Base64.
`:crc`	Four hexadecimal digits representing the CRC calculated over the :encoded_data field.

The printer calculates a CRC across the received data bytes and compare this to the CRC in the header. A CRC mismatch is treated as an aborted download.

The B64 and Z64 encodings can be used in place of the ASCII hexadecimal encoding in any download command. The commands are:

`~DB`	Download Bitmap Font
`~DE`	Download Encoding
`~DG`	Download Graphic
`~DL`	Download Unicode Bitmap Font
`~DS`	Download Scalable Font
`~DT`	Download TrueType Font
`~DU`	Download Unbounded TrueType Font
`^GF`	Graphic Field (with compression type set to “ASCII hex”)

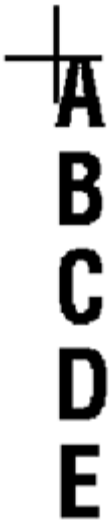
The ~DB (Download Bitmap Font) command can use the new encodings in place of the ASCII hexadecimal encoding in data sub-fields. Each character is encoded individually. However, for small amounts of data, the identifying B64 or Z64 header and trailing CRC may negate any gains made by using the new format.

For backward compatibility, the ^HG (Host Graphic) command uses the ASCII hexadecimal encoding. It does not use the new encodings.

Field Interactions

This section provides you with examples that show how commands interact with various justification parameters. These charts are designed so that you can identify the location of the field origin and interactions between the rotation, formatting and justification commands.

Normal Orientation

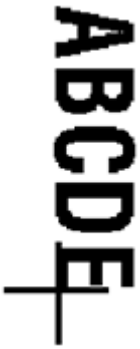
	^FPH	^FPV	^FPR
^FO Left Justified			

	^FPH	^FPV	^FPR
^FT Left Justified			
^FO Right Justified			

	^FPH	^FPV	^FPR
^FT Right Justified			

Rotated Orientation

	^FPH	^FPV	^FPR
^FO Left Justified			
^FT Left Justified			

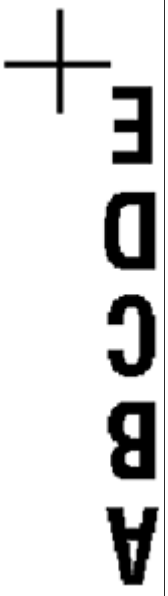
	^FPH	^FPV	^FPR
^FO Right Justified			
^FT Right Justified			

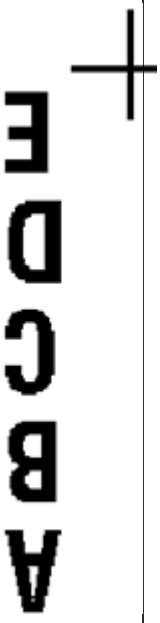
Bottom Up Orientation

	^FPH	^FPV	^FPR
^FO Left Justified			

	^FPH	^FPV	^FPR
^FT Left Justified			
^FO Right Justified			
^FT Right Justified			

Inverted Orientation

	^FPH	^FPV	^FPR
^FO Left Justified			
^FT Left Justified			

	^FPH	^FPV	^FPR
^FO Right Justified			
^FT Right Justified			

Real Time Clock

This appendix contains the information needed to install, program, and operate the Real Time Clock (RTC) option. This hardware option is available as either a factory-installed or field-installable option in specific printer products manufactured and sold by Zebra Technologies Corporation.

The Real Time Clock option is currently available for following printers, and requires that the firmware version shown is installed on that printer.

Supported Zebra Printer and Print Engine Models		Requires this Firmware Version or Later
105SL printers that meet any one of the following criteria: • if the printer was manufactured after April 2006 • if the RTC Date and the RTC Time fields are printed on the configuration label • if the RTC Date and the RTC Time appear on the LCD display		60.13.0.13Z
DA402 & T402 printers		32.8.4
LP2844-Z, TPL2844-Z, and TPL3844-Z printers		45.10.x
PAX Series print engines	170PAX2 print engines	29.9.x or 31.9.x
	110PAX3 print engines using Standard Font	34.10.x
	110PAX3 print engines using TT Font	49.10.x
	116PAX3 print engines using TT Font	35.10.x
	170PAX3 print engines using Standard Font	37.10.x
	170PAX3 print engines using TT Font	38.10.x
	110PAX4 and 170PAX4 print engines	60.13.0.12
S4M printers (field-installable kit)		50.13.x
S600 printers		27.10.3
Xi Series printers	90XiII, 140XiII, 170XiII & 220XiII printers	18.9.x
	90XiIII, 96XiIII, 140XiIII, 170XiIII, and 220XiIII printers	33.10.0
	90XiIIIPlus, 96XiIIIPlus, 140XiIIIPlus, 170XiIIIPlus, and 220XiIIIPlus printers	60.13.0.12
Z4Mplus and Z6Mplus printers		60.13.0.12
ZM400/ZM600/RZ400/RZ600 printers		53.15.xZ

Control Panel Programming

New parameters for the Real Time Clock have been added to the Control Panel Configuration. These parameters are located immediately following the **FORMAT CONVERT** prompt. Refer to the printer/print engine User Guide for complete configuration information.

- X.9.x firmware added the parameters to XiII printers and 170PAX/170PAX2 print engines.
- X.10.x firmware added the parameters to XiIII printers and PAX3 print engines.
- X.13.x firmware added the parameters to XiIIIPlus printers, PAX4 print engines.
- X.13.0.13Z firmware added the parameters to 105SL printers.

Real Time Clock Parameters

The parameters listed on the following pages are added to the Control Panel Configuration prompts only when both the Real Time Clock hardware option and the appropriate version of firmware are installed.

- X.9.x or later firmware installed in the XiII series printers or the 170PAX/170PAX2 series print engines.
- X.10.x or later firmware installed in the XiIII series printers or the PAX3 series print engines.
- X.13.0.13Z or later firmware installed in the 105SL printers
- X.13.x or later firmware installed in the XiIIIPlus series printers, the PAX4 series print engines, or the S4M printers.

The RTC ZPL II commands apply to all printers/print engines with the Real Time Clock hardware option and proper firmware.

Idle Display

Selects the printer/print engine Idle Display format and the method of displaying the time/date information.

This parameter also affects the Configuration Label printout and the RTC DATE and RTC TIME formats.

Selections

- FW VERSION
- MM/DD/YY 24HR
- MM/DD/YY 12HR
- DD/MM/YY 24HR
- DD/MM/YY 12HR

If FW VERSION is selected, the format on the Configuration Label and on the RTC DATE and RTC TIME parameters is MM/DD/YY 24HR.

RTC Date

Allows entry of the RTC date in the format selected by the **IDLE DISPLAY** parameter.

NOTE: The RTC parameters are password-protected. Refer to your printer's user guide for specific instructions on accessing and modifying printer parameters.

Printer Model	Action
PAX Series print engines, Xi Series printers, and 105SL printers	Use the LEFT oval key to select the position to be adjusted Then, use the RIGHT oval key to select the correct value for that position.
Z4Mplus and Z6Mplus printers	Press **SELECT** to select the parameter. Use the **MINUS** (-) key to select the position to be adjusted Then, use the **PLUS** (+) key to select the correct value for that position. Press **SELECT** to accept any changes and deselect the parameter.
S4M printer	Press ENTER. The printer displays the current RTC date. Modify the values as follows: • Press the right arrow to move to the next digit position. • To increase the value, press the up arrow. • To decrease the value, press the down arrow. Press **ENTER** to accept the value shown.

NOTE: Invalid dates, such as 2/30/1999, may be entered, but they will not be saved.

RTC Time

Allows entry of the RTC time in the format selected by the **IDLE DISPLAY** parameter.

NOTE: The RTC parameters are password-protected. Refer to your printer's user guide for specific instructions on accessing and modifying printer parameters.

Printer Model	Action
PAX Series print engines, Xi Series printers, and 105SL printers	Use the LEFT oval key to select the position to be adjusted Use the RIGHT oval key to select the correct value for that position.
Z4Mplus and Z6Mplus printers	Press **SELECT** to select the parameter. Use the **MINUS** (-) key to select the position to be adjusted. Use the **PLUS** (+) key to select the correct value for that position. Press **SELECT** to accept any changes and deselect the parameter.
S4M printer	Press ENTER. The printer displays the current RTC date. Modify the values as follows: • Press the right arrow to move to the next digit position. • To increase the value, press the up arrow. • To decrease the value, press the down arrow. Press **ENTER** to accept the value shown.

RTC General Information

The Real Time Clock commands are only applicable if the Real Time Clock option is installed in the printer. For those printers with an LCD control panel display, additional control panel configuration parameters are also included.

The ZPL II Field Clock ^FC command is used to specify the clock-indicator character for the primary, secondary, and third clocks. This command must be included within each label field command string whenever the date or time clock values are required within the field. No date or time clock information can be printed in a label field unless this command is included. The ^FC command can now be combined with the ^SN command in V60.13.0.10 and later.

A clock-indicator can be any printable character except the ZPL II Format Prefix, Control Prefix, or Delimiter characters. The default value for the primary clock-indicator is the percent sign %. The secondary and third clock-indicators have no defaults and must be specified in order for that clock to be used.

The Field Data ^FD command has been expanded to recognize the clock-indicators and associated command characters, and to replace them during the printing process with the corresponding time or date parameter. For example, if the primary clock-indicator is the percent sign %, then during printing, the character sequence %H in the ^FD statement would be replaced by the 2-digit current hour.

NOTE: If the Real Time Clock is not installed, or the ^FC command has not preceded the ^FD statement, no replacement would occur. In this case, the characters %H would print as text on the label.

The name of the day of the week, the name of the month, and the AM or PM designation can also be inserted in place of a specific clock-indicator/command character sequence. This table lists command characters and their functions.

Table 39 Command Characters

Command Character	Function
%a	is replaced by the abbreviated weekday name
%A	is replaced by the weekday name
%b	is replaced by the abbreviated month name
%B	is replaced by the month name
%d	is replaced by the day of the month number, 01 to 31
%H	is replaced by the hour of the day (military), 00 to 23
%I	is replaced by the hour of the day (civilian), 01 to 12
%j	is replaced by the day of the year, 001 to 366
%m	is replaced by the month number, 01 to 12
%M	is replaced by the minute, 00 to 59
%p	is replaced by the AM or PM designation
%S	is replaced by the seconds, 00 to 59
%U	is replaced by the week# of the year, 00 to 53, Sunday is 1st day[1]
%W	is replaced by the week# of the year, 00 to 53, Monday is 1st day[2]
%w	is replaced by the day# of the week, 00 (Sunday) to 06 (Saturday)

Table 39 Command Characters (Continued)

Command Character	Function
%y	is replaced by the 2 digits of the year, 00 to 99
%Y	is replaced by the full 4 digit year number—where% is the specified clock-indicator character

1. %U establishes Sunday as the first day of the year.

2.%W establishes Monday as the first day of the year.

The Set Offset ^S0 command permits the printing of specific times and dates relative to the primary clock. The secondary (or third) clock is enabled when secondary (or third) offsets are entered using this command. The secondary (or third) clock time and date are determined by adding the offsets to the current clock reading.

One ^S0 command is required to set the secondary offset; an additional ^S0 command is required for a third offset. The offsets remain until changed or until the printer is either powered down or reset.

NOTE: Only dates from January 1, 1998 to December 31, 2097 are supported. Setting the offsets to values that result in dates outside this range is not recommended and may have unexpected results.

The Set Mode/Language (see ^SL) command is used to select the language the days of the week and the months are printed in. This command also sets the printing mode, which can be S for START TIME, T for TIME NOW, or a Numeric Value for the time accuracy. In START TIME mode, the time printed on the label is the time that is read from the Real Time Clock when the label formatting begins (when the ^XA command is received by the printer). In TIME NOW mode, the time printed on the label is the time that is read from the Real Time Clock when the label is placed in the queue to be printed. In Numeric Value mode, a time accuracy tolerance can be specified.

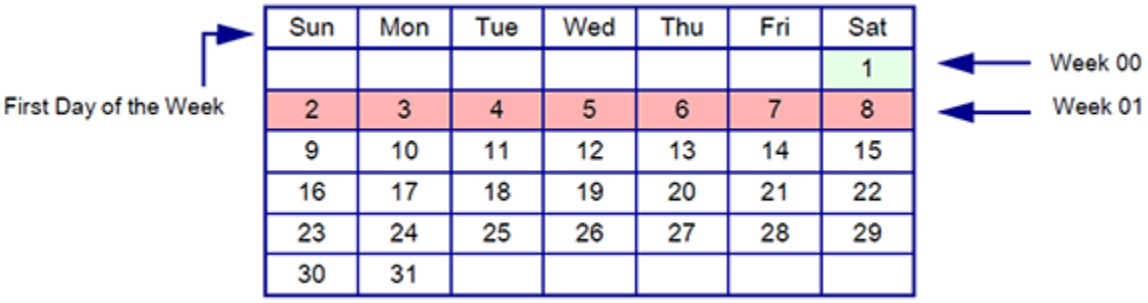
First Day of the Week Affects Calendar Week

The %U and %W commands set the first day of the week. The week numbering starts at the beginning of the year with Week 01 representing the first full week of the year. Any day(s) before that established first day of the week are part of the Week 00. The following examples show how setting different days as the first day of the week affect the calendar week.

IMPORTANT: The %U and %W commands determine the numbering for all weeks in the year.

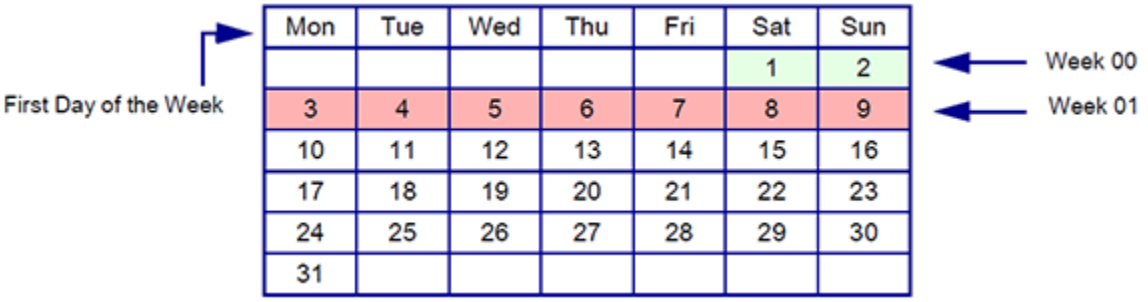
January, 2005 with Week 00

Set Sunday as the first day of the week using the %U command. In this example, notice that Saturday, January 1st is Week 00 and Sunday, January 2nd begins Week 01.



January, 2005 with Week 00

Set Monday as the first day of the week using the %W command. In this example, notice that Saturday, January 1st and Sunday, January 2nd are Week 00 and Monday, January 3rd begins Week 01.



January, 2006 without Week 00

Set Sunday as the first day of the week using the %U command. Since 2006 begins on a Sunday, there is no Week 00 in this example.

Time and Date Precision

As of V60.13.0.1 firmware, the ^C0 command is now ignored. While the S4M printer has a lower firmware version number (V50.x), its firmware was recently released and follows the rule to ignore the ^C0 command.

The time and date placed in a label field is determined at the time the label bitmap is created by the printer (start time mode). If a batch of labels is formatted, the date and time will be the same for all labels in the batch. If the printer is paused during the printing process and remains in that state for a period of time, when printing resumes, the time and date will still be the same as when the batch was first started.

If more precise time and date stamps are required on versions prior to V60, follow the process below. For versions after V60, use the Numeric Value mode as shown in ^SL.

Cycle the printer/print engine power Off (O) and On (I) to clear the memory before performing the steps below.

1. Print a Memory Usage Label (^XA^WD*:*.*^XZ) and note the following value: Available RAM (in BYTES)	(A)	_____
2. Print a Configuration Label and note these values: Printer “Print Width” (in DOTS) (NOT the Label Width)	(B)	_____
Label Length (in DOTS)	(C)	_____

3. Determine the desired maximum number of queued labels with the same Time and Date value.	(D) _____

NOTE: Increasing the number of queued labels will improve throughput performance, but Real Time Clock values will be less accurate. Two is usually a good compromise.

4. Substitute the values for B through D from the previous page into the following formula: The “label queue” memory required (in BYTES)(B x C x D)/8 =	(E) _____
5. Substitute the values for A and E into the following formula: The ^CO command memory required (in KBYTES) (A-E)/1024)-5=	(F) _____

NOTE: If the value of (F) is less than zero, then no ^CO command is needed. If the value of (F) is greater than zero, use the integer portion in the ^CO command.

Available RAM (A)	= 71478 BYTES
Print Width (B)	= 832 DOTS
Label Length (C)	= 1000 DOTS
Max Labels Queued (D)	= 2
Then —	
The label queue memory required (E) =(B x C x D)/8 = 208000 BYTES	
And —	
The ^CO command memory required (F) = (71478-208000)/1024)-5=489.87 KBYTES	
Therefore, the correct ^C0 command string to add to the label format would be:	
^XA^COY,489^XZ	

This command string will cause 489 KBYTES to be set aside as Font Memory and make it unavailable as label format memory. The memory remaining will only allow two labels to be formatted at one time, and the time and date will be more precise for those two labels.

ZPL II Samples

The ZPL II scripts shown on this page establish the initial settings for the date and time clock. The script below then references these settings to provide the output shown in Figure 26 Printed Result of the Above ZPL II Script on page 1597.

Setting the date and time for the Real Time Clock only needs to be done once. The date and time are maintained by an on-board battery when the printer is reset or the printer is turned Off (O).

To set the date and time to April 23, 2005 at 2:30pm, the following command string should be sent to the printer:

```
^XA
^ST04,23,2005,02,30,0,P^FS
^XZ
```

To initialize the Real Time Clock and set up two offset values (offset #2 set to 3 months and 1 hour in the future, offset #3 set to 1 year in the past), the following command sequence should be sent to the printer:

```
^XA
^SL
^SO2,3,0,0,1,0,0^FS
^SO3,0,0,-1,0,0,0^FS
^XZ
```

The above ZPL II scripts initialize the RTC date and time and must be sent to a printer to provide proper date and time parameters for the ZPL II script below.

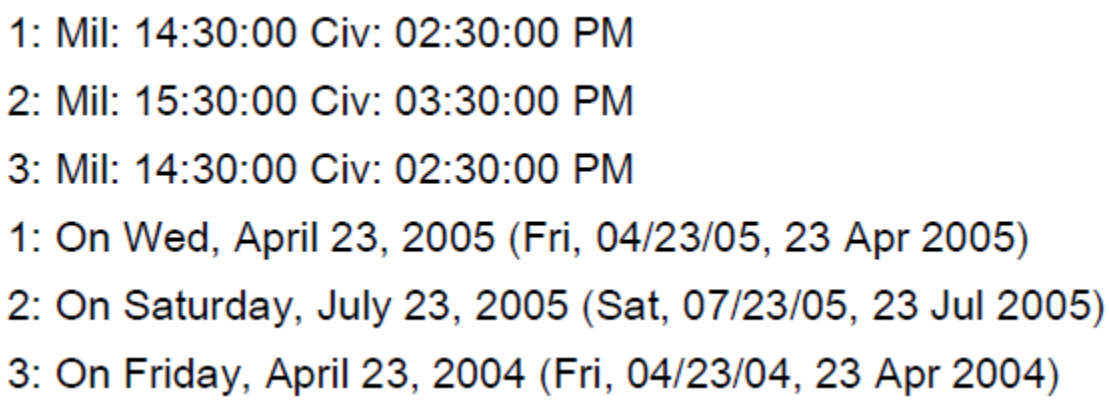
The following ZPL II script illustrates the various methods of printing the date and time initialized in the script above within separate fields on continuous media. Figure 26 Printed Result of the Above ZPL II Script on page 1597illustrates the printout of this script on a label.

For the below example, the ^FC command delimiters are:

% Primary clock indicator { Secondary clock indicator # Third clock indicator

```
^XA
^LL175
^FO10,025^AD^FC%,{,#^FD1: Mil: %H:%M:%S Civ: %I:%M:%S %p^FS
^FO10,050^AD^FC%,{,#^FD2: Mil: {H:{M:{S Civ: {I:{M:{S {p^FS
^FO10,075^AD^FC%,{,#^FD3: Mil: #H:#M:#S Civ: #I:#M:#S #p^FS
^FO10,100^AD^FC%,{,#^FD1: On %A, %B %d
, %Y (%a, %m/%d/%y, %d %b %Y).^FS
^FO10,125^AD^FC%,{,#^FD2: On {A, {B {d, {Y (
 {a, {m/{d/{y, {d {b {Y).^FS
^FO10,150^AD^FC%,{,#^FD3: On #A, #B #d, #Y (
#a, #m/#d/#y, #d #b #Y).^FS
^XZ
```

Figure 26 Printed Result of the Above ZPL II Script



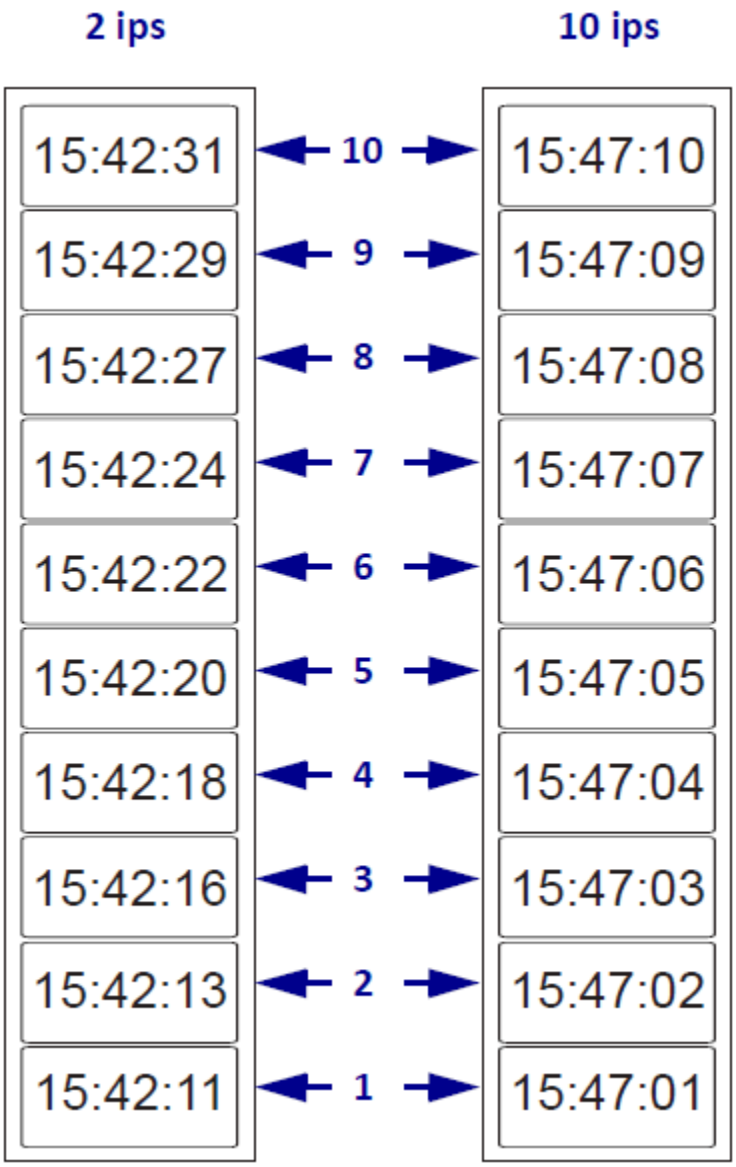
The following are examples of the time stamp using the ^SL1 and ^SL5 at 2 ips and 10 ips for the Enhanced Real Time Clock (V60.13.0.10 and later).

NOTE: They show the variation of time due to print speed and label complexity.

```
^XA
^SL1^FS
```

```
^FO187,184^A0N,101,121^FC%^FD%H:%M:%S^FS
^PQ10
^XZ
```

Figure 27 Example of ^SL1, 2 ips and 10 ips

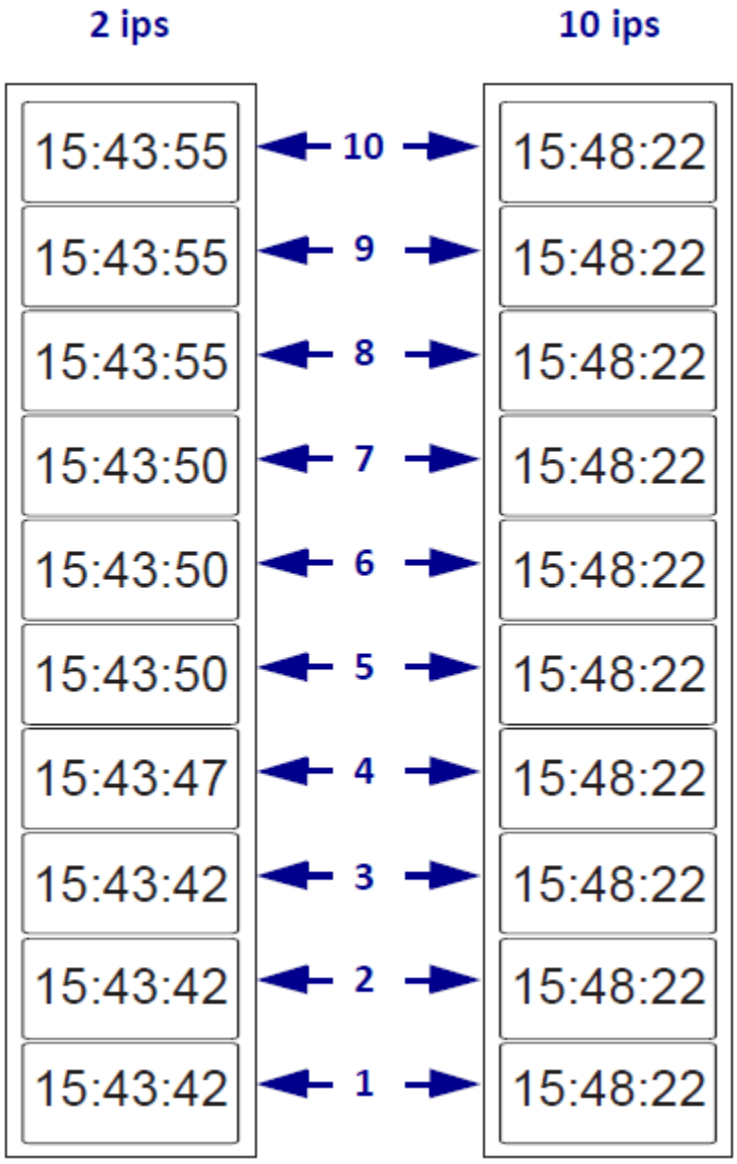


1	Label 1		6	Label 6
2	Label 2		7	Label 7
3	Label 3		8	Label 8
4	Label 4		9	Label 9
5	Label 5		10	Label 10

```
^XA
```

```
^SL5^FS
^FO187,184^A0N,101,121^FC%^FD%H:%M:%S^FS
^PQ10
^XZ
```

Figure 28 Example of ^SL5, 2 ips and 10 ips



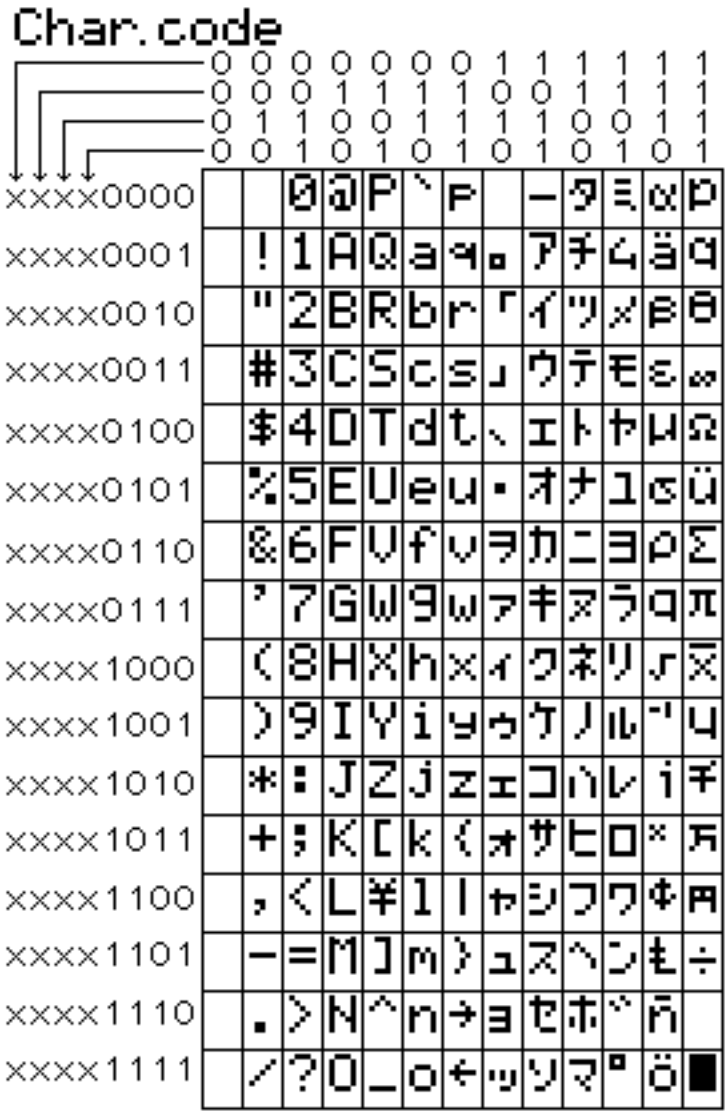
1	Label 1		6	Label 6
2	Label 2		7	Label 7
3	Label 3		8	Label 8
4	Label 4		9	Label 9
5	Label 5		10	Label 10

ZBI Character Set

This section provides you with the set of characters that are supported on the front panel of the following Zebra printers with ZBI 2.0: XiIIIPlus, 105SL, Z4M/Z6M, ZM400/ZM600, RZ400/RZ600, PAX4 and S4M when V60.16.0Z or V53.16.0Z, or later firmware are loaded. These characters can be used in ZBI programs to display content on the LCD.

These are the supported characters for these Zebra printers:

- XiIIIPlus
- 105SL
- Z4M/Z6M
- PAX4
- S4M



These are the supported characters for these Zebra printers:

- ZM400/ZM600
- RZ400/RZ600
- Xi4
- RXi4

space 0020	" 0022	% 0025	& 0026	(0028	) 0029	+ 002B	- 002D	. 002E	/ 002F
0 0030	1 0031	2 0032	3 0033	4 0034	6 0035	7 0037	8 0038	9 0039	: 003A
>003E	? 003F	A 0041	B 0042	C 0043	D 0044	E 0045	F 0046	G 0047	H 0048
I 0049	J 004A	K 004B	L 004C	M 004D	N 004E	O 004F	P 0050	Q 0051	R 0052
S 0053	T 0054	U 0055	V 0056	W 0057	X 0058	Y 0059	Z 005A	_ 005F	

SGD Command Support

This appendix provides you with details identifying which SGD commands can be used with different printers.

SGDs Supported for Industrial Printers

This table provides details of the list of SGDs supported for Industrial Printers.

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
alerts.add				X	X	X	X
alerts.conditions				X	X	X	X
alerts.configured				X	X	X	X
alerts.destinations				X	X	X	X
alerts.http.authentication.add				X	X	X	X
alerts.http.authentication.entries				X	X	X	X
alerts.http.authentication.remove				X	X	X	X
alerts.http.logging.clear				X	X	X	X
alerts.http.logging.entries				X	X	X	X
alerts.http.logging.max_entries				X	X	X	X
alerts.http.proxy				X	X	X	X
alerts.tracked_settings.clear_log				X	X	X	X
alerts.tracked_settings.log_tracked				X	X	X	X
alerts.tracked_settings.max_log_entries				X	X	X	X
alerts.tracked_settings.zbi_notified				X	X	X	X
apl.enable				X	X	X	X
apl.framework_version				X	X	X	X
apl.version				X	X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
appl.bootblock	X	X	X	X	X	X	X
appl.date	X	X	X	X	X	X	X
appl.link_os_version				X	X	X	X
appl.name	X	X	X	X	X	X	X
bluetooth.address					X	X	X
bluetooth.authentication							
bluetooth.bluetooth_pin					X	X	X
bluetooth.clear_bonding_cache					X	X	X
bluetooth.discoverable				X	X	X	X
bluetooth.enable					X	X	X
bluetooth.enable_reconnect					X	X	X
bluetooth.friendly_name				X	X	X	X
bluetooth.json_config_channel_enable					X	X	X
bluetooth.le.controller_mode						X	X
bluetooth.le.minimum_security						X	X
bluetooth.le.power_class						X	X
bluetooth.local_name					X	X	X
bluetooth.minimum_security_mode					X	X	X
bluetooth.page_scan_window					X	X	X
bluetooth.radio_version					X	X	X
bluetooth.short_address					X	X	X
bluetooth.version					X	X	X
capture.channel1.count				X	X	X	X
capture.channel1.data.mime				X	X	X	X
capture.channel1.data.raw				X	X	X	X
capture.channel1.delimiter				X	X	X	X
capture.channel1.max_length				X	X	X	X
capture.channel1.port				X	X	X	X
comm.baud	X	X	X	X	X	X	X
comm.halt							
comm.parity	X	X	X	X	X	X	X
comm.stop_bits	X	X		X	X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
comm.type							
cradle.comm.baud							
cradle.comm.handshake							
cutter.clean_cutter	X	X	X	X	X	X	X
device.allow_firmware_downloads				X	X	X	X
device.applicator.data_ready							X
device.applicator.data_ready_activation							X
device.applicator.end_print							X
device.applicator.error_on_pause							X
device.applicator.feed							X
device.applicator.media_out						X	X
device.applicator.pause						X	X
device.applicator.reprint						X	X
device.applicator.rfid_void						X	X
device.applicator.ribbon_low						X	X
device.applicator.ribbon_out						X	X
device.applicator.service_required						X	X
device.applicator.start_print							X
device.applicator.start_print_mode							X
device.applicator.voltage							X
device.bluetooth_installed					X	X	X
device.command_override.active				X	X	X	X
device.command_override.add				X	X	X	X
device.command_override.clear				X	X	X	X
device.command_override.list				X	X	X	X
device.company_contact				X	X	X	X
device.configuration_number							X
device.cpcl_synchronous_mode				X	X	X	X
device.cutter_installed				X	X	X	X
device.download_connection_timeout	X	X	X	X	X	X	X
device.download_interactive_mode							
device.epl_legacy_mode				X	X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
device.feature.802_11ac				X	X	X	X
device.feature.bluetooth_le					X	X	X
device.feature.head_element_test					X	X	X
device.feature.lighted_arrows				X	X	X	X
device.feature.mcr				X	X	X	X
device.feature.nfc				X	X	X	X
device.feature.ribbon_cartridge				X	X		X
device.friendly_name	X	X	X	X	X	X	X
device.frontpanel.key_press				X	X	X	X
device.frontpanel.line1	X	X	X	X	X	X	X
device.frontpanel.line2	X	X	X	X	X	X	X
device.frontpanel.wml_menus	X	X	X				
device.frontpanel.xml	X	X	X	X	X	X	X
device.host_identification				X	X	X	X
device.host_status				X	X	X	X
device.idle_display_format				X	X	X	X
device.idle_display_value				X	X	X	X
device.internal_wired_setting_location				X	X	X	X
device.job_log.total_jobs				X	X	X	X
device.jobs_print	X	X	X	X	X	X	X
device.languages	X	X	X	X	X	X	X
device.light.cover_open_brightness					X	X	X
device.light.head_open_brightness					X		
device.light.media_path_brightness						X	X
device.light.ribbon_path_brightness						X	X
device.ltu_installed				X	X	X	X
device.pause				X	X	X	X
device.pmcu.revision						X	X
device.pnp_option	X	X	X	X	X	X	X
device.position.accuracy				X	X	X	X
device.position.altitude				X	X	X	X
device.position.latitude				X	X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
device.position.longitude				X	X	X	X
device.print_2key							
device.print_reprogram_2key				X	X	X	X
device.printhead.odometer						X	X
device.printhead.resolution				X	X	X	X
device.printhead.test.summary							
device.product_name_submodel				X	X	X	X
device.reset				X	X	X	X
device.restore_defaults				X	X	X	X
device.rewinder_installed				X	X	X	X
device.save_2key							
device.sensor_profile				X	X	X	X
device.sensor_select				X	X	X	X
device.serial_numbers.applicator_option_board						X	X
device.serial_numbers.applicator_option_board_date							X
device.serial_numbers.control_panel_date				X	X	X	X
device.serial_numbers.cutter				X	X	X	X
device.serial_numbers.cutter_date				X	X	X	X
device.serial_numbers.mlb_date				X	X	X	X
device.serial_numbers.parallel_option_board						X	X
device.serial_numbers.parallel_option_board_date							X
device.serial_numbers.printhead				X	X	X	X
device.serial_numbers.printhead_date				X	X	X	X
device.serial_numbers.usb_host_option_board						X	X
device.serial_numbers.usb_host_option_board_date						X	X
device.serial_numbers.wired_ethernet_option_board				X	X	X	X
device.serial_numbers.wired_ethernet_option_board_ date							X
device.set_clock_to_build_date				X	X		X
device.slot_1				X	X	X	X
device.slot_2					X	X	X
device.super_host_status				X	X	X	X
device.syslog.clear_log				X	X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
device.syslog.configuration				X	X	X	X
device.syslog.enable				X	X	X	X
device.syslog.entries				X	X	X	X
device.syslog.log_max_file_size				X	X	X	X
device.syslog.save_local_file				X	X	X	X
device.unique_id	X	X	X	X	X	X	X
device.unpause				X	X	X	X
device.uptime	X	X	X	X	X	X	X
device.user_p1	X	X	X	X	X	X	X
device.user_p2	X	X	X	X	X	X	X
device.user_vars.create				X	X	X	X
device.user_vars.set_range				X	X	X	X
device.xml.enable	X	X	X	X	X	X	X
display.backlight				X	X	X	X
display.backlight_on_time				X	X	X	X
display.batch_counter				X	X	X	X
display.bluetooth.mac							
display.language				X	X	X	X
display.load_card				X	X	X	X
display.password.level				X	X	X	X
display.root_wml				X	X	X	X
display.status_bar_suppress				X	X	X	X
display.suppress_all_alerts				X	X	X	X
display.text	X	X	X		X	X	X
external_wired.check	X	X	X				
external_wired.ip.addr	X	X	X				
external_wired.ip.arp_interval	X	X	X				
external_wired.ip.default_addr_enable	X	X	X				
external_wired.ip.gateway	X	X	X				
external_wired.ip.netmask	X	X	X				
external_wired.ip.port	X	X	X				
external_wired.ip.protocol	X	X	X				

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
external_wired.ip.timeout.enable	X	X	X				
external_wired.ip.timeout.value	X	X	X				
external_wired.ip.v6.addr	X	X	X				
external_wired.ip.v6.gateway	X	X	X				
external_wired.ip.v6.prefix_length	X	X	X				
external_wired.mac_addr	X	X	X				
external_wired.mac_raw	X	X	X	X	X	X	X
ezpl.head_close_action				X	X	X	X
ezpl.label_length_max				X	X	X	X
ezpl.label_sensor				X	X	X	X
ezpl.manual_calibration				X	X	X	X
ezpl.media_type				X	X	X	X
ezpl.power_up_action				X	X	X	X
ezpl.print_method				X	X	X	X
ezpl.print_width				X	X	X	X
ezpl.reprint_mode				X	X	X	X
ezpl.restore_defaults				X	X	X	X
ezpl.take_label				X	X	X	X
ezpl.take_label_calibration					X		
ezpl.tear_off				X	X	X	X
file.capture_response.destination				X	X	X	X
file.cert.expiration				X	X	X	X
file.cert.supported_curves				X	X	X	X
file.delete				X	X	X	X
file.dir	X	X	X	X	X	X	X
file.dir_format				X	X	X	X
file.run				X	X	X	X
file.type				X	X	X	X
formats.cancel_all				X	X	X	X
head.authenticated						X	X
head.darkness_switch							
head.darkness_switch_enable							

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
head.element_test							
head.latch	X	X	X	X	X	X	X
head.resolution.in_dpi				X	X	X	X
input.capture				X	X	X	X
interface.network.active.arp_interval				X	X	X	X
interface.network.active.cable_type				X	X	X	X
interface.network.active.dhcp_received_host_name				X	X	X	X
interface.network.active.gateway	X	X	X	X	X	X	X
interface.network.active.ip_addr	X	X	X	X	X	X	X
interface.network.active.mac_addr	X	X	X	X	X	X	X
interface.network.active.mac_raw	X	X	X	X	X	X	X
interface.network.active.netmask	X	X	X	X	X	X	X
interface.network.active.printserver	X	X	X	X	X	X	X
interface.network.active.protocol	X	X	X	X	X	X	X
interface.network.active.protocol_error				X	X	X	X
interface.network.active.rx_errors				X	X	X	X
interface.network.active.rx_packets				X	X	X	X
interface.network.active.server_address				X	X	X	X
interface.network.active.speed				X	X	X	X
interface.network.active.tx_errors				X	X	X	X
interface.network.active.tx_packets				X	X	X	X
interface.network.active.wins_addr				X	X	X	X
interface.network.settings_require_reset				X	X	X	X
internal_wired.8021x.password				X	X		X
internal_wired.8021x.peap.anonymous_identity				X	X	X	X
internal_wired.8021x.peap.validate_server_certificate				X	X	X	X
internal_wired.8021x.security				X	X		X
internal_wired.8021x.ttls_tunnel				X	X	X	X
internal_wired.8021x.username				X	X		X
internal_wired.activity_led				X	X	X	X
internal_wired.auto_switchover	X	X	X	X	X	X	X
internal_wired.enable				X	X		X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
internal_wired.installed				X	X	X	X
internal_wired.ip.addr	X	X	X	X	X	X	X
internal_wired.ip.arp_interval	X	X	X	X	X	X	X
internal_wired.ip.default_addr_enable	X	X	X	X	X	X	X
internal_wired.ip.dhcp.arp_verify				X	X	X	X
internal_wired.ip.dhcp.cache_ip	X	X	X	X	X	X	X
internal_wired.ip.dhcp.cid_all	X	X	X	X	X	X	X
internal_wired.ip.dhcp.cid_enable	X	X	X	X	X	X	X
internal_wired.ip.dhcp.cid_prefix	X	X	X	X	X	X	X
internal_wired.ip.dhcp.cid_suffix	X	X	X	X	X	X	X
internal_wired.ip.dhcp.cid_type	X	X	X	X	X	X	X
internal_wired.ip.dhcp.lease.last_attempt	X	X	X	X	X	X	X
internal_wired.ip.dhcp.lease.length	X	X	X	X	X	X	X
internal_wired.ip.dhcp.lease.server	X	X	X	X	X	X	X
internal_wired.ip.dhcp.lease.time_left	X	X	X	X	X	X	X
internal_wired.ip.dhcp.option12	X	X	X	X	X	X	X
internal_wired.ip.dhcp.option12_format	X	X	X	X	X	X	X
internal_wired.ip.dhcp.option12_value	X	X	X	X	X	X	X
internal_wired.ip.dhcp.requests_per_session	X	X	X	X	X	X	X
internal_wired.ip.dns.servers				X	X	X	X
internal_wired.ip.gateway	X	X	X	X	X	X	X
internal_wired.ip.netmask	X	X	X	X	X	X	X
internal_wired.ip.port	X	X	X	X	X	X	X
internal_wired.ip.port_alternate				X	X	X	X
internal_wired.ip.port_json_config				X	X	X	X
internal_wired.ip.protocol	X	X	X	X	X	X	X
internal_wired.ip.timeout.enable	X	X	X	X	X	X	X
internal_wired.ip.timeout.value	X	X	X	X	X	X	X
internal_wired.ip.wins.addr	X	X	X	X	X	X	X
internal_wired.ip.wins.permanent_source	X	X	X	X	X	X	X
internal_wired.mac_addr	X	X	X	X	X	X	X
internal_wired.mac_raw	X	X	X	X	X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
ip.active_network	X	X	X	X	X	X	X
ip.addr	X	X	X	X	X	X	X
ip.arp_interval	X	X	X	X	X	X	X
ip.bootp.enable	X	X	X	X	X	X	X
ip.dhcp.arp_verify				X	X	X	X
ip.dhcp.auto_provision_enable				X	X	X	X
ip.dhcp.cache_ip	X	X	X	X	X	X	X
ip.dhcp.cid_all	X	X	X	X	X	X	X
ip.dhcp.cid_enable	X	X	X	X	X	X	X
ip.dhcp.cid_prefix	X	X	X	X	X	X	X
ip.dhcp.cid_suffix	X	X	X	X	X	X	X
ip.dhcp.cid_type	X	X	X	X	X	X	X
ip.dhcp.cid_value	X	X	X	X	X	X	X
ip.dhcp.enable	X	X	X	X	X	X	X
ip.dhcp.lease.last_attempt	X	X	X	X	X	X	X
ip.dhcp.lease.server	X	X	X	X	X	X	X
ip.dhcp.lease.time_left	X	X	X	X	X	X	X
ip.dhcp.ntp.enable				X	X	X	X
ip.dhcp.ntp.received_servers				X	X	X	X
ip.dhcp.option12	X	X	X	X	X	X	X
ip.dhcp.option12_format	X	X	X	X	X	X	X
ip.dhcp.option12_value	X	X	X	X	X	X	X
ip.dhcp.request_timeout	X	X	X	X	X	X	X
ip.dhcp.requests_per_session	X	X	X	X	X	X	X
ip.dhcp.session_interval	X	X	X	X	X	X	X
ip.dhcp.user_class_id				X	X	X	X
ip.dhcp.vendor_class_id				X	X	X	X
ip.dns.servers	X	X	X	X	X	X	X
ip.firewall.whitelist_in				X	X	X	X
ip.ftp.enable	X	X	X	X	X	X	X
ip.ftp.execute_file	X	X	X	X	X	X	X
ip.ftp.request_password				X	X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
ip.gateway	X	X	X	X	X	X	X
ip.http.admin_name				X	X	X	X
ip.http.admin_password				X	X	X	X
ip.http.custom_link_name				X	X	X	X
ip.http.custom_link_url				X	X	X	X
ip.http.enable	X	X	X	X	X	X	X
ip.http.faq_url				X	X	X	X
ip.http.port				X	X	X	X
ip.https.enable				X	X	X	X
ip.https.port				X	X	X	X
ip.lpd.enable	X	X	X	X	X	X	X
ip.mirror.appl_path				X	X	X	X
ip.mirror.auto	X	X	X	X	X	X	X
ip.mirror.error_retry	X	X	X	X	X	X	X
ip.mirror.feedback.auto	X	X	X	X	X	X	X
ip.mirror.feedback.freq	X	X	X	X	X	X	X
ip.mirror.feedback.odometer	X	X	X	X	X	X	X
ip.mirror.feedback.path	X	X	X	X	X	X	X
ip.mirror.fetch				X	X	X	X
ip.mirror.freq	X	X	X	X	X	X	X
ip.mirror.freq_hours	X	X	X	X	X	X	X
ip.mirror.interface				X	X	X	X
ip.mirror.last_error	X	X	X	X	X	X	X
ip.mirror.last_time	X	X	X	X	X	X	X
ip.mirror.mode				X	X	X	X
ip.mirror.password	X	X	X	X	X	X	X
ip.mirror.path	X	X	X	X	X	X	X
ip.mirror.reset_delay				X	X	X	X
ip.mirror.server	X	X	X	X	X	X	X
ip.mirror.success	X	X	X	X	X	X	X
ip.mirror.success_time	X	X	X	X	X	X	X
ip.mirror.username	X	X	X	X	X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
ip.mirror.version	X	X	X	X	X	X	X
ip.netmask	X	X	X	X	X	X	X
ip.ntp.enable				X	X	X	X
ip.ntp.log				X	X	X	X
ip.ntp.servers				X	X	X	X
ip.ping_gateway_interval				X	X	X	X
ip.ping_remote				X	X	X	X
ip.pop3.enable	X	X	X	X	X	X	X
ip.pop3.password	X	X	X	X	X	X	X
ip.pop3.poll	X	X	X	X	X	X	X
ip.pop3.print_body				X	X	X	X
ip.pop3.print_headers				X	X	X	X
ip.pop3.save_attachments				X	X	X	X
ip.pop3.server_addr	X	X	X	X	X	X	X
ip.pop3.username	X	X	X	X	X	X	X
ip.pop3.verbose_headers				X	X	X	X
ip.port	X	X	X	X	X	X	X
ip.port_alternate				X	X	X	X
ip.port_json_config				X	X	X	X
ip.port_single_conn				X	X	X	X
ip.port_single_conn_idle_timeout				X	X	X	X
ip.primary_network	X	X	X	X	X	X	X
ip.smtp.domain	X	X	X	X	X	X	X
ip.smtp.enable	X	X	X	X	X	X	X
ip.smtp.server_addr	X	X	X	X	X	X	X
ip.snmp.enable	X	X	X	X	X	X	X
ip.snmp.get_community_name				X	X	X	X
ip.snmp.set_community_name				X	X	X	X
ip.snmp.trap_community_name				X	X	X	X
ip.tcp.enable	X	X	X	X	X	X	X
ip.tcp.nagle_algorithm				X	X	X	X
ip.telnet.enable	X	X	X	X	X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
ip.tls.enable				X	X	X	X
ip.tls.port				X	X	X	X
ip.tls.port_json_config				X	X	X	X
ip.udp.enable	X	X	X	X	X	X	X
log.reboot.code				X	X	X	X
log.reboot.codes				X	X	X	X
log.reboot.reason				X	X	X	X
log.reboot.report				X	X	X	X
mcr.cancel							
mcr.crypt.algorithm							
mcr.crypt.enabled							
mcr.crypt.key_mgmt							
mcr.out							
mcr.query							
mcr.revision							
media.bar_location				X	X	X	X
media.cartridge.darkness							
media.cartridge.inserted							
media.cartridge.labels_remaining							
media.cartridge.length							
media.cartridge.part_number							
media.cartridge.serial_number							
media.cartridge.speed							
media.cartridge.total_label_cnt							
media.cartridge.width							
media.cut_now				X	X	X	X
media.darkness_mode	X	X	X				
media.draft_mode							
media.dynamic_length_calibration				X	X	X	X
media.extended_presentation					X		
media.feed_skip							
media.linerless_offset				X	X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
media.media_low.external							
media.media_low.warning	X	X	X				
media.part_number				X	X	X	X
media.present.cut_amount							
media.present.cut_margin							
media.present.eject							
media.present.length_addition							
media.present.loop_length							
media.present.loop_length_max							
media.present.present_timeout							
media.present.present_type							
media.printmode	X	X	X	X	X	X	X
media.serial_number				X	X	X	X
media.speed	X	X	X	X	X	X	X
memory.flash_free	X	X	X	X	X	X	X
memory.flash_size	X	X	X	X	X	X	X
memory.ram_free	X	X	X	X	X	X	X
memory.ram_size	X	X	X	X	X	X	X
memory.types	X	X	X				
netmanage.avalanche.agent_addr				X	X	X	X
netmanage.avalanche.available_agent				X	X	X	X
netmanage.avalanche.available_port				X	X	X	X
netmanage.avalanche.encryption_type				X	X	X	X
netmanage.avalanche.interval				X	X	X	X
netmanage.avalanche.interval_update				X	X	X	X
netmanage.avalanche.model_name				X	X	X	X
netmanage.avalanche.set_property				X	X	X	X
netmanage.avalanche.startup_update				X	X	X	X
netmanage.avalanche.tcp_connection_timeout				X	X	X	X
netmanage.avalanche.terminal_id				X	X	X	X
netmanage.avalanche.text_msg.beep				X	X	X	X
netmanage.avalanche.text_msg.display				X	X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
netmanage.avalanche.text_msg.print				X	X	X	X
netmanage.avalanche.udp_timeout				X	X	X	X
netmanage.error_code	X	X	X	X	X	X	X
netmanage.state_code	X	X	X	X	X	X	X
netmanage.status_code	X	X	X	X	X	X	X
odometer.headclean				X	X	X	X
odometer.headnew				X	X	X	X
odometer.label_dot_length	X	X	X	X	X	X	X
odometer.latch_open_count				X	X	X	X
odometer.media_marker_count	X	X	X	X	X	X	X
odometer.media_marker_count1	X	X	X	X	X	X	X
odometer.media_marker_count2	X	X	X	X	X	X	X
odometer.retracts_count							
odometer.rfid.valid_resettable					X	X	X
odometer.rfid.void_resettable					X	X	X
odometer.total_cuts				X	X	X	X
odometer.total_label_count				X	X	X	X
odometer.total_print_length	X	X	X	X	X	X	X
odometer.user_label_count				X	X	X	X
odometer.user_label_count1				X	X	X	X
odometer.user_label_count2				X	X	X	X
odometer.user_total_cuts				X	X	X	X
power.average_current				X	X		
power.battery_led_blink_rate							
power.battery_led_enable							
power.battery_led_off_duration							
power.battery_led_on_duration							
power.battery_replacement_cyclecount_threshold							
power.battery_type							
power.current							
power.cycle_count							
power.dtr_power_off				X	X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
power.energy_star.enable						X	X
power.energy_star.timeout						X	X
power.label_queue.shutdown							
power.part_number							
power.percent_health							
power.power_on_mode							
power.remaining_capacity							
power.sleep.cradle							
power.temperature							
power.voltage							
power.wake.radio							
print.legacy_compatibility							X
print.tone	X	X	X	X	X	X	X
print.troubleshooting_label_print						X	X
rfid.adaptive_antenna					X	X	X
rfid.antenna_sweep					X	X	X
rfid.country_code					X	X	X
rfid.enable					X	X	X
rfid.error.response					X	X	X
rfid.hop_table_version					X	X	X
rfid.log.clear					X		
rfid.log.enabled					X		
rfid.log.entries					X		
rfid.position.program					X	X	X
rfid.reader_1.antenna_port					X	X	X
rfid.reader_1.firmware_version					X	X	X
rfid.reader_1.hardware_version					X	X	X
rfid.reader_1.model					X	X	X
rfid.reader_1.power.read					X	X	X
rfid.reader_1.power.write					X	X	X
rfid.recipe_version					X	X	X
rfid.region_code					X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
rfid.tag.calibrate					X	X	X
rfid.tag.read.content					X	X	X
rfid.tag.read.execute					X	X	X
rfid.tag.read.result_line1					X	X	X
rfid.tag.read.result_line1_alternate					X	X	X
rfid.tag.read.result_line2					X	X	X
rfid.tag.read.result_line2_alternate					X	X	X
rfid.tag.result_line1							
rfid.tag.test.content					X	X	X
rfid.tag.test.execute					X	X	X
rfid.tag.test.result_line1					X	X	X
rfid.tag.test.result_line2					X	X	X
ribbon.cartridge.authenticated							
ribbon.cartridge.inserted							
ribbon.cartridge.length							
ribbon.cartridge.length_remaining							
ribbon.cartridge.part_number							
rtc.timezone				X	X	X	X
rtc.unix_timestamp				X			
sensor.back_bar.brightness						X	X
sensor.back_bar.cur				X	X	X	X
sensor.back_bar.gain						X	X
sensor.back_bar.offset					X	X	X
sensor.back_bar.ppr_out_thold				X	X	X	X
sensor.back_bar.thold				X	X	X	X
sensor.battery.in_volts							
sensor.cover_open					X	X	X
sensor.front_bar.brightness				X	X		
sensor.front_bar.cur				X	X		
sensor.front_bar.gain				X	X		
sensor.front_bar.offset				X	X		
sensor.front_bar.ppr_out_thold				X	X		

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
sensor.front_bar.thold				X	X		
sensor.gap.brightness							
sensor.gap.cur				X	X	X	X
sensor.gap.gain							
sensor.gap.offset							
sensor.gap.thold				X	X	X	X
sensor.head.temp				X	X	X	X
sensor.head.temp_avg				X	X	X	X
sensor.head.temp_celsius				X	X	X	X
sensor.peel.brightness							
sensor.peel.cur				X	X	X	X
sensor.peel.gain							
sensor.peel.thold				X	X		
sensor.peeler				X	X	X	X
sensor.width.cur							
sensor.width.in_dots							
usb.device.device_id_string				X	X	X	X
usb.device.device_unique_id				X	X	X	X
usb.device.device_version				X	X	X	X
usb.device.manufacturer_string				X	X	X	X
usb.device.product_id				X	X	X	X
usb.device.product_string				X	X	X	X
usb.device.serial_string				X	X	X	X
usb.device.vendor_id				X	X	X	X
usb.halt							
usb.host.config_info_to_usb						X	X
usb.host.fn_field_data					X	X	X
usb.host.fn_last_field					X	X	X
usb.host.hid_count					X	X	X
usb.host.keyboard_input					X	X	X
usb.host.lock_out					X	X	X
usb.host.mass_storage_count					X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
usb.host.read_list					X	X	X
usb.host.read_list_print_delay					X	X	X
usb.host.template_list					X	X	X
usb.host.template_print_amount					X	X	X
usb.host.write_list					X	X	X
usb.mirror.appl_path					X	X	X
usb.mirror.auto					X	X	X
usb.mirror.enable					X	X	X
usb.mirror.error_retry					X	X	X
usb.mirror.feedback.auto					X	X	X
usb.mirror.feedback.odometer					X	X	X
usb.mirror.feedback.path					X	X	X
usb.mirror.fetch					X	X	X
usb.mirror.last_error					X	X	X
usb.mirror.last_time					X	X	X
usb.mirror.path					X	X	X
usb.mirror.reset_delay					X	X	X
usb.mirror.success					X	X	X
usb.mirror.success_time					X	X	X
weblink.cloud_connect.enable				X	X	X	X
weblink.enable				X	X	X	X
weblink.ip.conn1.authentication.add				X	X	X	X
weblink.ip.conn1.authentication.entries				X	X	X	X
weblink.ip.conn1.authentication.remove				X	X	X	X
weblink.ip.conn1.location				X	X	X	X
weblink.ip.conn1.maximum_simultaneous_connections				X	X	X	X
weblink.ip.conn1.num_connections				X	X	X	X
weblink.ip.conn1.proxy				X	X	X	X
weblink.ip.conn1.retry_interval				X	X	X	X
weblink.ip.conn1.retry_interval_random_max					X		X
weblink.ip.conn1.test.location				X	X	X	X
weblink.ip.conn1.test.retry_interval				X	X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
weblink.ip.conn1.test.test_on				X	X	X	X
weblink.ip.conn2.authentication.add				X	X	X	X
weblink.ip.conn2.authentication.entries				X	X	X	X
weblink.ip.conn2.authentication.remove				X	X	X	X
weblink.ip.conn2.location				X	X	X	X
weblink.ip.conn2.maximum_simultaneous_connections				X	X	X	X
weblink.ip.conn2.num_connections				X	X	X	X
weblink.ip.conn2.proxy				X	X	X	X
weblink.ip.conn2.retry_interval				X	X	X	X
weblink.ip.conn2.retry_interval_random_max					X		X
weblink.ip.conn2.test.location				X	X	X	X
weblink.ip.conn2.test.retry_interval				X	X	X	X
weblink.ip.conn2.test.test_on				X	X	X	X
weblink.logging.clear				X	X	X	X
weblink.logging.entries				X	X	X	X
weblink.logging.max_entries				X	X	X	X
weblink.printer_reset_required				X	X	X	X
weblink.restore_defaults				X	X	X	X
weblink.zebra_connector.authentication.add				X	X	X	X
weblink.zebra_connector.authentication.entries				X	X	X	X
weblink.zebra_connector.authentication.remove				X	X	X	X
weblink.zebra_connector.enable				X	X	X	X
weblink.zebra_connector.proxy				X	X	X	X
weblink.zebra_connector.version				X	X	X	X
wlan.11ac.80mhz_enable							X
wlan.11d.enable				X	X	X	X
wlan.11n.20mhz_only				X	X	X	X
wlan.11n.greenfield				X	X	X	X
wlan.11n.rifs				X	X	X	X
wlan.11n.short_gi_20mhz				X	X	X	X
wlan.11n.short_gi_40mhz				X	X	X	X
wlan.8021x.authentication	X	X	X	X	X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
wlan.8021x.eap.password				X	X	X	X
wlan.8021x.eap.privkey_password				X	X	X	X
wlan.8021x.eap.username	X	X	X	X	X	X	X
wlan.8021x.enable	X	X	X	X	X	X	X
wlan.8021x.peap.anonymous_identity				X	X		X
wlan.8021x.peap.peap_password				X	X	X	X
wlan.8021x.peap.peap_username	X	X	X	X	X	X	X
wlan.8021x.peap.privkey_password				X	X	X	X
wlan.8021x.ttls_tunnel				X	X	X	X
wlan.active_channels				X	X	X	X
wlan.adhoc_last_channel							
wlan.adhocautomode	X	X	X	X	X	X	X
wlan.adhocchannel	X	X	X	X	X	X	X
wlan.allowed_band				X	X	X	X
wlan.associated	X	X	X	X	X	X	X
wlan.auth_type	X	X	X	X	X	X	X
wlan.authenticated				X	X	X	X
wlan.authentication_error				X	X	X	X
wlan.available				X	X	X	X
wlan.band_preference				X	X	X	X
wlan.bssid	X	X	X	X	X	X	X
wlan.channel	X	X	X	X	X	X	X
wlan.channel_mask				X	X	X	X
wlan.country_code				X	X	X	X
wlan.current_tx_rate	X	X	X	X	X	X	X
wlan.enable				X	X	X	X
wlan.encryption_index	X	X	X				
wlan.encryption_key1							
wlan.encryption_key2							
wlan.encryption_key3							
wlan.encryption_key4							
wlan.encryption_mode	X	X	X				

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
wlan.essid	X	X	X	X	X	X	X
wlan.firmware_version	X	X	X	X	X	X	X
wlan.ip.addr	X	X	X	X	X	X	X
wlan.ip.arp_interval	X	X	X	X	X	X	X
wlan.ip.default_addr_enable	X	X	X	X	X	X	X
wlan.ip.dhcp.arp_verify				X	X	X	X
wlan.ip.dhcp.cache_ip	X	X	X	X	X	X	X
wlan.ip.dhcp.cid_all				X	X	X	X
wlan.ip.dhcp.cid_enable	X	X	X	X	X	X	X
wlan.ip.dhcp.cid_prefix				X	X	X	X
wlan.ip.dhcp.cid_suffix				X	X	X	X
wlan.ip.dhcp.cid_type	X	X	X	X	X	X	X
wlan.ip.dhcp.lease.last_attempt	X	X	X	X	X	X	X
wlan.ip.dhcp.lease.length	X	X	X	X	X	X	X
wlan.ip.dhcp.lease.server	X	X	X	X	X	X	X
wlan.ip.dhcp.lease.time_left	X	X	X	X	X	X	X
wlan.ip.dhcp.option12	X	X	X	X	X	X	X
wlan.ip.dhcp.option12_format	X	X	X	X	X	X	X
wlan.ip.dhcp.option12_value				X	X	X	X
wlan.ip.dhcp.request_timeout	X	X	X	X	X	X	X
wlan.ip.dhcp.requests_per_session	X	X	X	X	X	X	X
wlan.ip.dhcp.session_interval	X	X	X	X	X	X	X
wlan.ip.dns.domain				X	X	X	X
wlan.ip.dns.servers				X	X	X	X
wlan.ip.gateway	X	X	X	X	X	X	X
wlan.ip.netmask	X	X	X	X	X	X	X
wlan.ip.port	X	X	X	X	X	X	X
wlan.ip.port_alternate				X	X	X	X
wlan.ip.port_json_config				X	X	X	X
wlan.ip.protocol	X	X	X	X	X	X	X
wlan.ip.timeout.enable	X	X	X	X	X	X	X
wlan.ip.timeout.value	X	X	X	X	X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
wlan.ip.wins.addr	X	X	X	X	X	X	X
wlan.ip.wins.permanent_source	X	X	X	X	X	X	X
wlan.keep_alive.enable	X	X	X	X	X	X	X
wlan.keep_alive.timeout	X	X	X	X	X	X	X
wlan.kerberos.kdc	X	X	X				
wlan.kerberos.mode	X	X	X				
wlan.kerberos.password	X	X	X				
wlan.kerberos.realm	X	X	X				
wlan.kerberos.username	X	X	X				
wlan.leap_mode	X	X	X	X	X	X	X
wlan.leap_password				X	X	X	X
wlan.leap_username	X	X	X	X	X	X	X
wlan.mac_addr	X	X	X	X	X	X	X
wlan.mac_raw	X	X	X	X	X	X	X
wlan.operating_mode	X	X	X	X	X	X	X
wlan.password				X	X	X	X
wlan.permitted_channels				X	X	X	X
wlan.pmf				X	X	X	X
wlan.poor_signal_threshold				X	X	X	X
wlan.preamble	X	X	X	X	X	X	X
wlan.private_key_password				X	X	X	X
wlan.region_code				X	X	X	X
wlan.roam.interchannel_delay				X	X	X	X
wlan.roam.interval	X	X	X	X	X	X	X
wlan.roam.max_chan_scan_time				X	X	X	X
wlan.roam.max_fail				X	X	X	X
wlan.roam.monitor				X	X	X	X
wlan.roam.rssi				X	X	X	X
wlan.roam.signal	X	X	X	X	X	X	X
wlan.rts_cts_enabled				X	X		
wlan.scrambler							
wlan.secure_ssid				X	X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
wlan.security				X	X	X	X
wlan.signal_noise	X	X	X				
wlan.signal_quality	X	X	X				
wlan.signal_strength	X	X	X	X	X	X	X
wlan.station_name	X	X	X	X	X	X	X
wlan.tx_power	X	X	X				
wlan.tx_rate	X	X	X				
wlan.user_channel_list				X	X	X	X
wlan.username	X	X	X	X	X	X	X
wlan.wep.auth_type	X	X	X				
wlan.wep.index	X	X	X				
wlan.wep.key_format	X	X	X				
wlan.wep.key1							
wlan.wep.key2							
wlan.wep.key3							
wlan.wep.key4							
wlan.wpa.groupkey_ciphersuite				X	X	X	X
wlan.wpa.pairwise_ciphersuite				X	X	X	X
wlan.wpa.psk				X	X	X	X
wlan.wpa.timecheck				X	X	X	X
wlan.wpa.wpa_version				X	X	X	X
zbi.control.add_breakpoint				X	X	X	X
zbi.control.break				X	X	X	X
zbi.control.clear_breakpoints				X	X	X	X
zbi.control.delete_breakpoint				X	X	X	X
zbi.control.line_number				X	X	X	X
zbi.control.restart				X	X	X	X
zbi.control.run				X	X	X	X
zbi.control.step				X	X	X	X
zbi.control.terminate				X	X	X	X
zbi.control.variable_name				X	X	X	X
zbi.control.variable_value				X	X	X	X

Command Name	Industrial						
	105SL	110Xi4/ 140Xi4/ 170Xi4/ 220Xi4/	ZM400/ ZM600	ZT210/ ZT220/ ZT230	ZT410/ ZT420	ZT510	ZT610/ ZT620
zbi.enable				X	X	X	X
zbi.key	X	X	X	X	X	X	X
zbi.last_error				X	X	X	X
zbi.program_list				X	X	X	X
zbi.reseller_key				X	X	X	X
zbi.revision	X	X	X	X	X	X	X
zbi.running_program_name				X	X	X	X
zbi.start_info.execute				X	X	X	X
zbi.start_info.file_name				X	X	X	X
zbi.start_info.memory_alloc				X	X	X	X
zbi.state	X	X	X	X	X	X	X
zpl.calibrate				X	X	X	X
zpl.command_prefix				X	X	X	X
zpl.format_prefix				X	X	X	X
zpl.label_length	X	X		X	X	X	X
zpl.label_orientation							
zpl.left_position				X	X	X	X
zpl.relative_darkness				X	X	X	X
zpl.system_error				X	X	X	X
zpl.system_status				X	X	X	X
zpl.zpl_mode				X	X	X	X
zpl.zpl_override							

An X indicates that the SGD is supported for the particular device.

This table indicates support based on the most recent Link-OS system release.

SGDs Supported for Desktop Printers

This table provides details of the list of SGDs supported for Desktop Printers.

Command Name	Desktop									
	GK42	GK888	GT80	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
alerts.add						X	X	X	X	X
alerts.conditions						X	X	X	X	X

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
alerts.configured						X	X	X	X	X
alerts.destinations						X	X	X	X	X
alerts.http.authentication.add						X	X	X	X	X
alerts.http.authentication.entries						X	X	X	X	X
alerts.http.authentication.remove						X	X	X	X	X
alerts.http.logging.clear						X	X	X	X	X
alerts.http.logging.entries						X	X	X	X	X
alerts.http.logging.max_entries						X	X	X	X	X
alerts.http.proxy						X	X	X	X	X
alerts.tracked_settings.clear_log						X	X	X	X	X
alerts.tracked_settings.log_tracked						X	X	X	X	X
alerts.tracked_settings.max_log_entries						X	X	X	X	X
alerts.tracked_settings.zbi_notified						X	X	X	X	X
apl.enable						X	X	X	X	X
apl.framework_version						X	X	X	X	X
apl.version						X	X	X	X	X
appl.bootblock	X	X	X	X	X	X	X	X	X	X
appl.date	X	X	X	X	X	X	X	X	X	X
appl.link_os_version						X	X	X	X	X
appl.name	X	X	X		X	X	X	X	X	X
bluetooth.address				X		X	X	X	X	X
bluetooth.authentication				X						
bluetooth.bluetooth_pin						X	X	X	X	X
bluetooth.clear_bonding_cache						X	X	X	X	X
bluetooth.discoverable				X		X	X	X	X	X
bluetooth.enable				X		X	X	X	X	X
bluetooth.enable_reconnect						X	X	X	X	X
bluetooth.friendly_name				X		X	X	X	X	X
bluetooth.json_config_channel_enable						X	X	X	X	X
bluetooth.le.controller_mode						X	X		X	X
bluetooth.le.minimum_security						X	X		X	X
bluetooth.le.power_class										
bluetooth.local_name				X		X	X	X	X	X
bluetooth.minimum_security_mode						X	X	X	X	X

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
bluetooth.page_scan_window						X	X	X	X	X
bluetooth.radio_version				X		X	X	X	X	X
bluetooth.short_address				X		X	X	X	X	X
bluetooth.version				X		X	X	X	X	X
capture.channel1.count						X	X	X	X	X
capture.channel1.data.mime						X	X	X	X	X
capture.channel1.data.raw						X	X	X	X	X
capture.channel1.delimiter						X	X	X	X	X
capture.channel1.max_length						X	X	X	X	X
capture.channel1.port						X	X	X	X	X
comm.baud	X	X	X	X	X	X	X	X	X	X
comm.halt										
comm.parity	X	X	X	X	X	X	X	X	X	X
comm.stop_bits	X	X	X		X	X	X	X	X	X
comm.type						X	X	X	X	X
cradle.comm.baud										
cradle.comm.handshake										
cutter.clean_cutter	X	X	X	X	X	X	X	X	X	X
device.allow_firmware_downloads						X	X	X	X	X
device.applicator.data_ready										
device.applicator.data_ready_activation										
device.applicator.end_print										
device.applicator.error_on_pause										
device.applicator.feed										
device.applicator.media_out										
device.applicator.pause										
device.applicator.reprint										
device.applicator.rfid_void										
device.applicator.ribbon_low										
device.applicator.ribbon_out										
device.applicator.service_required										
device.applicator.start_print										
device.applicator.start_print_mode										
device.applicator.voltage										

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
device.bluetooth_installed						X	X	X	X	X
device.command_override.active						X	X	X	X	X
device.command_override.add						X	X	X	X	X
device.command_override.clear						X	X	X	X	X
device.command_override.list						X	X	X	X	X
device.company_contact						X	X	X	X	X
device.configuration_number						X	X		X	X
device.cpcl_synchronous_mode						X	X	X	X	X
device.cutter_installed						X	X	X	X	X
device.download_connection_timeout	X	X	X	X	X	X	X	X	X	X
device.download_interactive_mode						X				
device.epl_legacy_mode						X	X	X	X	X
device.feature.802_11ac						X	X	X	X	X
device.feature.bluetooth_le						X	X	X	X	X
device.feature.head_element_test						X	X	X	X	X
device.feature.lighted_arrows						X	X	X	X	X
device.feature.mcr						X	X	X	X	X
device.feature.nfc						X	X	X	X	X
device.feature.ribbon_cartridge						X	X	X	X	X
device.friendly_name	X	X	X	X	X	X	X	X	X	X
device.frontpanel.key_press						X	X	X	X	X
device.frontpanel.line1	X	X		X		X	X	X	X	X
device.frontpanel.line2	X	X		X		X	X	X	X	X
device.frontpanel.xml	X	X	X	X	X	X	X	X	X	X
device.host_identification						X	X	X	X	X
device.host_status						X	X	X	X	X
device.idle_display_format						X	X	X	X	X
device.idle_display_value						X	X	X	X	X
device.internal_wired_setting_location						X	X	X	X	X
device.job_log.total_jobs						X	X	X	X	X
device.jobs_print	X	X	X	X	X	X	X	X	X	X
device.languages	X	X	X	X	X	X	X	X	X	X
device.light.cover_open_brightness										
device.light.head_open_brightness										

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
device.light.media_path_brightness										
device.light.ribbon_path_brightness										
device.locked						X	X		X	X
device.ltu_installed										
device.pause						X	X	X	X	X
device.pmcu.revision						X	X		X	X
device.pnp_option	X	X	X	X	X	X	X	X	X	X
device.position.accuracy						X	X	X	X	X
device.position.altitude						X	X	X	X	X
device.position.latitude						X	X	X	X	X
device.position.longitude						X	X	X	X	X
device.print_2key										
device.print_reprogram_2key						X	X	X	X	X
device.printhead.odometer										
device.printhead.resolution						X	X	X	X	X
device.printhead.test.summary										
device.product_name_submodel						X	X	X	X	X
device.reset						X	X	X	X	X
device.restore_defaults						X	X	X	X	X
device.rewinder_installed										
device.save_2key										
device.sensor_profile							X	X	X	
device.sensor_select						X	X	X	X	X
device.serial_numbers.applicator_option_board										
device.serial_numbers.applicator_option_board_date										
device.serial_numbers.control_panel_date						X	X	X	X	X
device.serial_numbers.cutter						X	X	X	X	X
device.serial_numbers.cutter_date						X	X	X	X	X
device.serial_numbers.mlb_date						X	X	X	X	X
device.serial_numbers.parallel_option_board										
device.serial_numbers.parallel_option_board_date										
device.serial_numbers.printhead										
device.serial_numbers.printhead_date										
device.serial_numbers.usb_host_option_board										

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
device.serial_numbers.usb_host_option_board_date										
device.serial_numbers.wired_ethernet_option_board										
device.serial_numbers.wired_ethernet_option_board_date										
device.set_clock_to_build_date						X	X	X	X	X
device.slot_1						X	X		X	X
device.slot_2										
device.super_host_status						X	X	X	X	X
device.syslog.clear_log						X	X	X	X	X
device.syslog.configuration						X	X	X	X	X
device.syslog.enable						X	X	X	X	X
device.syslog.entries						X	X	X	X	X
device.syslog.log_max_file_size						X	X	X	X	X
device.syslog.save_local_file						X	X	X	X	X
device.unique_id	X	X	X	X	X	X	X	X	X	X
device.unpause						X	X	X	X	X
device.uptime	X	X	X	X	X	X	X	X	X	X
device.user_p1	X	X	X	X	X	X	X	X	X	X
device.user_p2	X	X	X	X	X	X	X	X	X	X
device.user_vars.create						X	X	X	X	X
device.user_vars.set_range						X	X	X	X	X
device.xml.enable	X	X	X	X	X	X	X	X	X	X
display.backlight						X	X	X	X	X
display.backlight_on_time						X	X	X	X	X
display.batch_counter								X		
display.bluetooth.mac										
display.language						X	X	X	X	X
display.load_card								X		
display.password.level								X		
display.root_wml								X		
display.text	X	X	X	X						
external_wired.check	X	X	X	X	X					
external_wired.ip.addr	X	X	X	X	X					
external_wired.ip.arp_interval	X	X	X	X	X					

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
external_wired.ip.default_addr_enable	X	X	X	X	X					
external_wired.ip.gateway	X	X	X	X	X					
external_wired.ip.netmask	X	X	X	X	X					
external_wired.ip.port	X	X	X	X	X					
external_wired.ip.protocol	X	X	X	X	X					
external_wired.ip.timeout.enable	X	X	X	X	X					
external_wired.ip.timeout.value	X	X	X	X	X					
external_wired.ip.v6.addr	X	X	X							
external_wired.ip.v6.gateway	X	X	X							
external_wired.ip.v6.prefix_length	X	X	X							
external_wired.mac_addr	X	X	X	X	X					
external_wired.mac_raw	X	X	X	X	X	X	X	X	X	X
ezpl.head_close_action						X	X	X	X	X
ezpl.label_length_max						X	X	X	X	X
ezpl.label_sensor						X	X	X	X	X
ezpl.manual_calibration						X	X	X	X	X
ezpl.media_type						X	X	X	X	X
ezpl.power_up_action						X	X	X	X	X
ezpl.print_method						X	X	X	X	X
ezpl.print_width						X	X	X	X	X
ezpl.reprint_mode						X	X	X	X	X
ezpl.restore_defaults						X	X	X	X	X
ezpl.take_label						X	X	X	X	X
ezpl.take_label_calibration										
ezpl.tear_off						X	X	X	X	X
file.capture_response.destination						X	X	X	X	X
file.cert.expiration							X	X	X	
file.cert.supported_curves							X	X	X	
file.delete						X	X	X	X	X
file.dir	X	X	X		X	X	X	X	X	X
file.dir_format						X	X	X	X	X
file.run						X	X	X	X	X
file.type						X	X	X	X	X
formats.cancel_all						X	X	X	X	X

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
head.authenticated										
head.darkness_switch						X	X		X	X
head.darkness_switch_enable						X	X		X	X
head.element_test										
head.latch	X	X	X	X	X	X	X	X	X	X
head.resolution.in_dpi						X	X	X	X	X
input.capture						X	X	X	X	X
interface.network.active.arp_interval						X	X	X	X	X
interface.network.active.cable_type						X	X	X	X	X
interface.network.active.dhcp_received_host_name						X	X	X	X	X
interface.network.active.gateway	X	X	X	X	X	X	X	X	X	X
interface.network.active.ip_addr	X	X	X	X	X	X	X	X	X	X
interface.network.active.mac_addr	X	X	X	X	X	X	X	X	X	X
interface.network.active.mac_raw	X	X	X	X	X	X	X	X	X	X
interface.network.active.netmask	X	X	X	X	X	X	X	X	X	X
interface.network.active.printserver	X	X	X	X	X	X	X	X	X	X
interface.network.active.protocol	X	X	X	X	X	X	X	X	X	X
interface.network.active.protocol_error						X	X	X	X	X
interface.network.active.rx_errors						X	X	X	X	X
interface.network.active.rx_packets						X	X	X	X	X
interface.network.active.server_address						X	X	X	X	X
interface.network.active.speed						X	X	X	X	X
interface.network.active.tx_errors						X	X	X	X	X
interface.network.active.tx_packets						X	X	X	X	X
interface.network.active.wins_addr						X	X	X	X	X
interface.network.settings_require_reset						X	X	X	X	X
internal_wired.8021x.password						X	X	X	X	X
internal_wired.8021x.peap.anonymous_identity						X	X	X	X	X
internal_wired.8021x.peap.validate_server_certificate						X	X	X	X	X
internal_wired.8021x.security								X		
internal_wired.8021x.ttls_tunnel						X	X	X	X	X
internal_wired.8021x.username						X	X	X	X	X
internal_wired.activity_led						X	X	X	X	X
internal_wired.auto_switchover	X	X	X	X	X	X	X	X	X	X

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
internal_wired.enable								X		
internal_wired.installed						X	X	X	X	X
internal_wired.ip.addr	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.arp_interval	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.default_addr_enable	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.dhcp.arp_verify						X	X	X	X	X
internal_wired.ip.dhcp.cache_ip	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.dhcp.cid_all	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.dhcp.cid_enable	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.dhcp.cid_prefix	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.dhcp.cid_suffix	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.dhcp.cid_type	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.dhcp.lease.last_attempt	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.dhcp.lease.length	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.dhcp.lease.server	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.dhcp.lease.time_left	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.dhcp.option12	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.dhcp.option12_format	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.dhcp.option12_value	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.dhcp.requests_per_session	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.dns.servers						X	X	X	X	X
internal_wired.ip.gateway	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.netmask	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.port	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.port_alternate						X	X	X	X	X
internal_wired.ip.port_json_config						X	X	X	X	X
internal_wired.ip.protocol	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.timeout.enable	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.timeout.value	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.wins.addr	X	X	X	X	X	X	X	X	X	X
internal_wired.ip.wins.permanent_source	X	X	X	X	X	X	X	X	X	X
internal_wired.mac_addr	X	X	X	X	X	X	X	X	X	X
internal_wired.mac_raw	X	X	X	X	X	X	X	X	X	X
ip.active_network	X	X	X	X	X	X	X	X	X	X

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
ip.addr	X	X	X	X	X	X	X	X	X	X
ip.arp_interval	X	X	X	X	X	X	X	X	X	X
ip.bootp.enable	X	X	X	X	X	X	X	X	X	X
ip.dhcp.arp_verify						X	X	X	X	X
ip.dhcp.auto_provision_enable						X	X	X	X	X
ip.dhcp.cache_ip	X	X	X	X	X	X	X	X	X	X
ip.dhcp.cid_all	X	X	X	X	X	X	X	X	X	X
ip.dhcp.cid_enable	X	X	X	X	X	X	X	X	X	X
ip.dhcp.cid_prefix	X	X	X	X	X	X	X	X	X	X
ip.dhcp.cid_suffix	X	X	X	X	X	X	X	X	X	X
ip.dhcp.cid_type	X	X	X	X	X	X	X	X	X	X
ip.dhcp.cid_value	X	X	X	X	X	X	X	X	X	X
ip.dhcp.enable	X	X	X	X	X	X	X	X	X	X
ip.dhcp.lease.last_attempt	X	X	X	X	X	X	X	X	X	X
ip.dhcp.lease.server	X	X	X	X	X	X	X	X	X	X
ip.dhcp.lease.time_left	X	X	X	X	X	X	X	X	X	X
ip.dhcp.ntp.enable						X	X	X	X	X
ip.dhcp.ntp.received_servers						X	X	X	X	X
ip.dhcp.option12	X	X	X	X	X	X	X	X	X	X
ip.dhcp.option12_format	X	X	X	X	X	X	X	X	X	X
ip.dhcp.option12_value	X	X	X	X	X	X	X	X	X	X
ip.dhcp.request_timeout	X	X	X	X	X	X	X	X	X	X
ip.dhcp.requests_per_session	X	X	X	X	X	X	X	X	X	X
ip.dhcp.session_interval	X	X	X	X	X	X	X	X	X	X
ip.dhcp.user_class_id						X	X	X	X	X
ip.dhcp.vendor_class_id						X	X	X	X	X
ip.dns.servers	X	X	X	X	X	X	X	X	X	X
ip.firewall.whitelist_in						X	X	X	X	X
ip.ftp.enable	X	X	X	X	X	X	X	X	X	X
ip.ftp.execute_file	X	X	X	X	X	X	X	X	X	X
ip.ftp.request_password						X	X	X	X	X
ip.gateway	X	X	X	X	X	X	X	X	X	X
ip.http.admin_name						X	X	X	X	X
ip.http.admin_password						X	X	X	X	X

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
ip.http.custom_link_name						X	X	X	X	X
ip.http.custom_link_url						X	X	X	X	X
ip.http.enable	X	X	X	X	X	X	X	X	X	X
ip.http.faq_url						X	X	X	X	X
ip.http.port						X	X	X	X	X
ip.https.enable						X	X	X	X	X
ip.https.port						X	X	X	X	X
ip.lpd.enable	X	X	X	X	X	X	X	X	X	X
ip.mirror.appl_path						X	X	X	X	X
ip.mirror.auto	X	X	X	X	X	X	X	X	X	X
ip.mirror.error_retry	X	X	X	X	X	X	X	X	X	X
ip.mirror.feedback.auto	X	X	X	X	X	X	X	X	X	X
ip.mirror.feedback.freq	X	X	X	X	X	X	X	X	X	X
ip.mirror.feedback.odometer	X	X	X	X	X	X	X	X	X	X
ip.mirror.feedback.path	X	X	X	X	X	X	X	X	X	X
ip.mirror.fetch						X	X	X	X	X
ip.mirror.freq	X	X	X	X	X	X	X	X	X	X
ip.mirror.freq_hours	X	X	X	X	X	X	X	X	X	X
ip.mirror.interface						X	X	X	X	X
ip.mirror.last_error	X	X	X	X	X	X	X	X	X	X
ip.mirror.last_time	X	X	X	X	X	X	X	X	X	X
ip.mirror.mode						X	X	X	X	X
ip.mirror.password	X	X	X	X		X	X	X	X	X
ip.mirror.path	X	X	X	X	X	X	X	X	X	X
ip.mirror.reset_delay						X	X	X	X	X
ip.mirror.server	X	X	X	X	X	X	X	X	X	X
ip.mirror.success	X	X	X	X	X	X	X	X	X	X
ip.mirror.success_time	X	X	X	X	X	X	X	X	X	X
ip.mirror.username	X	X	X	X	X	X	X	X	X	X
ip.mirror.version	X	X	X	X	X	X	X	X	X	X
ip.netmask	X	X	X	X	X	X	X	X	X	X
ip.ntp.enable						X	X	X	X	X
ip.ntp.log						X	X	X	X	X
ip.ntp.servers						X	X	X	X	X

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
ip.ping_gateway_interval						X	X	X	X	X
ip.ping_remote						X	X	X	X	X
ip.pop3.enable	X	X	X	X	X	X	X	X	X	X
ip.pop3.password	X	X	X	X		X	X	X	X	X
ip.pop3.poll	X	X	X	X	X	X	X	X	X	X
ip.pop3.print_body						X	X	X	X	X
ip.pop3.print_headers						X	X	X	X	X
ip.pop3.save_attachments						X	X	X	X	X
ip.pop3.server_addr	X	X	X	X	X	X	X	X	X	X
ip.pop3.username	X	X	X	X	X	X	X	X	X	X
ip.pop3.verbose_headers						X	X	X	X	X
ip.port	X	X	X	X	X	X	X	X	X	X
ip.port_alternate						X	X	X	X	X
ip.port_json_config						X	X	X	X	X
ip.port_single_conn						X	X	X	X	X
ip.port_single_conn_idle_timeout						X	X	X	X	X
ip.primary_network	X	X	X	X	X	X	X	X	X	X
ip.smtp.domain	X	X	X	X	X	X	X	X	X	X
ip.smtp.enable	X	X	X	X	X	X	X	X	X	X
ip.smtp.server_addr	X	X	X	X	X	X	X	X	X	X
ip.snmp.enable	X	X	X	X	X	X	X	X	X	X
ip.snmp.get_community_name						X	X	X	X	X
ip.snmp.set_community_name						X	X	X	X	X
ip.snmp.trap_community_name						X	X	X	X	X
ip.tcp.enable	X	X	X	X	X	X	X	X	X	X
ip.tcp.nagle_algorithm						X	X	X	X	X
ip.telnet.enable	X	X	X	X	X	X	X	X	X	X
ip.tls.enable						X	X	X	X	X
ip.tls.port						X	X	X	X	X
ip.tls.port_json_config						X	X	X	X	X
ip.udp.enable	X	X	X	X	X	X	X	X	X	X
log.reboot.code						X	X	X	X	X
log.reboot.codes						X	X	X	X	X
log.reboot.reason						X	X	X	X	X

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
log.reboot.report						X	X	X	X	X
mcr.cancel										
mcr.crypt.algorithm										
mcr.crypt.enabled										
mcr.crypt.key_mgmt										
mcr.out										
mcr.query										
mcr.revision										
media.bar_location						X	X	X	X	X
media.cartridge.darkness										
media.cartridge.inserted										
media.cartridge.labels_remaining										
media.cartridge.length										
media.cartridge.part_number					X					
media.cartridge.serial_number										
media.cartridge.speed					X					
media.cartridge.total_label_cnt										
media.cartridge.width										
media.cut_now						X	X	X	X	X
media.darkness_mode	X	X	X	X	X					
media.draft_mode										
media.dynamic_length_calibration						X	X	X	X	X
media.extended_presentation										
media.feed_skip										
media.media_low.external										
media.media_low.warning	X	X		X	X					
media.part_number						X	X	X	X	X
media.present.cut_amount										
media.present.cut_margin										
media.present.eject										
media.present.length_addition										
media.present.loop_length										
media.present.loop_length_max										
media.present.present_timeout										

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
media.present.present_type										
media.printmode	X	X	X	X	X	X	X	X	X	X
media.serial_number						X	X	X	X	X
media.speed	X	X	X	X	X	X	X	X	X	X
memory.flash_free	X	X	X	X	X	X	X	X	X	X
memory.flash_size	X	X	X	X	X	X	X	X	X	X
memory.ram_free	X	X	X	X	X	X	X	X	X	X
memory.ram_size	X	X	X	X	X	X	X	X	X	X
memory.types	X	X	X	X	X					
netmanage.avalanche.agent_addr						X	X	X	X	X
netmanage.avalanche.available_agent						X	X	X	X	X
netmanage.avalanche.available_port						X	X	X	X	X
netmanage.avalanche.encryption_type						X	X	X	X	X
netmanage.avalanche.interval						X	X	X	X	X
netmanage.avalanche.interval_update						X	X	X	X	X
netmanage.avalanche.model_name						X	X	X	X	X
netmanage.avalanche.set_property						X	X	X	X	X
netmanage.avalanche.startup_update						X	X	X	X	X
netmanage.avalanche.tcp_connection_timeout						X	X	X	X	X
netmanage.avalanche.terminal_id						X	X	X	X	X
netmanage.avalanche.text_msg.beep						X	X	X	X	X
netmanage.avalanche.text_msg.display						X	X	X	X	X
netmanage.avalanche.text_msg.print						X	X	X	X	X
netmanage.avalanche.udp_timeout						X	X	X	X	X
netmanage.error_code	X	X	X	X	X	X	X	X	X	X
netmanage.state_code	X	X	X	X	X	X	X	X	X	X
netmanage.status_code	X	X	X	X	X	X	X	X	X	X
odometer.headclean						X	X	X	X	X
odometer.headnew						X	X	X	X	X
odometer.label_dot_length	X	X	X	X	X	X	X	X	X	X
odometer.latch_open_count						X	X	X	X	X
odometer.media_marker_count	X	X	X	X	X	X	X	X	X	X
odometer.media_marker_count1	X	X	X	X	X	X	X	X	X	X
odometer.media_marker_count2	X	X	X	X	X	X	X	X	X	X

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
odometer.retracts_count										
odometer.rfid.valid_resettable								X		
odometer.rfid.void_resettable								X		
odometer.total_cuts						X	X	X	X	X
odometer.total_label_count						X	X	X	X	X
odometer.total_print_length	X	X	X	X	X	X	X	X	X	X
odometer.user_label_count						X	X	X	X	X
odometer.user_label_count1						X	X	X	X	X
odometer.user_label_count2						X	X	X	X	X
odometer.user_total_cuts						X	X	X	X	X
power.average_current						X	X	X	X	X
power.battery_led_blink_rate										
power.battery_led_enable										
power.battery_led_off_duration										
power.battery_led_on_duration										
power.battery_replacement_cyclecount_threshold										
power.battery_type										
power.current										
power.cycle_count										
power.dtr_power_off										
power.energy_star.enable										
power.energy_star.timeout										
power.label_queue.shutdown										
power.part_number										
power.percent_health										
power.power_on_mode							X		X	X
power.remaining_capacity										
power.sleep.cradle										
power.temperature										
power.voltage										
power.wake.radio						X	X		X	X
print.legacy_compatibility										
print.tone	X	X	X	X	X	X	X	X	X	X
print.troubleshooting_label_print										

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
rfid.adaptive_antenna								X		
rfid.antenna_sweep								X		
rfid.country_code								X		
rfid.enable								X		
rfid.error.response								X		
rfid.hop_table_version								X		
rfid.log.clear								X		
rfid.log.enabled								X		
rfid.log.entries								X		
rfid.position.program								X		
rfid.reader_1.antenna_port								X		
rfid.reader_1.firmware_version								X		
rfid.reader_1.hardware_version								X		
rfid.reader_1.model								X		
rfid.reader_1.power.read								X		
rfid.reader_1.power.write								X		
rfid.recipe_version								X		
rfid.region_code								X		
rfid.tag.calibrate								X		
rfid.tag.read.content								X		
rfid.tag.read.execute								X		
rfid.tag.read.result_line1								X		
rfid.tag.read.result_line1_alternate								X		
rfid.tag.read.result_line2								X		
rfid.tag.read.result_line2_alternate								X		
rfid.tag.result_line1										
rfid.tag.test.content								X		
rfid.tag.test.execute								X		
rfid.tag.test.result_line1								X		
rfid.tag.test.result_line2								X		
ribbon.cartridge.authenticated[a]						X	X		X	X
ribbon.cartridge.inserteda						X	X		X	X
ribbon.cartridge.lengtha						X	X		X	X

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
ribbon.cartridge.length_remaininga						X	X		X	X
ribbon.cartridge.part_numbera						X	X		X	X
rtc.timezone						X	X	X	X	X
rtc.unix_timestamp										
sensor.back_bar.brightness										
sensor.back_bar.cur						X	X	X	X	X
sensor.back_bar.gain										
sensor.back_bar.offset										
sensor.back_bar.ppr_out_thold						X	X	X	X	X
sensor.back_bar.thold						X	X	X	X	X
sensor.battery.in_volts										
sensor.cover_open										
sensor.front_bar.brightness										
sensor.front_bar.cur										
sensor.front_bar.gain										
sensor.front_bar.offset										
sensor.front_bar.ppr_out_thold										
sensor.front_bar.thold										
sensor.gap.brightness										
sensor.gap.cur						X	X	X	X	X
sensor.gap.gain										
sensor.gap.offset										
sensor.gap.thold						X	X	X	X	X
sensor.head.temp						X	X	X	X	X
sensor.head.temp_avg						X	X	X	X	X
sensor.head.temp_celsius						X	X	X	X	X
sensor.peel.brightness										
sensor.peel.cur						X	X	X	X	X
sensor.peel.gain										
sensor.peel.thold						X	X	X	X	X
sensor.peeler						X	X	X	X	X
sensor.width.cur										
sensor.width.in_dots										
usb.device.device_id_string						X	X	X	X	X

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
usb.device.device_unique_id						X	X	X	X	X
usb.device.device_version						X	X	X	X	X
usb.device.manufacturer_string						X	X	X	X	X
usb.device.product_id						X	X	X	X	X
usb.device.product_string						X	X	X	X	X
usb.device.serial_string						X	X	X	X	X
usb.device.vendor_id						X	X	X	X	X
usb.halt							X			
usb.host.config_info_to_usb									X	
usb.host.fn_field_data										
usb.host.fn_last_field										
usb.host.hid_count						X	X		X	X
usb.host.keyboard_input										
usb.host.lock_out						X	X		X	X
usb.host.mass_storage_count						X	X		X	X
usb.host.read_list						X	X		X	X
usb.host.read_list_print_delay						X	X		X	X
usb.host.template_list										
usb.host.template_print_amount										
usb.host.write_list						X	X		X	X
usb.mirror.appl_path						X	X		X	X
usb.mirror.auto						X	X		X	X
usb.mirror.enable						X	X		X	X
usb.mirror.error_retry						X	X		X	X
usb.mirror.feedback.auto						X	X		X	X
usb.mirror.feedback.odometer						X	X		X	X
usb.mirror.feedback.path						X	X		X	X
usb.mirror.fetch						X	X		X	X
usb.mirror.last_error						X	X		X	X
usb.mirror.last_time						X	X		X	X
usb.mirror.path						X	X		X	X
usb.mirror.reset_delay						X	X		X	X
usb.mirror.success						X	X		X	X
usb.mirror.success_time						X	X		X	X

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
weblink.cloud_connect.enable						X	X	X	X	X
weblink.enable						X	X	X	X	X
weblink.ip.conn1.authentication.add						X	X	X	X	X
weblink.ip.conn1.authentication.entries						X	X	X	X	X
weblink.ip.conn1.authentication.remove						X	X	X	X	X
weblink.ip.conn1.location						X	X	X	X	X
weblink.ip.conn1.maximum_simultaneous_connections						X	X	X	X	X
weblink.ip.conn1.num_connections						X	X	X	X	X
weblink.ip.conn1.proxy						X	X	X	X	X
weblink.ip.conn1.retry_interval						X	X	X	X	X
weblink.ip.conn1.retry_interval_random_max								X		
weblink.ip.conn1.test.location						X	X	X	X	X
weblink.ip.conn1.test.retry_interval						X	X	X	X	X
weblink.ip.conn1.test.test_on						X	X	X	X	X
weblink.ip.conn2.authentication.add						X	X	X	X	X
weblink.ip.conn2.authentication.entries						X	X	X	X	X
weblink.ip.conn2.authentication.remove						X	X	X	X	X
weblink.ip.conn2.location						X	X	X	X	X
weblink.ip.conn2.maximum_simultaneous_connections						X	X	X	X	X
weblink.ip.conn2.num_connections						X	X	X	X	X
weblink.ip.conn2.proxy						X	X	X	X	X
weblink.ip.conn2.retry_interval						X	X	X	X	X
weblink.ip.conn2.retry_interval_random_max								X		
weblink.ip.conn2.test.location						X	X	X	X	X
weblink.ip.conn2.test.retry_interval						X	X	X	X	X
weblink.ip.conn2.test.test_on						X	X	X	X	X
weblink.logging.clear						X	X	X	X	X
weblink.logging.entries						X	X	X	X	X
weblink.logging.max_entries						X	X	X	X	X
weblink.printer_reset_required						X	X	X	X	X
weblink.restore_defaults						X	X	X	X	X
weblink.zebra_connector.authentication.add						X	X	X	X	X
weblink.zebra_connector.authentication.entries						X	X	X	X	X
weblink.zebra_connector.authentication.remove						X	X	X	X	X

Command Name	Desktop									
	GK42	GK888	GT80	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
weblink.zebra_connector.enable						X	X	X	X	X
weblink.zebra_connector.proxy						X	X	X	X	X
weblink.zebra_connector.version						X	X	X	X	X
wlan.11ac.80mhz_enable						X	X		X	X
wlan.11d.enable						X	X	X	X	X
wlan.11n.20mhz_only						X	X	X	X	X
wlan.11n.greenfield						X	X	X	X	X
wlan.11n.rifs						X	X	X	X	X
wlan.11n.short_gi_20mhz						X	X	X	X	X
wlan.11n.short_gi_40mhz						X	X	X	X	X
wlan.8021x.authentication	X	X	X	X	X	X	X	X	X	X
wlan.8021x.eap.password						X	X	X	X	X
wlan.8021x.eap.privkey_password						X	X	X	X	X
wlan.8021x.eap.username	X	X	X	X	X	X	X	X	X	X
wlan.8021x.enable	X	X	X	X	X	X	X	X	X	X
wlan.8021x.peap.anonymous_identity						X		X		X
wlan.8021x.peap.peap_password						X	X	X	X	X
wlan.8021x.peap.peap_username	X	X	X	X	X	X	X	X	X	X
wlan.8021x.peap.privkey_password						X	X	X	X	X
wlan.active_channels						X	X	X	X	X
wlan.adhoc_last_channel										
wlan.adhocautomode	X	X	X	X	X	X	X	X	X	X
wlan.adhocchannel	X	X	X	X	X	X	X	X	X	X
wlan.allowed_band						X	X	X	X	X
wlan.associated	X	X	X	X	X	X	X	X	X	X
wlan.auth_type	X	X	X	X	X	X	X	X	X	X
wlan.authenticated						X	X	X	X	X
wlan.authentication_error						X	X	X	X	X
wlan.available						X	X	X	X	X
wlan.band_preference						X	X	X	X	X
wlan.bssid	X	X	X	X	X	X	X	X	X	X
wlan.channel	X	X	X	X	X	X	X	X	X	X
wlan.channel_mask						X	X	X	X	X
wlan.country_code						X	X	X	X	X

Command Name	Desktop									
	GK42	GK888	GT80	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
wlan.current_tx_rate	X	X	X	X	X	X	X	X	X	X
wlan.enable						X	X	X	X	X
wlan.encryption_index	X	X	X	X	X		X			
wlan.encryption_key1							X			
wlan.encryption_key2							X			
wlan.encryption_key3							X			
wlan.encryption_key4							X			
wlan.encryption_mode	X	X	X	X	X		X			
wlan.essid	X	X	X	X	X	X	X	X	X	X
wlan.firmware_version	X	X	X	X	X	X	X	X	X	X
wlan.ip.addr	X	X	X	X	X	X	X	X	X	X
wlan.ip.arp_interval	X	X	X	X	X	X	X	X	X	X
wlan.ip.default_addr_enable	X	X	X	X	X	X	X	X	X	X
wlan.ip.dhcp.arp_verify						X	X	X	X	X
wlan.ip.dhcp.cache_ip	X	X	X	X	X	X	X	X	X	X
wlan.ip.dhcp.cid_all						X	X	X	X	X
wlan.ip.dhcp.cid_enable	X	X	X	X	X	X	X	X	X	X
wlan.ip.dhcp.cid_prefix						X	X	X	X	X
wlan.ip.dhcp.cid_suffix						X	X	X	X	X
wlan.ip.dhcp.cid_type	X	X	X	X	X	X	X	X	X	X
wlan.ip.dhcp.lease.last_attempt	X	X	X	X	X	X	X	X	X	X
wlan.ip.dhcp.lease.length	X	X	X	X	X	X	X	X	X	X
wlan.ip.dhcp.lease.server	X	X	X	X	X	X	X	X	X	X
wlan.ip.dhcp.lease.time_left	X	X	X	X	X	X	X	X	X	X
wlan.ip.dhcp.option12	X	X	X	X	X	X	X	X	X	X
wlan.ip.dhcp.option12_format	X	X	X	X	X	X	X	X	X	X
wlan.ip.dhcp.option12_value						X	X	X	X	X
wlan.ip.dhcp.request_timeout	X	X	X	X	X	X	X	X	X	X
wlan.ip.dhcp.requests_per_session	X	X	X	X	X	X	X	X	X	X
wlan.ip.dhcp.session_interval	X	X	X	X	X	X	X	X	X	X
wlan.ip.dns.domain						X	X	X	X	X
wlan.ip.dns.servers						X	X	X	X	X
wlan.ip.gateway	X	X	X	X	X	X	X	X	X	X
wlan.ip.netmask	X	X	X	X	X	X	X	X	X	X

Command Name	Desktop									
	GK42	GK888	GT80	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
wlan.ip.port	X	X	X	X	X	X	X	X	X	X
wlan.ip.port_alternate						X	X	X	X	X
wlan.ip.port_json_config						X	X	X	X	X
wlan.ip.protocol	X	X	X	X	X	X	X	X	X	X
wlan.ip.timeout.enable	X	X	X	X	X	X	X	X	X	X
wlan.ip.timeout.value	X	X	X	X	X	X	X	X	X	X
wlan.ip.wins.addr	X	X	X	X	X	X	X	X	X	X
wlan.ip.wins.permanent_source	X	X	X	X	X	X	X	X	X	X
wlan.keep_alive.enable	X	X	X	X	X	X	X	X	X	X
wlan.keep_alive.timeout	X	X	X	X	X	X	X	X	X	X
wlan.kerberos.kdc	X	X	X	X	X					
wlan.kerberos.mode	X	X	X	X	X					
wlan.kerberos.password	X	X	X		X					
wlan.kerberos.realm	X	X	X	X	X					
wlan.kerberos.username	X	X	X	X	X					
wlan.leap_mode	X	X	X	X	X	X	X	X	X	X
wlan.leap_password						X	X	X	X	X
wlan.leap_username	X	X	X	X	X	X	X	X	X	X
wlan.mac_addr	X	X	X	X	X	X	X	X	X	X
wlan.mac_raw	X	X	X	X	X	X	X	X	X	X
wlan.operating_mode	X	X	X	X	X	X	X	X	X	X
wlan.password						X	X	X	X	X
wlan.permitted_channels						X	X	X	X	X
wlan.pmf						X	X	X	X	X
wlan.poor_signal_threshold						X	X	X	X	X
wlan.preamble	X	X	X	X	X	X	X	X	X	X
wlan.private_key_password						X	X	X	X	X
wlan.region_code						X	X	X	X	X
wlan.roam.interchannel_delay						X	X	X	X	X
wlan.roam.interval	X	X	X	X	X	X	X	X	X	X
wlan.roam.max_chan_scan_time						X	X	X	X	X
wlan.roam.max_fail						X	X	X	X	X
wlan.roam.monitor						X	X	X	X	X
wlan.roam.rssi						X	X	X	X	X

Command Name	Desktop									
	GK42	GK888	GT80	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
wlan.roam.signal	X	X	X	X	X	X	X	X	X	X
wlan.rts_cts_enabled						X	X	X	X	X
wlan.secure_ssid						X	X	X	X	X
wlan.security						X	X	X	X	X
wlan.signal_noise	X	X	X	X						
wlan.signal_quality	X	X	X	X						
wlan.signal_strength	X	X	X	X	X	X	X	X	X	X
wlan.station_name	X	X	X	X	X	X	X	X	X	X
wlan.tx_power	X	X	X	X	X					
wlan.tx_rate	X	X	X	X	X					
wlan.user_channel_list						X	X	X	X	X
wlan.username	X	X	X	X	X	X	X	X	X	X
wlan.wep.auth_type	X	X	X	X	X		X			
wlan.wep.index	X	X	X	X	X		X			
wlan.wep.key_format	X	X	X	X	X		X			
wlan.wep.key1							X			
wlan.wep.key2							X			
wlan.wep.key3							X			
wlan.wep.key4							X			
wlan.wpa.groupkey_ciphersuite						X	X	X	X	X
wlan.wpa.pairwise_ciphersuite						X	X	X	X	X
wlan.wpa.psk						X	X	X	X	X
wlan.wpa.timecheck						X	X	X	X	X
wlan.wpa.wpa_version						X	X	X	X	X
zbi.control.add_breakpoint						X	X	X	X	X
zbi.control.break						X	X	X	X	X
zbi.control.clear_breakpoints						X	X	X	X	X
zbi.control.delete_breakpoint						X	X	X	X	X
zbi.control.line_number						X	X	X	X	X
zbi.control.restart						X	X	X	X	X
zbi.control.run						X	X	X	X	X
zbi.control.step						X	X	X	X	X
zbi.control.terminate						X	X	X	X	X
zbi.control.variable_name						X	X	X	X	X

Command Name	Desktop									
	GK42	GK888	GT800	GX420	HC100	ZD220/ ZD230	ZD410/ ZD420	ZD500/ ZD510	ZD62	ZD888
zbi.control.variable_value						X	X	X	X	X
zbi.enable						X	X	X	X	X
zbi.key	X	X	X	X	X	X	X	X	X	X
zbi.last_error						X	X	X	X	X
zbi.program_list						X	X	X	X	X
zbi.reseller_key						X	X	X	X	X
zbi.revision	X	X	X	X	X	X	X	X	X	X
zbi.running_program_name						X	X	X	X	X
zbi.start_info.execute						X	X	X	X	X
zbi.start_info.file_name						X	X	X	X	X
zbi.start_info.memory_alloc						X	X	X	X	X
zbi.state	X	X	X	X	X	X	X	X	X	X
zpl.calibrate						X	X	X	X	X
zpl.command_prefix						X	X	X	X	X
zpl.format_prefix						X	X	X	X	X
zpl.label_length	X	X	X		X	X	X	X	X	X
zpl.label_orientation							X		X	
zpl.left_position						X	X	X	X	X
zpl.relative_darkness						X	X	X	X	X
zpl.system_error						X	X	X	X	X
zpl.system_status						X	X	X	X	X
zpl.zpl_mode						X	X	X	X	X
zpl.zpl_override										
An X indicates that the SGD is supported for the particular device. This table indicates support based on the most recent Link-OS system release.										

SGDs Supported for Mobile Printers

This table provides details of the list of SGDs supported for Mobile Printers.

Command Name	Mobile									
	iMZ220/ iMZ320	R400	QLn220/ QLn320/ QLn420	ZQ112/ ZQ120/ ZQ220	ZQ310/ ZQ320	ZQ510/ ZQ520	ZQ610/ ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
alerts.add	X		X		X	X	X		X	
alerts.conditions	X		X		X	X	X		X	X
alerts.configured	X		X		X	X	X		X	X
alerts.destinations	X		X		X	X	X		X	X
alerts.http.authentication.add	X		X		X	X	X		X	X
alerts.http.authentication.entries	X		X		X	X	X		X	X
alerts.http.authentication.remove	X		X		X	X	X		X	X
alerts.http.logging.clear	X		X		X	X	X		X	X
alerts.http.logging.entries	X		X		X	X	X		X	X
alerts.http.logging.max_entries	X		X		X	X	X		X	X
alerts.http.proxy	X		X		X	X	X		X	X
alerts.tracked_settings.clear_log	X		X		X	X	X		X	X
alerts.tracked_settings.log_tracked	X		X		X	X	X		X	X
alerts.tracked_settings.max_log_entries	X		X		X	X	X		X	X
alerts.tracked_settings.zbi_notified	X		X		X	X	X		X	X
apl.enable	X		X		X	X	X		X	X
apl.framework_version	X		X		X	X	X		X	X
apl.version	X		X		X	X	X		X	X
appl.bootblock	X	X	X	X	X	X	X	X	X	X
appl.date	X	X	X	X	X	X	X	X	X	X
appl.link_os_version	X		X		X	X	X		X	X
appl.name	X	X	X	X	X	X	X	X	X	X
bluetooth.address	X		X	X	X	X	X	X	X	X
bluetooth.authentication										
bluetooth.bluetooth_pin	X		X	X	X	X	X	X	X	X
bluetooth.clear_bonding_cache	X		X	X	X	X	X	X	X	X
bluetooth.discoverable	X		X	X	X	X	X	X	X	X
bluetooth.enable	X		X	X	X	X	X	X	X	X
bluetooth.enable_reconnect	X		X	X	X	X	X	X	X	X
bluetooth.friendly_name	X		X	X	X	X	X	X	X	X

Command Name	Mobile									
	iMZ22 iMZ32	R40	QLn220/ QLn320/ QLn420	ZQ112/ZQ310 ZQ120/ZQ32 ZQ220		ZQ510 ZQ52	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
bluetooth.json_config_channel_enable	X		X		X	X	X		X	X
bluetooth.le.controller_mode						X	X		X	X
bluetooth.le.minimum_security						X				
bluetooth.le.power_classa							X			
bluetooth.local_name	X		X	X	X	X	X	X	X	X
bluetooth.minimum_security_mode	X		X	X	X	X	X	X	X	X
bluetooth.page_scan_window	X		X	X	X	X	X	X	X	X
bluetooth.radio_version	X		X	X	X	X	X	X	X	X
bluetooth.short_address	X		X	X	X	X	X	X	X	X
bluetooth.version	X		X	X	X	X	X	X	X	X
capture.channel1.count	X		X		X	X	X		X	X
capture.channel1.data.mime	X		X		X	X	X		X	X
capture.channel1.data.raw	X		X		X	X	X		X	X
capture.channel1.delimiter	X		X		X	X	X		X	X
capture.channel1.max_length	X		X		X	X	X		X	X
capture.channel1.port	X		X		X	X	X		X	X
comm.baud		X	X				X			X
comm.halt			X				X			X
comm.parity		X	X				X			X
comm.stop_bits		X	X				X			X
comm.type										
cradle.comm.baud						X				
cradle.comm.handshake						X				
cutter.clean_cutter		X								
device.allow_firmware_downloads	X		X	X	X	X	X	X		
device.applicator.data_ready										
device.applicator.data_ready_activation										
device.applicator.end_print										
device.applicator.error_on_pause										
device.applicator.feed										
device.applicator.media_out										
device.applicator.pause										

Command Name	Mobile									
	iMZ22(iMZ32(	R40	QLn220/ QLn320/ QLn420	ZQ112/ZQ310 ZQ120/ZQ32(ZQ220	ZQ510 ZQ52(	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
device.applicator.reprint										
device.applicator.rfid_void										
device.applicator.ribbon_low										
device.applicator.ribbon_out										
device.applicator.service_required										
device.applicator.start_print										
device.applicator.start_print_mode										
device.applicator.supplies_low_selection										
device.applicator.voltage										
device.bluetooth_installed	X		X		X	X	X		X	X
device.command_override.active	X		X		X	X	X		X	X
device.command_override.add	X		X		X	X	X		X	X
device.command_override.clear	X		X		X	X	X		X	X
device.command_override.list	X		X		X	X	X		X	X
device.company_contact	X		X		X	X	X		X	X
device.configuration_number	X		X		X	X	X		X	
device.cpcl_synchronous_mode	X		X		X	X	X		X	X
device.cutter_installed	X		X		X	X	X		X	X
device.download_connection_timeout	X	X	X		X	X	X		X	X
device.download_interactive_mode										
device.epl_legacy_modea	X		X		X	X	X		X	
device.feature.802_11aca	X		X		X	X	X		X	
device.feature.bluetooth_lea	X		X		X	X	X		X	
device.feature.head_element_testa	X		X		X	X	X		X	
device.feature.lighted_arrowsa	X		X		X	X	X		X	
device.feature.mcr	X		X		X	X	X		X	X
device.feature.nfc	X		X		X	X	X		X	X
device.feature.ribbon_cartridge	X		X		X	X	X		X	X
device.friendly_name	X	X	X	X	X	X	X	X	X	X
device.frontpanel.key_press			X			X	X			X
device.frontpanel.line1		X	X				X			X
device.frontpanel.line2		X	X				X			X

Command Name	Mobile									
	iMZ22(iMZ32(	R40	QLn220/ QLn320/ QLn420	ZQ112/ ZQ120/ ZQ220	ZQ310 ZQ32(	ZQ510 ZQ52(	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
device.frontpanel.xml		X	X			X	X			X
device.host_identification	X		X		X	X	X		X	X
device.host_status	X		X		X	X	X		X	X
device.idle_display_format			X			X	X			X
device.idle_display_value			X			X	X			X
device.internal_wired_setting_location			X				X			X
device.job_log.total_jobs	X		X		X	X	X		X	X
device.jobs_printa	X	X	X	X	X	X	X	X	X	
device.languages	X	X	X	X	X	X	X	X	X	X
device.light.cover_open_brightness										
device.light.head_open_brightness										
device.light.media_path_brightness										
device.light.ribbon_path_brightness										
device.locked										
device.ltu_installed	X		X		X	X	X		X	X
device.pause	X		X		X	X	X		X	X
device.pmcu.revisiona						X	X			
device.pnp_option	X	X	X		X	X	X		X	X
device.position.accuracy	X		X		X	X	X		X	X
device.position.altitude	X		X		X	X	X		X	X
device.position.latitude	X		X		X	X	X		X	X
device.position.longitude	X		X		X	X	X		X	X
device.print_2key	X		X		X	X	X		X	X
device.print_reprogram_2key	X		X		X	X	X		X	X
device.printhead.odometer										
device.printhead.resolution	X		X		X	X	X		X	X
device.printhead.test.summary										
device.product_name_submodel	X		X		X	X	X		X	X
device.reset	X		X	X	X	X	X	X	X	X
device.restore_defaults	X		X	X	X	X	X	X	X	X
device.rewinder_installed	X		X		X	X	X		X	X
device.save_2key	X		X	X	X	X	X	X	X	X

Command Name	Mobile									
	iMZ22 iMZ32	R40	QLn220/ QLn320/ QLn420	ZQ112/ZQ310 ZQ120/ZQ32 ZQ220		ZQ510 ZQ52	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
device.sensor_profile	X		X		X	X	X		X	X
device.sensor_selecta							X			
device.serial_numbers.applicator_option_board										
device.serial_numbers.applicator_option_board_date										
device.serial_numbers.control_panel_datea	X		X		X	X	X		X	
device.serial_numbers.cuttera	X		X		X	X	X		X	
device.serial_numbers.cutter_datea	X		X		X	X	X		X	
device.serial_numbers.mlb_date	X		X		X	X	X		X	
device.serial_numbers.option_board	X		X		X	X	X		X	X
device.serial_numbers.option_board_datea	X		X		X	X	X		X	
device.serial_numbers.parallel_option_board										
device.serial_numbers.parallel_option_board_date										
device.serial_numbers.printhead										
device.serial_numbers.printhead_date										
device.serial_numbers.usb_host_option_board										
device.serial_numbers.usb_host_option_board_date										
device.serial_numbers.wired_ethernet_option_board										
device.serial_numbers.wired_ethernet_option_board_date										
device.set_clock_to_build_datea	X		X				X			
device.slot_1										
device.slot_2										
device.super_host_statusa	X				X		X		X	
device.syslog.clear_log	X		X		X	X	X		X	X
device.syslog.configuration	X		X		X	X	X		X	X
device.syslog.enable	X		X		X	X	X		X	X
device.syslog.entriesa	X		X		X	X	X		X	
device.syslog.log_max_file_size	X		X		X	X	X		X	X
device.syslog.save_local_file	X		X		X	X	X		X	X
device.unique_id	X	X	X		X	X	X		X	X
device.unpause	X		X		X	X	X		X	X
device.uptime	X	X	X	X	X	X	X	X	X	X
device.user_p1	X	X	X	X	X	X	X	X	X	X

Command Name	Mobile									
	iMZ22 iMZ32	R40	QLn220/ QLn320/ QLn420	ZQ112/ ZQ120/ ZQ220	ZQ310 ZQ32	ZQ510 ZQ52	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
device.user_p2	X	X	X	X	X	X	X	X	X	X
device.user_vars.create	X		X		X	X	X		X	X
device.user_vars.set_range	X		X		X	X	X		X	X
device.xml.enable	X	X	X		X	X	X		X	X
display.backlight			X			X	X			X
display.backlight_on_time			X			X	X			X
display.batch_countera			X			X	X			
display.bluetooth.mac				X				X		
display.language			X	X		X	X	X		X
display.load_carda			X				X			
display.password.levela			X				X			
display.root_wml			X				X			X
display.text		X	X			X	X			X
external_wired.check		X								
external_wired.ip.addr		X								
external_wired.ip.arp_interval		X								
external_wired.ip.default_addr_enable		X								
external_wired.ip.gateway		X								
external_wired.ip.netmask		X								
external_wired.ip.port		X								
external_wired.ip.protocol		X								
external_wired.ip.timeout.enable		X								
external_wired.ip.timeout.value		X								
external_wired.ip.v6.addr										
external_wired.ip.v6.gateway										
external_wired.ip.v6.prefix_length										
external_wired.mac_addr		X								
external_wired.mac_raw		X								
ezpl.head_close_action	X		X	X	X	X	X	X	X	X
ezpl.label_length_max	X		X		X	X	X		X	X
ezpl.label_sensor	X		X		X	X	X		X	X
ezpl.manual_calibration	X		X		X	X	X		X	X

Command Name	Mobile									
	iMZ22(iMZ32(	R40	QLn220/ QLn320/ QLn420	ZQ112/ZQ310 ZQ120/ZQ32(ZQ220		ZQ510 ZQ52(	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
ezpl.media_type	X		X		X	X	X		X	X
ezpl.power_up_action	X		X	X	X	X	X	X	X	X
ezpl.print_method	X		X		X	X	X		X	X
ezpl.print_width	X		X		X	X	X		X	X
ezpl.reprint_mode	X		X		X	X	X		X	X
ezpl.restore_defaults	X		X		X	X	X		X	X
ezpl.take_label	X		X		X	X	X		X	X
ezpl.take_label_calibration										
ezpl.tear_off	X		X		X	X	X		X	X
file.capture_response.destinationa	X		X		X	X	X		X	
file.capture_response.enda	X		X		X	X	X		X	
file.cert.expiration					X	X	X			
file.cert.supported_curvesa					X	X	X		X	
file.delete	X		X	X	X	X	X	X	X	X
file.dir	X		X	X	X	X	X	X	X	X
file.dir_format	X		X		X	X	X		X	X
file.run	X		X	X	X	X	X	X	X	X
file.type	X		X	X	X	X	X	X	X	X
formats.cancel_all	X		X		X	X	X		X	X
head.authenticated										
head.darkness_switch										
head.darkness_switch_enable										
head.element_test						X	X			X
head.latch	X	X	X	X	X	X	X	X	X	X
head.resolution.in_dpia	X		X		X	X	X		X	
input.capture	X		X	X	X	X	X	X	X	X
interface.network.active.arp_interval	X		X		X	X	X		X	X
interface.network.active.cable_type	X		X		X	X	X		X	X
interface.network.active.dhcp_received_host_name	X		X		X	X	X		X	X
interface.network.active.gateway	X	X	X		X	X	X		X	X
interface.network.active.ip_addr	X	X	X		X	X	X		X	X
interface.network.active.mac_addr	X	X	X		X	X	X		X	X

Command Name	Mobile									
	iMZ22(iMZ32(	R40	QLn220/ QLn320/ QLn420	ZQ112/ ZQ120/ ZQ220	ZQ310 ZQ32(	ZQ510 ZQ52(	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
interface.network.active.mac_raw	X	X	X		X	X	X		X	X
interface.network.active.netmask	X	X	X		X	X	X		X	X
interface.network.active.printserver	X	X	X		X	X	X		X	X
interface.network.active.protocol	X	X	X		X	X	X		X	X
interface.network.active.protocol_error	X		X		X	X	X		X	X
interface.network.active.rx_errors	X		X		X	X	X		X	X
interface.network.active.rx_packets	X		X		X	X	X		X	X
interface.network.active.server_address	X		X		X	X	X		X	X
interface.network.active.speed	X		X		X	X	X		X	X
interface.network.active.tx_errors	X		X		X	X	X		X	X
interface.network.active.tx_packets	X		X		X	X	X		X	X
interface.network.active.wins_addr	X		X		X	X	X		X	X
interface.network.settings_require_reset	X		X		X	X	X		X	X
internal_wired.8021x.passworda			X				X			
internal_wired.8021x.peap.anonymous_identitya			X				X			
internal_wired.8021x.peap.validate_server_certificatea			X				X			
internal_wired.8021x.securitya			X				X			
internal_wired.8021x.ttls_tunnela			X				X			
internal_wired.8021x.usernamea			X				X			
internal_wired.activity_led			X				X			X
internal_wired.auto_switchover	X	X	X		X	X	X		X	X
internal_wired.enablea			X				X			
internal_wired.installed	X		X		X	X	X		X	X
internal_wired.ip.addr	X	X	X		X	X	X		X	X
internal_wired.ip.arp_interval	X	X	X		X	X	X		X	X
internal_wired.ip.default_addr_enable	X	X	X		X	X	X		X	X
internal_wired.ip.dhcp.arp_verify	X		X		X	X	X		X	X
internal_wired.ip.dhcp.cache_ip	X	X	X		X	X	X		X	X
internal_wired.ip.dhcp.cid_all		X	X				X			X
internal_wired.ip.dhcp.cid_enable	X	X	X		X	X	X		X	X
internal_wired.ip.dhcp.cid_prefix		X	X				X			X
internal_wired.ip.dhcp.cid_suffix		X	X				X			X

Command Name	Mobile									
	iMZ22(iMZ32(	R40	QLn220/ QLn320/ QLn420	ZQ112/ZQ310 ZQ120/ZQ32(ZQ220		ZQ510 ZQ52(	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
internal_wired.ip.dhcp.cid_type	X	X	X		X	X	X		X	X
internal_wired.ip.dhcp.lease.last_attempt	X	X	X		X	X	X		X	X
internal_wired.ip.dhcp.lease.length	X	X	X		X	X	X		X	X
internal_wired.ip.dhcp.lease.server	X	X	X		X	X	X		X	X
internal_wired.ip.dhcp.lease.time_left	X	X	X		X	X	X		X	X
internal_wired.ip.dhcp.option12	X	X	X		X	X	X		X	X
internal_wired.ip.dhcp.option12_format	X	X	X		X	X	X		X	X
internal_wired.ip.dhcp.option12_value		X	X				X			X
internal_wired.ip.dhcp.requests_per_session	X	X	X		X	X	X		X	X
internal_wired.ip.dns.servers			X				X			X
internal_wired.ip.gateway	X	X	X		X	X	X		X	X
internal_wired.ip.netmask	X	X	X		X	X	X		X	X
internal_wired.ip.port	X	X	X		X	X	X		X	X
internal_wired.ip.port_alternate	X		X		X	X	X		X	X
internal_wired.ip.port_json_config			X				X			X
internal_wired.ip.protocol	X	X	X		X	X	X		X	X
internal_wired.ip.timeout.enable	X	X	X		X	X	X		X	X
internal_wired.ip.timeout.value	X	X	X		X	X	X		X	X
internal_wired.ip.wins.addr	X	X	X		X	X	X		X	X
internal_wired.ip.wins.permanent_source	X	X	X		X	X	X		X	X
ip.active_network	X	X	X		X	X	X		X	X
ip.addr	X		X		X	X	X		X	X
ip.arp_interval	X		X		X	X	X		X	X
ip.bootp.enable	X		X		X	X	X		X	X
ip.dhcp.arp_verify	X		X		X	X	X		X	X
ip.dhcp.auto_provision_enable	X		X		X	X	X		X	X
ip.dhcp.cache_ip	X		X		X	X	X		X	X
ip.dhcp.cid_all	X		X		X	X	X		X	X
ip.dhcp.cid_enable	X		X		X	X	X		X	X
ip.dhcp.cid_prefix	X		X		X	X	X		X	X
ip.dhcp.cid_suffix	X		X		X	X	X		X	X
ip.dhcp.cid_type	X		X		X	X	X		X	X

Command Name	Mobile									
	iMZ22(iMZ32(	R40	QLn220/ QLn320/ QLn420	ZQ112/ ZQ120/ ZQ220	ZQ310 ZQ32(	ZQ510 ZQ52(	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
ip.dhcp.cid_value	X		X		X	X	X		X	X
ip.dhcp.enable	X		X		X	X	X		X	X
ip.dhcp.lease.last_attempt	X		X		X	X	X		X	X
ip.dhcp.lease.server	X		X		X	X	X		X	X
ip.dhcp.lease.time_left	X		X		X	X	X		X	X
ip.dhcp.ntp.enable	X		X		X	X	X		X	X
ip.dhcp.ntp.received_servers	X		X		X	X	X		X	X
ip.dhcp.option12	X		X		X	X	X		X	X
ip.dhcp.option12_format	X		X		X	X	X		X	X
ip.dhcp.option12_value	X		X		X	X	X		X	X
ip.dhcp.request_timeout	X		X		X	X	X		X	X
ip.dhcp.requests_per_session	X		X		X	X	X		X	X
ip.dhcp.session_interval	X		X		X	X	X		X	X
ip.dhcp.user_class_id	X		X		X	X	X		X	X
ip.dhcp.vendor_class_id	X		X		X	X	X		X	X
ip.dns.servers	X	X	X		X	X	X		X	X
ip.firewall.whitelist_ina	X		X		X	X	X			
ip.ftp.enable	X	X	X		X	X	X		X	X
ip.ftp.execute_file	X	X	X		X	X	X		X	X
ip.ftp.request_password	X		X		X	X	X		X	X
ip.gateway	X		X		X	X	X		X	X
ip.http.admin_name	X		X		X	X	X		X	X
ip.http.admin_password	X		X		X	X	X		X	X
ip.http.custom_link_name	X		X		X	X	X		X	X
ip.http.custom_link_url	X		X		X	X	X		X	X
ip.http.enable	X	X	X		X	X	X		X	X
ip.http.faq_url	X		X		X	X	X		X	X
ip.http.port	X		X		X	X	X		X	X
ip.https.enablea	X		X		X	X	X			
ip.https.porta	X		X		X	X	X			
ip.lpd.enable	X	X	X		X	X	X		X	X
ip.mirror.appl_path	X		X		X	X	X		X	X

Command Name	Mobile									
	iMZ22(iMZ32(	R40	QLn220/ QLn320/ QLn420	ZQ112/ ZQ120/ ZQ220	ZQ310 ZQ32(	ZQ510 ZQ52(	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
ip.mirror.auto	X	X	X		X	X	X		X	X
ip.mirror.error_retry	X	X	X		X	X	X		X	X
ip.mirror.feedback.auto	X	X	X		X	X	X		X	X
ip.mirror.feedback.freq	X	X	X		X	X	X		X	X
ip.mirror.feedback.odometer	X	X	X		X	X	X		X	X
ip.mirror.feedback.path	X	X	X		X	X	X		X	X
ip.mirror.fetch	X		X		X	X	X		X	X
ip.mirror.freq	X	X	X		X	X	X		X	X
ip.mirror.freq_hours	X	X	X		X	X	X		X	X
ip.mirror.interface	X		X		X	X	X		X	X
ip.mirror.last_error	X	X	X		X	X	X		X	X
ip.mirror.last_time	X	X	X		X	X	X		X	X
ip.mirror.mode	X		X		X	X	X		X	X
ip.mirror.password	X		X		X	X	X		X	X
ip.mirror.path	X	X	X		X	X	X		X	X
ip.mirror.reset_delay	X		X		X	X	X		X	X
ip.mirror.server	X	X	X		X	X	X		X	X
ip.mirror.success	X	X	X		X	X	X		X	X
ip.mirror.success_time	X	X	X		X	X	X		X	X
ip.mirror.username	X	X	X		X	X	X		X	X
ip.mirror.version	X	X	X		X	X	X		X	X
ip.netmask	X		X		X	X	X		X	X
ip.ntp.enable	X		X		X	X	X		X	X
ip.ntp.log	X		X		X	X	X		X	X
ip.ntp.servers	X		X		X	X	X		X	X
ip.ping_gateway_interval	X		X		X	X	X		X	X
ip.ping_remote	X		X		X	X	X		X	X
ip.pop3.enable	X	X	X		X	X	X		X	X
ip.pop3.password	X		X		X	X	X		X	X
ip.pop3.poll	X	X	X		X	X	X		X	X
ip.pop3.print_body	X		X		X	X	X		X	X
ip.pop3.print_headers	X		X		X	X	X		X	X

Command Name	Mobile									
	iMZ22(iMZ32(	R40	QLn220/ QLn320/ QLn420	ZQ112/ZQ310 ZQ120/ZQ32(ZQ220		ZQ510 ZQ52(	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
ip.pop3.save_attachments	X		X		X	X	X		X	X
ip.pop3.server_addr	X	X	X		X	X	X		X	X
ip.pop3.username	X	X	X		X	X	X		X	X
ip.pop3.verbose_headers	X		X		X	X	X		X	X
ip.port	X		X		X	X	X		X	X
ip.port_alternate	X		X		X	X	X		X	X
ip.port_json_config	X		X		X	X	X		X	X
ip.port_single_conn	X		X		X	X	X		X	X
ip.port_single_conn_idle_timeout	X		X		X	X	X		X	X
ip.primary_network	X	X	X		X	X	X		X	X
ip.smtp.domain	X	X	X		X	X	X		X	X
ip.smtp.enable	X	X	X		X	X	X		X	X
ip.smtp.server_addr	X	X	X		X	X	X		X	X
ip.snmp.enable	X	X	X		X	X	X		X	X
ip.snmp.get_community_name	X		X		X	X	X		X	X
ip.snmp.set_community_name	X		X		X	X	X		X	X
ip.snmp.trap_community_name	X		X		X	X	X		X	X
ip.tcp.enable	X	X	X		X	X	X		X	X
ip.tcp.nagle_algorithm	X		X		X	X	X		X	X
ip.telnet.enablea	X	X	X		X	X	X		X	X
ip.tls.enablea	X		X		X	X	X			
ip.tls.porta	X		X		X	X	X			
ip.tls.port_json_configa	X		X		X	X	X			
ip.udp.enable	X	X	X		X	X	X		X	X
log.reboot.code	X		X	X	X	X	X	X	X	X
log.reboot.codes	X		X	X	X	X	X	X	X	X
log.reboot.reason	X		X	X	X	X	X	X	X	X
log.reboot.report	X		X	X	X	X	X	X	X	X
mcr.cancel						X				
mcr.crypt.algorithm						X				
mcr.crypt.enabled						X				
mcr.crypt.key_mgmt						X				

Command Name	Mobile									
	iMZ22 iMZ32	R40	QLn220/ QLn320/ QLn420	ZQ112/ZQ310 ZQ120/ZQ32 ZQ220		ZQ510 ZQ52	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
mcr.out						X				
mcr.query						X				
mcr.revision						X				
media.bar_locationa	X		X	X	X	X	X	X	X	
media.cartridge.darkness										
media.cartridge.inserted										
media.cartridge.labels_remaining										
media.cartridge.length										
media.cartridge.part_number										
media.cartridge.serial_number										
media.cartridge.speed										
media.cartridge.total_label_cnt										
media.cartridge.width										
media.cut_now										
media.darkness_mode		X								
media.draft_modea	X		X		X	X	X		X	
media.dynamic_length_calibration										
media.extended_presentation										
media.feed_skip	X		X		X	X	X		X	X
media.linerless_offset										
media.media_low.external		X								
media.media_low.warning		X								
media.part_numbera	X		X		X	X	X		X	
media.present.cut_amount		X								
media.present.cut_margin		X								
media.present.eject		X								
media.present.length_addition		X								
media.present.loop_length		X								
media.present.loop_length_max		X								
media.present.present_timeout		X								
media.present.present_type		X								
media.printmode	X	X	X		X	X	X		X	X

Command Name	Mobile									
	iMZ22(iMZ32(	R40	QLn220/ QLn320/ QLn420	ZQ112/ ZQ120/ ZQ220	ZQ310 ZQ32(	ZQ510 ZQ52(	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
media.serial_numbera	X		X		X	X	X		X	
media.speed	X	X	X		X	X	X		X	X
memory.flash_free	X	X	X	X	X	X	X	X	X	X
memory.flash_size	X	X	X	X	X	X	X	X	X	X
memory.ram_free	X	X	X	X	X	X	X	X	X	X
memory.ram_size	X	X	X	X	X	X	X	X	X	X
memory.types		X								
netmanage.avalanche.agent_addr	X		X		X	X	X		X	X
netmanage.avalanche.available_agent	X		X		X	X	X		X	X
netmanage.avalanche.available_port	X		X		X	X	X		X	X
netmanage.avalanche.encryption_type	X		X		X	X	X		X	X
netmanage.avalanche.interval	X		X		X	X	X		X	X
netmanage.avalanche.interval_update	X		X		X	X	X		X	X
netmanage.avalanche.model_name	X		X		X	X	X		X	X
netmanage.avalanche.set_property	X		X		X	X	X		X	X
netmanage.avalanche.startup_update	X		X		X	X	X		X	X
netmanage.avalanche.tcp_connection_timeout	X		X		X	X	X		X	X
netmanage.avalanche.terminal_id	X		X		X	X	X		X	X
netmanage.avalanche.text_msg.beep	X		X		X	X	X		X	X
netmanage.avalanche.text_msg.display	X		X		X	X	X		X	X
netmanage.avalanche.text_msg.print	X		X		X	X	X		X	X
netmanage.avalanche.udp_timeout	X		X		X	X	X		X	X
netmanage.error_code	X	X	X		X	X	X		X	X
netmanage.state_code	X	X	X		X	X	X		X	X
netmanage.status_code	X	X	X		X	X	X		X	X
odometer.headclean	X		X		X	X	X		X	X
odometer.headnew	X		X		X	X	X		X	X
odometer.label_dot_length	X	X	X	X	X	X	X	X	X	X
odometer.latch_open_count	X		X	X	X	X	X	X	X	X
odometer.media_marker_count	X	X	X	X	X	X	X	X	X	X
odometer.media_marker_count1	X	X	X		X	X	X		X	X
odometer.media_marker_count2	X	X	X		X	X	X		X	X

Command Name	Mobile									
	iMZ22(iMZ32(	R40	QLn220/ QLn320/ QLn420	ZQ112/ZQ310 ZQ120/ZQ32(ZQ220	ZQ510 ZQ52(	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
odometer.retracts_count		X								
odometer.rfid.valid_resettable										
odometer.rfid.void_resettable										
odometer.total_cuts										
odometer.total_label_count	X		X	X	X	X	X	X	X	X
odometer.total_print_length	X	X	X	X	X	X	X	X	X	X
odometer.user_label_count	X		X	X	X	X	X	X	X	X
odometer.user_label_count1	X		X		X	X	X		X	X
odometer.user_label_count2	X		X		X	X	X		X	X
odometer.user_total_cuts										
power.average_current	X		X		X	X	X		X	X
power.battery_led_blink_ratea			X			X	X			
power.battery_led_enablea			X			X	X			
power.battery_led_off_durationa			X			X	X			
power.battery_led_on_durationa			X			X	X			
power.battery_replacement_cyclecount_thresholda			X			X	X			
power.battery_replacement_stateofcharge_thresholda			X			X	X			
power.battery_typea	X		X		X	X	X			
power.currenta					X		X		X	
power.cycle_count			X		X	X	X		X	X
power.dtr_power_off	X		X		X	X	X		X	X
power.energy_star.enable										
power.energy_star.timeout										
power.label_queue.shutdown	X		X		X	X	X		X	X
power.part_numbera					X		X		X	
power.percent_healtha					X		X		X	
power.power_on_mode										
power.remaining_capacitya					X		X		X	
power.serial_number			X		X	X	X		X	X
power.sleep.cradle						X	X			X
power.temperaturea			X		X		X		X	
power.voltage	X		X	X	X	X	X	X	X	X

Command Name	Mobile									
	iMZ22 iMZ32	R40	QLn220/ QLn320/ QLn420	ZQ112/ZQ310 ZQ120/ZQ32 ZQ220		ZQ510 ZQ52	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
power.wake.radio						X	X		X	X
print.legacy_compatibilitya							X			
print.tone	X	X	X	X	X	X	X	X	X	X
print.troubleshooting_label_printa							X			
rfid.adaptive_antenna										
rfid.antenna_sweep										
rfid.country_code										
rfid.enable										
rfid.error.response										
rfid.hop_table_version										
rfid.log.clear										
rfid.log.enabled										
rfid.log.entries										
rfid.position.program										
rfid.reader_1.antenna_port										
rfid.reader_1.firmware_version										
rfid.reader_1.hardware_version										
rfid.reader_1.model										
rfid.reader_1.power.read										
rfid.reader_1.power.write										
rfid.recipe_version										
rfid.region_code										
rfid.tag.calibrate										
rfid.tag.read.content										
rfid.tag.read.execute										
rfid.tag.read.result_line1										
rfid.tag.read.result_line1_alternate										
rfid.tag.read.result_line2										
rfid.tag.read.result_line2_alternate										
rfid.tag.result_line1										
rfid.tag.test.content										
rfid.tag.test.execute										

Command Name	Mobile									
	iMZ22(iMZ32(	R40	QLn220/ QLn320/ QLn420	ZQ112/ZQ310 ZQ120/ZQ32(ZQ220		ZQ510 ZQ52(	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
rfid.tag.test.result_line1										
rfid.tag.test.result_line2										
ribbon.cartridge.authenticated										
ribbon.cartridge.inserted										
ribbon.cartridge.length										
ribbon.cartridge.length_remaining										
ribbon.cartridge.part_number										
rtc.timezone	X		X		X	X	X		X	X
rtc.unix_timestamp										
sensor.back_bar.brightnessa	X		X			X	X			
sensor.back_bar.cura	X		X		X	X	X			
sensor.back_bar.gaina	X		X			X	X			
sensor.back_bar.offseta	X		X			X	X			
sensor.back_bar.ppr_out_tholda	X		X		X	X	X			
sensor.back_bar.tholda	X		X		X	X	X			
sensor.battery.in_voltsa	X		X			X	X			
sensor.cover_open										
sensor.front_bar.brightness	X									
sensor.front_bar.cur	X									
sensor.front_bar.gain	X									
sensor.front_bar.offset	X									
sensor.front_bar.ppr_out_thold	X									
sensor.front_bar.thold	X									
sensor.gap.brightnessa	X		X			X	X			
sensor.gap.cura	X		X		X	X	X			
sensor.gap.gaina	X		X				X			
sensor.gap.offseta	X		X			X	X			
sensor.gap.tholda	X		X		X	X	X			
sensor.head.tempa	X		X		X	X	X			
sensor.head.temp_avga	X		X		X	X	X			
sensor.head.temp_celsiusa	X		X		X	X	X			
sensor.peel.brightnessa	X		X			X	X			

Command Name	Mobile									
	iMZ22(iMZ32(	R40	QLn220/ QLn320/ QLn420	ZQ112/ ZQ120/ ZQ220	ZQ310 ZQ32(	ZQ510 ZQ52(	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
sensor.peel.cura	X		X		X	X	X			
sensor.peel.gaina	X		X			X	X			
sensor.peel.thold	X		X		X	X	X		X	X
sensor.peeler	X		X		X	X	X		X	X
sensor.width.cura	X		X		X	X	X		X	
sensor.width.in_dotsa	X		X		X	X	X		X	
test.enable										
test.newdemo										
testdemo.enable										
usb.device.device_id_string	X		X	X	X	X	X	X	X	X
usb.device.device_unique_id	X		X	X	X	X	X	X	X	X
usb.device.device_version	X		X	X	X	X	X	X	X	X
usb.device.manufacturer_string	X		X	X	X	X	X	X	X	X
usb.device.product_id	X		X	X	X	X	X	X	X	X
usb.device.product_string	X		X	X	X	X	X	X	X	X
usb.device.serial_string	X		X	X	X	X	X	X	X	X
usb.device.vendor_id	X		X	X	X	X	X	X	X	X
usb.halt	X		X						X	X
usb.host.config_info_to_usb										
usb.host.fn_field_data										
usb.host.fn_last_field										
usb.host.hid_count						X				
usb.host.keyboard_input										
usb.host.lock_out						X				
usb.host.mass_storage_count						X				
usb.host.read_list										
usb.host.read_list_print_delay										
usb.host.template_list										
usb.host.template_print_amount										
usb.host.write_list										
usb.mirror.appl_path						X				
usb.mirror.auto						X				

Command Name	Mobile									
	iMZ22(iMZ32(	R40	QLn220/ QLn320/ QLn420	ZQ112/ZQ310 ZQ120/ZQ32(ZQ220		ZQ510 ZQ52(	ZQ610/ ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
usb.mirror.enable						X				
usb.mirror.error_retry						X				
usb.mirror.feedback.auto						X				
usb.mirror.feedback.odometer						X				
usb.mirror.feedback.path						X				
usb.mirror.fetch						X				
usb.mirror.last_error						X				
usb.mirror.last_time						X				
usb.mirror.path						X				
usb.mirror.reset_delay						X				
usb.mirror.success						X				
usb.mirror.success_time						X				
weblink.cloud_connect.enable	X		X		X	X	X		X	X
weblink.enable	X		X		X	X	X		X	X
weblink.ip.conn1.authentication.add	X		X		X	X	X		X	X
weblink.ip.conn1.authentication.entries	X		X		X	X	X		X	X
weblink.ip.conn1.authentication.remove	X		X		X	X	X		X	X
weblink.ip.conn1.location	X		X		X	X	X		X	X
weblink.ip.conn1.maximum_simultaneous_connections	X		X		X	X	X		X	X
weblink.ip.conn1.num_connections	X		X		X	X	X		X	X
weblink.ip.conn1.proxy	X		X		X	X	X		X	X
weblink.ip.conn1.retry_interval	X		X		X	X	X		X	X
weblink.ip.conn1.retry_interval_random_maxa	X		X			X	X			
weblink.ip.conn1.test.location	X		X		X	X	X		X	X
weblink.ip.conn1.test.retry_interval	X		X		X	X	X		X	X
weblink.ip.conn1.test.test_on	X		X		X	X	X		X	X
weblink.ip.conn2.authentication.add	X		X		X	X	X		X	X
weblink.ip.conn2.authentication.entries	X		X		X	X	X		X	X
weblink.ip.conn2.authentication.remove	X		X		X	X	X		X	X
weblink.ip.conn2.location	X		X		X	X	X		X	X
weblink.ip.conn2.maximum_simultaneous_connections	X		X		X	X	X		X	X
weblink.ip.conn2.num_connections	X		X		X	X	X		X	X

Command Name	Mobile									
	iMZ22(iMZ32(	R40	QLn220/ QLn320/ QLn420	ZQ112/ ZQ120/ ZQ220	ZQ310 ZQ32(	ZQ510 ZQ52(	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
weblink.ip.conn2.proxy	X		X		X	X	X		X	X
weblink.ip.conn2.retry_interval	X		X		X	X	X		X	X
weblink.ip.conn2.retry_interval_random_maxa	X		X			X	X			
weblink.ip.conn2.test.location	X		X		X	X	X		X	X
weblink.ip.conn2.test.retry_interval	X		X		X	X	X		X	X
weblink.ip.conn2.test.test_on	X		X		X	X	X		X	X
weblink.logging.clear	X		X		X	X	X		X	X
weblink.logging.entries	X		X		X	X	X		X	X
weblink.logging.max_entries	X		X		X	X	X		X	X
weblink.printer_reset_required	X		X		X	X	X		X	X
weblink.restore_defaults	X		X		X	X	X		X	X
weblink.zebra_connector.authentication.add	X		X		X	X	X		X	X
weblink.zebra_connector.authentication.entries	X		X		X	X	X		X	X
weblink.zebra_connector.authentication.remove	X		X		X	X	X		X	X
weblink.zebra_connector.enable	X		X		X	X	X		X	X
weblink.zebra_connector.proxy	X		X		X	X	X		X	X
weblink.zebra_connector.version	X		X		X	X	X		X	X
wlan.11ac.80mhz_enablea							X			
wlan.11d.enable	X		X		X	X	X		X	X
wlan.11n.20mhz_only	X		X		X	X	X		X	X
wlan.11n.greenfield	X		X		X	X	X		X	X
wlan.11n.rifs	X		X		X	X	X		X	X
wlan.11n.short_gi_20mhz	X		X		X	X	X		X	X
wlan.11n.short_gi_40mhz	X		X		X	X	X		X	X
wlan.8021x.authentication	X	X	X		X	X	X		X	X
wlan.8021x.eap.password	X		X		X	X	X		X	X
wlan.8021x.eap.privkey_password	X		X		X	X	X		X	X
wlan.8021x.eap.username	X	X	X		X	X	X		X	X
wlan.8021x.enable	X	X	X		X	X	X		X	X
wlan.8021x.peap.anonymous_identitya	X		X			X	X			
wlan.8021x.peap.peap_password	X		X		X	X	X		X	X
wlan.8021x.peap.peap_username	X	X	X		X	X	X		X	X

Command Name	Mobile									
	iMZ22 iMZ32	R40	QLn220/ QLn320/ QLn420	ZQ112/ ZQ120/ ZQ220	ZQ310 ZQ32	ZQ510 ZQ52	ZQ610/ ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
wlan.8021x.peap.privkey_password	X		X		X	X	X		X	X
wlan.active_channels	X		X		X	X	X		X	X
wlan.adhoc_last_channel										
wlan.adhocautomode	X	X	X		X	X	X		X	X
wlan.adhocchannel	X	X	X		X	X	X		X	X
wlan.allowed_band	X		X		X	X	X		X	X
wlan.associated	X	X	X		X	X	X		X	X
wlan.auth_type	X	X	X		X	X			X	X
wlan.authenticated	X		X		X	X	X		X	X
wlan.authentication_error	X		X		X	X	X		X	X
wlan.available	X		X		X	X	X		X	X
wlan.band_preference			X		X	X	X			
wlan.bssid	X	X	X		X	X	X		X	X
wlan.channel	X	X	X		X	X	X		X	X
wlan.channel_mask	X		X		X	X	X		X	X
wlan.country_code	X		X		X	X	X		X	X
wlan.current_tx_rate	X	X	X		X	X	X		X	X
wlan.enable	X		X		X	X	X		X	X
wlan.encryption_index		X							X	X
wlan.encryption_key1									X	X
wlan.encryption_key2									X	X
wlan.encryption_key3									X	X
wlan.encryption_key4									X	X
wlan.encryption_mode		X							X	X
wlan.essid	X	X	X		X	X	X		X	X
wlan.firmware_version	X	X	X		X	X	X		X	X
wlan.ip.addr	X	X	X		X	X	X		X	X
wlan.ip.arp_interval	X	X	X		X	X	X		X	X
wlan.ip.default_addr_enable	X	X	X		X	X	X		X	X
wlan.ip.dhcp.arp_verify	X		X		X	X	X		X	X
wlan.ip.dhcp.cache_ip	X	X	X		X	X	X		X	X
wlan.ip.dhcp.cid_all	X		X		X	X	X		X	X

Command Name	Mobile									
	iMZ22(iMZ32(	R40	QLn220/ QLn320/ QLn420	ZQ112/ ZQ120/ ZQ220	ZQ310 ZQ32(	ZQ510 ZQ52(	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
wlan.ip.dhcp.cid_enable	X	X	X		X	X	X		X	X
wlan.ip.dhcp.cid_prefix	X		X		X	X	X		X	X
wlan.ip.dhcp.cid_suffix	X		X		X	X	X		X	X
wlan.ip.dhcp.cid_type	X	X	X		X	X	X		X	X
wlan.ip.dhcp.lease.last_attempt	X	X	X		X	X	X		X	X
wlan.ip.dhcp.lease.length	X	X	X		X	X	X		X	X
wlan.ip.dhcp.lease.server	X	X	X		X	X	X		X	X
wlan.ip.dhcp.lease.time_left	X	X	X		X	X	X		X	X
wlan.ip.dhcp.option12	X	X	X		X	X	X		X	X
wlan.ip.dhcp.option12_format	X	X	X		X	X	X		X	X
wlan.ip.dhcp.option12_value	X		X		X	X	X		X	X
wlan.ip.dhcp.request_timeout	X	X	X		X	X	X		X	X
wlan.ip.dhcp.requests_per_session	X	X	X		X	X	X		X	X
wlan.ip.dhcp.session_interval	X	X	X		X	X	X		X	X
wlan.ip.dns.domain	X		X		X	X	X		X	X
wlan.ip.dns.servers	X		X		X	X	X		X	X
wlan.ip.gateway	X	X	X		X	X	X		X	X
wlan.ip.netmask	X	X	X		X	X	X		X	X
wlan.ip.port	X	X	X		X	X	X		X	X
wlan.ip.port_alternate	X		X		X	X	X		X	X
wlan.ip.port_json_config	X		X		X	X	X		X	X
wlan.ip.protocol	X	X	X		X	X	X		X	X
wlan.ip.timeout.enable	X	X	X		X	X	X		X	X
wlan.ip.timeout.value	X	X	X		X	X	X		X	X
wlan.ip.wins.addr	X	X	X		X	X	X		X	X
wlan.ip.wins.permanent_source	X	X	X		X	X	X		X	X
wlan.keep_alive.enable	X	X	X		X	X	X		X	X
wlan.keep_alive.timeout	X	X	X		X	X	X		X	X
wlan.kerberos.kdc		X								
wlan.kerberos.mode		X								
wlan.kerberos.password										
wlan.kerberos.realm		X								

Command Name	Mobile									
	iMZ22(iMZ32(	R40	QLn220/ QLn320/ QLn420	ZQ112/ZQ310 ZQ120/ZQ32(ZQ220		ZQ510 ZQ52(	ZQ610/ ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
wlan.kerberos.username		X								
wlan.leap_mode	X	X	X		X	X	X		X	X
wlan.leap_password	X		X		X	X	X		X	X
wlan.leap_username	X	X	X		X	X	X		X	X
wlan.mac_addr	X	X	X		X	X	X		X	X
wlan.mac_raw	X	X	X		X	X	X		X	X
wlan.operating_mode	X	X	X		X	X	X		X	X
wlan.password	X		X		X	X	X		X	X
wlan.permitted_channels	X		X		X	X	X		X	X
wlan.pmfa	X		X		X	X	X		X	
wlan.poor_signal_thresholda	X		X		X	X	X		X	
wlan.power_save	X		X		X	X	X		X	X
wlan.preamble	X	X	X		X	X	X		X	X
wlan.private_key_password	X		X		X	X	X		X	X
wlan.region_code	X		X		X	X	X		X	X
wlan.roam.interchannel_delay	X		X		X	X	X		X	X
wlan.roam.interval	X	X	X		X	X	X		X	X
wlan.roam.max_chan_scan_time	X		X		X	X	X		X	X
wlan.roam.max_fail	X		X		X	X	X		X	X
wlan.roam.monitor	X		X		X	X	X		X	X
wlan.roam.rssi	X		X		X	X	X		X	X
wlan.roam.signal	X	X	X		X	X	X		X	X
wlan.rts_cts_enabled	X		X		X	X	X		X	X
wlan.secure_ssid	X		X		X	X	X		X	X
wlan.security	X		X		X	X	X		X	X
wlan.signal_noise		X								
wlan.signal_quality		X								
wlan.signal_strength	X	X	X		X	X	X		X	X
wlan.station_name	X	X	X		X	X	X		X	X
wlan.tx_power		X								
wlan.tx_rate		X								
wlan.user_channel_list	X		X		X	X	X		X	X

Command Name	Mobile									
	iMZ22 iMZ32	R40	QLn220/ QLn320/ QLn420	ZQ112/ ZQ120/ ZQ220	ZQ310 ZQ32	ZQ510 ZQ52	ZQ610 ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
wlan.username	X	X	X		X	X	X		X	X
wlan.wep.auth_typeb		X							X	X
wlan.wep.indexb		X							X	X
wlan.wep.key_formatb		X							X	X
wlan.wep.key1b									X	X
wlan.wep.key2b									X	X
wlan.wep.key3b									X	X
wlan.wep.key4[a]									X	X
wlan.wpa.groupkey_ciphersuite	X		X		X	X	X		X	X
wlan.wpa.pairwise_ciphersuite	X		X		X	X	X		X	X
wlan.wpa.psk	X		X		X	X	X		X	X
wlan.wpa.timechecka	X		X		X	X	X		X	
wlan.wpa.wpa_version	X		X		X	X	X		X	X
zbi.control.add_breakpoint	X		X		X	X	X		X	X
zbi.control.break	X		X		X	X	X		X	X
zbi.control.clear_breakpoints	X		X		X	X	X		X	X
zbi.control.delete_breakpoint	X		X		X	X	X		X	X
zbi.control.line_number	X		X		X	X	X		X	X
zbi.control.restart	X		X		X	X	X		X	X
zbi.control.run	X		X		X	X	X		X	X
zbi.control.step	X		X		X	X	X		X	X
zbi.control.terminate	X		X		X	X	X		X	X
zbi.control.variable_name	X		X		X	X	X		X	X
zbi.control.variable_value	X		X		X	X	X		X	X
zbi.enablea	X		X		X	X	X			
zbi.key	X	X	X		X	X	X		X	X
zbi.last_error	X		X		X	X	X		X	X
zbi.program_list	X		X		X	X	X		X	X
zbi.reseller_key	X		X		X	X	X		X	X
zbi.revision	X	X	X		X	X	X		X	X
zbi.running_program_name	X		X		X	X	X		X	X
zbi.start_info.execute	X		X		X	X	X		X	X

Command Name	Mobile									
	iMZ22X/ iMZ32(	R40	QLn220/ QLn320/ QLn420	ZQ112/ ZQ120/ ZQ220	ZQ310/ ZQ320	ZQ510/ ZQ520	ZQ610/ ZQ620/ ZQ630	ZR13	ZR318/ ZR328/ ZR338	ZR628/ ZR638/ ZR658/ ZR668
zbi.start_info.file_name	X		X		X	X	X		X	X
zbi.start_info.memory_alloc	X		X		X	X	X		X	X
zbi.state	X	X	X		X	X	X		X	X
zpl.calibrate	X		X		X	X	X		X	X
zpl.command_prefix	X		X		X	X	X		X	X
zpl.format_prefix	X		X		X	X	X		X	X
zpl.label_length	X	X	X		X	X	X		X	X
zpl.label_orientation					X	X	X			
zpl.left_position	X		X		X	X	X		X	X
zpl.relative_darknessa	X		X		X	X	X		X	
zpl.system_error	X		X		X	X	X		X	X
zpl.system_status	X		X		X	X	X		X	X
zpl.zpl_mode	X		X		X	X	X		X	X
zpl.zpl_override										
An X indicates that the SGD is supported for the particular device.This table indicates support based on the most recent Link-OS system release.										

Mirror

This section provides an overview of Mirror, details on how to use it, and configuration examples.

Mirror Overview

Mirror is a feature that gives you the ability to:

- Centrally manage and monitor the deployment of your Zebra printers
- Centrally configure and maintain your Zebra printers through remote updates
- Remotely monitor printer updates - via the "Feedback" feature

There are several Set/Get/Do (SGD) commands that are used to configure and initiate Mirror. For details see, Mirror Printer Configuration.

Benefits

When using Mirror, updating the configuration and firmware on the printer is remotely managed from a centralized FTP server. Configurations can be uniformly deployed to individual printers or to groups of printers. Unique Configurations can also be targeted to printers as needed.

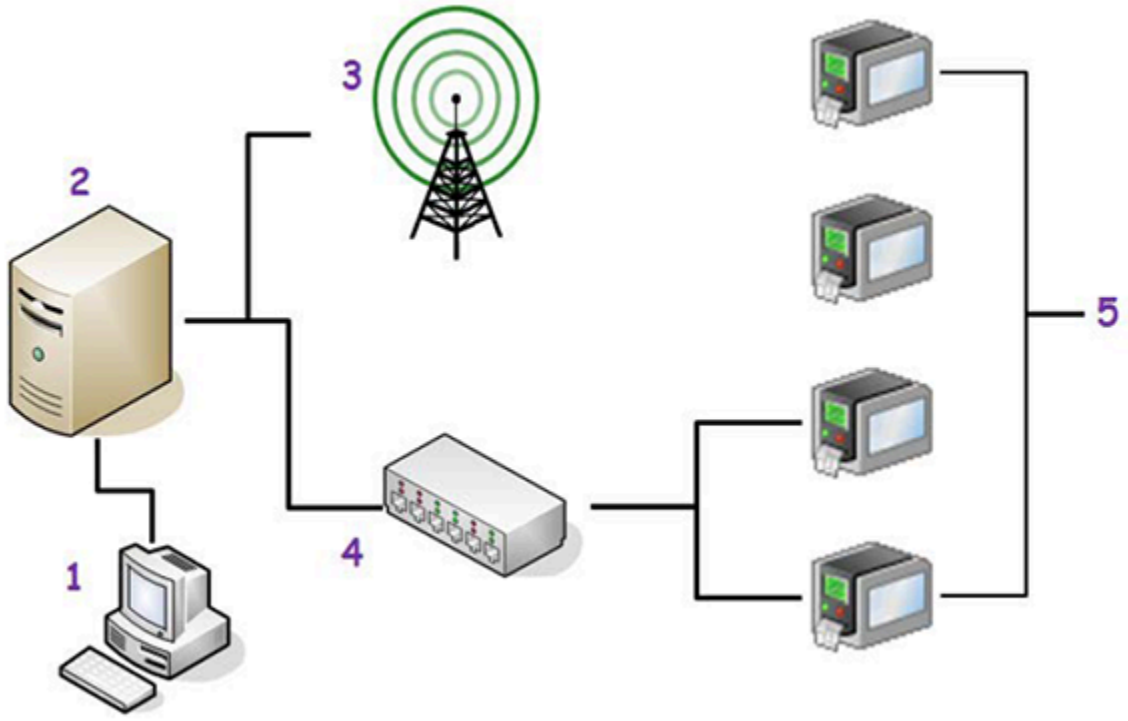
Through the "Feedback" feature, Administrators can easily log and monitor configuration updates on a printer-by-printer basis.

Typical uses of Mirror include:

- configuring printers as they are first received
- performing scheduled maintenance updates, sending firmware, fonts, graphics and other objects to the printer as needed
- changing printer Configurations in order to move printers from one role to another

This figure provides an illustration of Mirroring.

Figure 29 Mirror Illustration



1	Workstation — sends SGD commands to the printer to configure it for Mirror use.
2	FTP Server — Stores configuration files and responds to Mirror requests from the printer. Receives and stores “Feedback” content.
3	Access Point — wireless network infrastructure
4	Hub — wired network infrastructure
5	Zebra Printer(s) — Configured using SGD commands. Sends Mirror requests to the FTP server to receive files. Transmits “Feedback” content to the FTP server to log Mirror event transactions and resulting printer settings.

Professional Services for Mirror Configuration

Zebra offers a Professional Services group that can help with the configuring the Mirror feature. To inquire about Zebra's Professional Services, contact your Zebra account representative.

Requirements

These are the requirements for Mirror:

- Zebra printer loaded with Mirror capable firmware and Print Server. For details, see Mirror Printer Configuration.

- FTP server (with UNIX-style directory listings), configured with the following directories:
  - `"<update-root>"/appl` - This directory is used for printer firmware updates. During an update operation, the printer will first check the `"<update-root>/appl"` directory for new printer firmware
  - `"<update-root>"/files` - This directory is used for printer-resident files. Files in this directory will be stored locally on the printer's file system. Files are not processed by the printer; they are only stored.
  - `"<update-root>"/commands` - This directory is used for a limited number of printer executable commands. The contents of files in this directory will be sent to the printer's command interpreter for execution.
  - `"<feedback-root>"` - This optional directory is used to receive Mirror feedback content from the printer. The content sent to this directory is defined by the `"feedback.get"` template file stored on the printer.
- A client account (user name and password) on the FTP server for the printer to use.
- A Terminal Emulation program, used to send SGD commands to the printer to configure Mirror.

Supported Printers and Print Server Types

These are the Zebra printers, firmware versions, and Zebra print servers that support Mirror.

Table 40 Supported Printers and Print Server Types

Printer	Firmware	Print Servers
ZT400	V75.19.7Z (or later)	• ZebraNet a/b/g/n Print Server • Internal Wireless Plus • Wireless Plus
ZE500	V53.17.15Z (or later)	• ZebraNet a/b/g/n Print Server • Internal Wireless Plus • Wireless Plus
ZT200	V72.18.2Z (or later)	• ZebraNet a/b/g/n Print Server • Internal Wireless Plus • Wireless Plus
105SLPlus	V53.17.15Z (or later)	• ZebraNet b/g Print Server • Internal Wireless Plus • Wireless Plus
Xi4™ series	V53.17.5Z (or later)	• Internal, Integrated 10/100 IPV4 wired • Internal Wireless Plus
XiIIIPlus™ series	V60.17.5Z (or later)	• Internal Wireless Plus • Wireless Plus
105SL™	V60.17.5Z (or later)	• Internal Wireless Plus • Wireless Plus
PAX4™	V60.17.5Z (or later)	• Internal Wireless Plus • Wireless Plus
ZM400™	V53.17.5Z (or later)	• Internal 10/100 wired\ • Internal Wireless Plus • Wireless Plus
ZM600™	V53.17.5Z (or later)	• Internal 10/100 wired\ • Internal Wireless Plus • Wireless Plus

Table 40 Supported Printers and Print Server Types (Continued)

Printer	Firmware	Print Servers
S4M™	V53.17.5Z (or later)	• Internal Wireless Plus • Wireless Plus
G™ -series	V56.17.5ZV61.17.5Z (or later)	• Internal 10/100 wired\ • Internal Wireless Plus • Wireless Plus
LP 2824 Plus TLP 2824 Plus	V61.17.3Z (or later)	• Internal 10/100 wired\ • Internal Wireless Plus • Wireless Plus

NOTE: Print Servers connected to the parallel port, either externally or internally, are not supported for Mirror use.

How Mirror Works

Mirror utilizes a network connection and FTP communications to perform remote updates and feedback operations. At printer startup and/or at regular intervals, the printer will establish a FTP connection back to a central FTP server and check for updates. During an update operation, the printer will check for updated files in specific directories at the FTP server.

These are the specific FTP server directories that will be checked:

- `"<update-root>"/appl` - This directory is used for printer firmware updates. During an update operation, the printer will first check the "<update-root>/appl" directory for new printer firmware
- `"<update-root>"/files` - This directory is used for printer-resident files. Files in this directory will be stored locally on the printer's file system. Files are not processed by the printer; they are only stored.
- `"<update-root>"/commands` - This directory is used for a limited number of printer executable commands. The contents of files in this directory will be sent to the printer's command interpreter for execution.
- `"<feedback-root>"` - This optional directory is used to receive Mirror feedback content from the printer. The content sent to this directory is defined by the "feedback.get" template file stored on the printer.

NOTE: "`<update-root>`" refers to the value of the "`ip.mirror.path`" configuration parameter. Files in the `<update-root>/files` directory should not have download headers in them. They should be in the exact format they will be in when stored on the printer's file system. Examples of download headers are: `~DY, ~DG, ! CISDFCRC16` or `~DF`.

IMPORTANT:

When the printer is in the Mirror process:

- It is unavailable for other tasks.
- The LCD will indicate that it is performing a Mirror function, showing when the printer is downloading firmware and the names of the object files as they are transferred to the printer.

Mirror Process Summary

The Mirror process follows a specific series of steps:

1. After power-up, the printer will first check the `<update-root>/appl` directory on the FTP server for new printer firmware and update the printer if necessary
2. If the printer did not find new firmware to download, it will then check in the `<update-root>\files` and `<update-root>/commands` directories for updated files - and download them as needed.
3. As a final Mirror step, the printer can perform an optional Feedback operation, transmitting a file of user-defined printer configuration information to `<feedback-root>` directory on the FTP server.
4. Finally, if any files or commands were downloaded during Step 2:, the printer will automatically reset itself.

Mirror Process Details

The following items are important to be aware of when configuring the FTP server to support Firmware updated via Mirror.

- Firmware files must be named using the following format: `<firmware version>.zpl`, where <firmware version> is the exact Firmware revision contained in the file. For example, for Firmware version “V53.17.5Z”, the filename stored on the FTP server must be “V53.17.5Z.ZPL”. If the file name and Firmware version do not match, the update will not succeed.
- The firmware filename stored on the FTP server is not case sensitive. This means that “V53.17.5Z.ZPL” and “v53.17.5Z.zpl” will be processed in the exact same way.
- The `/appl` directory can contain only one (1) file at a time. If there is more than 1 file in this directory, the printer will not download anything and will skip the firmware update.

During a Mirror event, the printer will use `<firmware version>` part of the filename contained in the `/appl` directory on the FTP server to check if the Firmware stored on the FTP server is different than the firmware the printer is currently using. If the `<firmware version>` part of the Firmware file name in the `/appl` directory on the FTP server does not exactly match the printer's Firmware version, the file on the FTP server will be downloaded and used to update the printer. Once the printer is updated with the new Firmware, the printer will reset and being using the new Firmware.

NOTE: Performing the Firmware update first is important because it is possible that the files or commands to be downloaded via Mirror will be dependent on the new firmware.

After the printer has completed the Firmware portion of an update operation, it will check the "`<update-root>/files`" directory and "`<update-root>/commands`" directory (in that order) for updated files that need to be retrieved. If a file exists on the FTP server and the server timestamp for the file does not match the printer's archived timestamp for the file, the printer will re-download the file and update its timestamp info. Files that are on the Mirror FTP server but not currently present on the printer will be sent to the printer. The supported file types are the standard files supported on ZPL printers.

If any files are downloaded during this portion of an update operation, the printer will reboot after the file downloads have completed. At this point, the update operation is complete.

IMPORTANT:

- Files in the `<update-root>/files` directory should not have download headers in them. They should be in the exact format they will be in when stored on the printer's file system. Examples of download headers are: `~DY, ~DG, ! CISDFCRC16` or `~DF`.

- ZPL files in the <update-root>/files directory must use the printers internal characters for the Format Command Prefix (^), Delimiter Character (,) and Control Command Character (~). This means that the caret (^) should be replaced with a HEX 1E, the comma (,) should be replaced with a HEX 1F and the tilde (~) should be replaced with a HEX 10.

After an update operation is complete, a printer will perform a feedback operation, if configured to do so. During a feedback operation, the printer will open its feedback template file (named feedback.get), populate it based on its contents, and upload the resulting contents file to the FTP server, in the "<feedback-root>" directory.

NOTE: "<feedback-root>" refers to the value of the "ip.mirror.feedback.path" configuration parameter

Creating ZPL Files for Use in the <update-root>/files Directory

When creating ZPL formats that will be stored in the "<update-root>/files" directory it is necessary to edit the files using the following guidelines:

1. Files must contain only one format. This means that if a file contains multiple ^XA and ^XZ commands, those sections of the file must be split into separate formats, or combined as one format.
2. The characters used for the Format Command Prefix (^), Delimiter Character (,) and Control Command Characters (~) must be substituted for their Hexadecimal equivalents.
3. The ^XA and ^XZ commands must be removed from the formats.
4. The ^DF command should be removed.

One Format per File

Files must contain only one format. This means that if a file contains multiple ^XA and ^XZ commands, those sections of the file must be split into separate formats, or combined as one format. For example, if a ZPL file contains both an initialization string and a format, the two sections must be either split into two files, or combined into one format.

For example, given the following formats:

```
^XA
^LT0^MNW^MTT^PON^PMN^LH0,0^JMA^PR2,2^LRN^CI0
^XZ
^XA
^FO20,100^IME:ZEBRA.BMP^FS
^A@N,75,75,TT0003M_.TTF
^FO20,400^FDZebra Technologies^FS
^XZ
```

The following file should be created, which includes all of the command in one file:

```
^XA
^LT0^MNW^MTT^PON^PMN^LH0,0^JMA^PR2,2^LRN^CI0
^FO20,100^IME:ZEBRA.BMP^FS
^A@N,75,75,TT0003M_.TTF
^FO20,400^FDZebra Technologies^FS
^XZ
```

Character Substitution

The characters used for the Format Command Prefix (^), Delimiter Character (,) and Control Command Characters (~) must be substituted for their Hexadecimal equivalents. During normal operation, this is how the printer works with ZPL formats. For example, when a ZPL format is sent to the printer via a telnet, RS-232 or Ethernet connection and stored for later use, it automatically processes the file and makes these character substitutions.

When ZPL formats are sent to the printer from the `"<update-root>/files"` directory they are stored on the printers memory, but not processed. For this reason, it is necessary to preprocess the files so that they are ready for use.

The following character substitutions must be made to files sent from the `"<update-root>/files"` directory:

Original Character	Substitute Character
Command PrefixThe default is the Caret (^)	HEX 1E
Delimeter PrefixThe default is the comma (,)	HEX 1F
Control PrefixThe default is the tilde (~)	HEX 10

For example, given the following format:

```
^XA
^FO20,100^IME:ZEBRA.BMP^FS
^A@N,75,75,TT0003M_.TTF
^FO20,400^FDZebra Technologies^FS^XZ
```

It would be necessary to replace the ^ characters with a HEX 1E and the , characters with a HEX 1F. This can be done using a Text Editor. See Example Files for more information.

Removing the ^XA and ^XZ commands

Additionally, the ^XA and ^XZ commands should be removed from the format. The printer will automatically add these commands back in to process the file. See Example Files for more information.

Removing the ^DF command

In some cases, you might have been using files that contain the ^DF command. The purpose of the ^DF command is to instruct the printer to store everything that comes after it in a ZPL file. For example, you might have a file that contains the following:

```
^XA
^DFE:STOREFMT.ZPL^FS
^FO25,25^AD,36,20^FN1^FS
^FO165,25^AD,36,20^FN2^FS
^FO25,75^AB,22,14^FDBUILT BY^FS
^FO25,125^AE,28,15^FN1
^XZ
```

The purpose of the above format - when sent to a printer - would be to store a file called "STOREFMT" to the E memory location on the printer. In production, the goal would be to recall and print the "STOREFMT" file using the following ZPL commands:

```
^XA
^XFE:STOREFMT.ZPL^FS
^FN1^FDZEBRA^FS
^FN2^FDPRINTER^FS
^XZ
```

When this is done using Mirror, the format being sent to the printer must be altered. The line with the ^DF command must be removed - this is because the Mirror process is taking care of storing the format to the E memory location. In this scenario, the original format would be edited to look like this:

```
^FO25,25^AD,36,20^FN1^FS
^FO165,25^AD,36,20^FN2^FS
^FO25,75^AB,22,14^FDBUILT BY^FS
^FO25,125^AE,28,15^FN1
```

The character substitution described above must also be done on the file before it is stored in the "<update-root>/files" directory.

The `"recall"` format - using the `^XFE:STOREFMT.ZPL` command - does not need to be altered or edited. It can be used as it was previously.

Example Files

Example of files that have already been altered in the manner described above are available as "Mirror File Examples" at www.zebra.com.

To see an example file, right-click the paper-clip icon and select **Open File** or **Save Embedded File to Disk**.

File Naming Recommendations

Provides information on file naming recommendations

- Files in the `"<update-root>/files"` directory and `"<update-root>/commands"` directory should not have the same name.
- Files in the `"<update-root>/files"` should not contain multiple label formats. If you need to Mirror multiple formats, the recommended method is to split the formats into separate files.

Command Use Recommendations

Files in the`<update-root>/commands`directory should use only SGD commands or the following ZPL commands.

- `~CC`
- `~CD`
- `~CT`
- `~JA`
- `~JL`
- `~JS`
- `~JX`
- `~RO`

Do not add a`device.reset`SGD command to the end of a file in the `<update-root>/commands` directory. Mirror will reset itself automatically after performing an update, so there is no need for this command.

Configuration

This section provides detail on the configuring the printer and FTP server for Mirror.

Mirror FTP Server Configuration

For a Zebra printer to successfully use Mirror, the Mirror FTP server must have the following:

For a Zebra printer to successfully use Mirror, the Mirror FTP server must have the following:

- A client account (user name and password) for the printer to use
- A root (base) directory for Mirror updates. This root directory must have the following subdirectories:

```
/appl
```

```
/files
```

```
/commands
```

- A Mirror feedback folder (optional)

IMPORTANT: The printer's FTP user account must have the necessary permissions to read/write files in the update and feedback root directories. The `/appl`, `/files`, and `/commands` subdirectories are read only; the Mirror feedback folder is read/write. If these permissions are not properly set, the Mirror update and feedback processes will be unsuccessful.

Mirror Printer Configuration

The following SGD commands are used to configure Mirror on the printer.

- ip.mirror.auto
- ip.mirror.error_retry
- ip.mirror.feedback.auto
- ip.mirror.feedback.freq
- ip.mirror.feedback.odometer
- ip.mirror.feedback.path
- ip.mirror.fetch
- ip.mirror.freq
- ip.mirror.freq_hours
- ip.mirror.last_error
- ip.mirror.last_time

- ip.mirror.password
- ip.mirror.path
- ip.mirror.reset_delay
- ip.mirror.server
- ip.mirror.success
- ip.mirror.success_time on page 1283
- ip.mirror.username
- ip.mirror.version

The Feedback.get File

The Feedback feature is one of the key benefits of the Mirror process. During a Mirror operation, the printer can upload a file to the FTP Server that contains information about the configuration of the printer. This information can then be leveraged by the Administrator to monitor the printer's setup. Using the Feedback feature is optional.

The `"feedback.get"` file is a template file stored on the printer. It controls what content is uploaded to the`"<feedback-root>"`directory on the FTP server. The directory on the FTP server where the printer will send Feedback content to is controlled by the `"ip.mirror.feedback.path"` command.

Within the feedback.get file it is possible to leverage SGD commands to insert current printer status and configuration strings into the file. This feature can help make the Feedback file on the FTP server more unique and useful to the Administrator.

Additionally, the first line of the feedback.get file is used to control the name of the file that will be uploaded and stored on the FTP server.

For example, if first line of the `"feedback.get"` file was:

```
"zebra.<wlan.mac_raw>.<ip.mirror.feedback.odometer>"
```

That line would be evaluated by the printer and used as the Feedback destination file name to create on the FTP server.

Using the example above, if the MAC address of the wireless print server was "00a0f8ae56d7"and the Feedback odometer was currently at "33", the Feedback file created the FTP server would be named:

```
"zebra.00a0f8ae56d7.33.txt"
```

The "feedback.get" file can be sent to the printer using the `!CISDFCRC16` command. For additional information, see CISDFCRC16 Download Files on page 648.

NOTE: The first line of the feedback.get file is not included when the Feedback data is written to FTP server.

Example Feedback.get file

Here is an example of a "feedback.get" file and its resulting uploaded file:

```
zebra.<wlan.mac_raw>.<ip.mirror.feedback.odometer>
```

```
Application Name = <appl.name>
Serial Number = <device.friendly_name>

Mirror Success = <ip.mirror.success>
Mirror Auto = <ip.mirror.auto>
Mirror Path = <ip.mirror.path>
Mirror Last Update = <ip.mirror.success_time>

Bootp Enable = <ip.bootp.enable>
DHCP Enable = <ip.dhcp.enable>
Data Port = <ip.port>
Associated AP = <wlan.bssid>
RF ESSID = <wlan.essid>
RF Firmware = <wlan.firmware_version>
RF Signal Strength = <wlan.signal_strength>
RF Channel Mask = <wlan.channel_mask>

Label Length = <odometer.total_print_length>
Print Length = <odometer.label_dot_length>
When this file is processed by the printer and uploaded to the FTP server,
 the resulting upload file on the FTP server would contain data similar to
 this:
Application Name = V53.17.2Z
Serial Number = ZBR123456

Mirror Success = Yes
Mirror Auto = on
Mirror Path = /update-root/ZM400/
Mirror Last Update = 12345678

Bootp Enable = on
DHCP Enable = on
Data Port = 9100
Associated AP = 124
RF ESSID = MyEssid
RF Firmware = 5.2.1
RF Signal Strength = 98
RF Channel Mask = FF

Label Length = 100
Print Length = 200
```

How to Set Up and Use Mirror

This section provides multiple scenarios which include specific examples that demonstrate how to set up and use Mirror.

Scenario One

In this scenario, the printer is configured to perform a Mirror update operation ("`ip.mirror.auto = on`") and feedback operation ("`ip.mirror.feedback.auto = on`") every time the printer restarts.

On startup, after a network connection has been established, the printer will attempt to make a FTP connection to the server address 10.14.5.133, using the “user name” and “password” of the printer.

If the connection is successful, the printer will attempt to perform an update operation using the root directory "`/all_printers/s4m/role1`". After the update operation is complete, the printer will attempt a Feedback operation, uploading the resulting Feedback file to the "`/all_feedback`" directory on the server.

Using the command set in the example below, the printer will not attempt any periodic Mirror Update or Feedback operations. It will only perform Mirror operations on startup or when explicitly instructed to using the "`ip.mirror.fetch`" command.

NOTE: If a file starts with "/" it signifies the base directory of that file system. If a file is contained in the user’s account, they do not start with a "/".

This example shows a Mirror configuration command set. Each line item of the command set is identified with a number. For details on each line item, see the table below.

```
10 ! U1 SETVAR "ip.mirror.auto" "on"
20 ! U1 SETVAR "ip.mirror.username" "printer"
30 ! U1 SETVAR "ip.mirror.password" "printer"
40 ! U1 SETVAR "ip.mirror.server" "10.14.5.133"
50 ! U1 SETVAR "ip.mirror.path" "/all_printers/s4m/role1"
60 ! U1 SETVAR "ip.mirror.feedback.auto" "on"
70 ! U1 SETVAR "ip.mirror.feedback.path" "/all_feedback"
80 ! U1 SETVAR "ip.mirror.feedback.freq" "0"
```

10	Configures the printer to perform a Mirror Update operation at power-up.
20	Configures the FTP Server "user name" for the printer to use
30	Configures the FTP Server "password" for the printer to use
40	Configures the FTP server address the printer should make a FTP connection to.
50	If the FTP connection is successful, the printer should attempt to perform an Update operation using this root directory.
60	Configures a printer to automatically perform a Mirror Feedback operation at start-up.
70	Configures the printer to upload the resulting Feedback file to the designated directory on the server.
80	Configures a printer to repeat the Feedback operation zero times.

Scenario Two

In this scenario, the printer is configured to not perform a Mirror Update function at start-up. It is configured to perform a Mirror Feedback operation at start-up and thereafter at every 60 minutes.

When the 60 minutes elapses, the printer will attempt to make a FTP connection to the server address 10.14.5.133. If the FTP connection is successful, the printer will attempt a Feedback operation, uploading the resulting Feedback file to the "/all_feedback" directory on the server. After the initial Feedback operation, subsequent Feedback operations will occur at an interval of 60 minutes.

IMPORTANT: Using the command set in the example below, for the printer to attempt any Mirror Update operation unless the "`ip.mirror.fetch`" command is sent to the printer.

This example shows a Mirror configuration command set. Each line item of the command set is identified with a number. For details on each line item, see the table below.

```
10 ! U1 SETVAR "ip.mirror.auto" "off"
20 ! U1 SETVAR "ip.mirror.username" "printer"
30 ! U1 SETVAR "ip.mirror.password" "printer"
40 ! U1 SETVAR "ip.mirror.server" "10.14.5.133"
50 ! U1 SETVAR "ip.mirror.path" "/all_printers/s4m/role1"
60 ! U1 SETVAR "ip.mirror.feedback.auto" "on"
70 ! U1 SETVAR "ip.mirror.feedback.path" "/all_feedback"
80 ! U1 SETVAR "ip.mirror.feedback.freq" "60"
```

10	Configures the printer to not perform a Mirror Update operation at start-up
20	Configures the FTP Server "user name" for the printer to use
30	Configures the FTP Server "password" for the printer to use
40	Configures the FTP server address the printer should make a FTP connection to.
50	If the FTP connection is successful, the printer should attempt to perform an Update operation using this root directory.
60	Configures a printer to automatically perform a Mirror Feedback operation at start-up.
70	Configures the printer to upload the resulting Feedback file to the designated directory on the server.
80	Configures the printer to attempt a Feedback operation every 60 minutes.

Troubleshooting

If a Mirror process completes unsuccessfully, troubleshooting information can be retrieved by sending this command to the printer:

```
! U1 GETVAR "ip.mirror.last_error"
```

Table 41 Printer Response Troubleshooting on page 1690 lists possible printer responses, an explanation of each, and resolutions. Table 42 Problem Scenario Troubleshooting on page 1692 provides problem scenarios and solutions.

IMPORTANT: A mirror path can have up to 50 characters.

Table 41 Printer Response Troubleshooting

Printer Response	Explanation	Resolution
`"connection failed"`	The network connection to the Mirror FTP server failed while attempting to perform a printer update.	• Check the user name, password, and server address for the Mirror FTP server and ensure that these values are set correctly in the printer. • Ensure that the user name assigned to the printer has the proper permission to log into the Mirror FTP server. • Check that the printer has a successful network connection and is able to send and receive network data.
`"Failed to get File: [filename]"`	During an update operation, the printer's attempt to retrieve the file [filename] failed.	• Ensure that the printer's network connection has not been interrupted. If it has, re-establish network connectivity and retry the update. • Check the server's access permissions for the user name assigned to the printer. Make sure the user name is granted access to read [filename] from the server.
`"feedback connection failed"`	The network connection to the Mirror FTP server failed while attempting to send printer feedback.	• Check the user name, password, and server address for the Mirror FTP server and ensure that these values are set correctly in the printer. • Ensure that the user name assigned to the printer have the proper permission to log into the Mirror FTP server. • Check that the printer has a successful network connection and is able to send and receive network data.

Table 41 Printer Response Troubleshooting (Continued)

Printer Response	Explanation	Resolution
`"Failed getting file to parser : [filename]"`	During an update operation, the printer's attempt to retrieve a file [filename] in the `<mirror_path>/appl` directory failed.	• Ensure that the printer's network connection has not been interrupted. If it has, re-establish network connectivity and retry the update. • Check the server's access permissions for the user name assigned to the printer. Make sure the user name is granted access to read `[filename]` from the server.
`"Failed to send feedback file: <feedback_path>/ <feedback_filename>"`	During a feedback operation, the printer's attempt to store the feedback file in the `<feedback_path>` directory failed.	• Ensure that the printer's network connection has not been interrupted. If it has, re-establish network connectivity and retry the update. • Check the server's access permissions for the user name assigned to the printer. Make sure the user name is granted access to write to the `<feedback_path>` directory. • Ensure that the `<feedback_path>` directory exists on the remote server.
`"Too many files in the firmware download directory"`	The Mirror FTP server has more than one file in the `<mirror_path>/appl` directory.	Ensure that there is only one (1) firmware file in the `<mirror_path>/appl` directory.

Table 42 Problem Scenario Troubleshooting

Problem Scenario	Solution
I performed a Mirror Update and now my printer is continuously reprogramming.	Ensure that the name of the firmware file in `<mirror_path>/appl` matches the version of firmware contained in that file. For firmware version V53.17.2Z, the name of the file in the `<mirror_path>/appl` directory must be `V53.17.2Z.ZPL` to prevent the continuous reprogramming cycle.
Every time a Mirror Update is run, a file is fetched even though no changes have been made to the files on the server.	• Check the names of the files in the `<mirror_path>/files` and `<mirror_path>/commands` directories. If the names are longer than 16 characters (minus extensions), then the printer will truncate them to 16 characters when downloading. If two filenames truncate to the same 16 characters, the printer will not be able to tell the difference between them and will re-download one of the two files during every update operation. • Check the names of the files in the `<mirror_path>/files` and `<mirror_path>/commands` directories. If any of the names are the same, then the printer will not be able to tell the difference. Therefore it will get the one in the files directory, update the timestamp file, and then it will get the one in the commands directory and update the timestamp file. Then, the next time through the one in the files directory will have a different timestamp, so it will get that file again and then check the commands directory, and so on. • Check the server's access permissions for the user name assigned to the printer. Make sure the user name is granted access to read all files in the `<mirror_path>/files` and `<mirror_path>/commands` directories.
The printer is continually rebooting.	• Check the `/commands` directory. If a `file.delete` command resides, then you need to remove `file.delete` or modify your script.

Wireless Markup Language (WML)

Wireless Markup Language (WML) offers a text-based method of designing a menu structure for the display screen of selected printers. By leveraging Set/Get/Do (SGD) and files containing Zebra Programming Language (ZPL) commands, customized menus can be created.

WML Overview

Wireless Markup Language (WML) offers a text-based method of designing customized menus on the LCD front panel of selected printers.

By leveraging Set-Get-Do (SGD) and Zebra Programming Language (ZPL) commands, menus that feature both display and command features can be created. The WML “card” structure makes it possible to link from one menu screen to another, creating menus that are as many levels “deep” as desired or reduced to only those options needed by the printer operator.

For details on SGD commands, see SGD Command Support. For details on ZPL commands, see ZPL Commands.

WML Details

A WML file is made up of tags, which are similar to HTML tags. For a list of the supported WML tags, see WML Tags.

Using WML on the printer is dependent on the presence of a single `index.wml` file, stored in the printer's E: memory. The `index.wml` file can contain one or more “cards”, with each card defining the content of a single menu. Everything within the card tag (<card> </card>) constitutes one complete front panel menu. Cards can also contain hyperlinks to other menus. If the index.wml has three cards, with links between the cards, that means there are three front panel menus. It is also possible to create multiple .wml files, with links between them and the index.wml file. In cases where multiple .wml files are used, it is recommended that each file should be structured to provide a link back to the main menu as described in the index.wml card.

NOTE: Only one `index.wml` file can reside on a printer at any time

WML defined menus can use Set-Get-Do (SGD) commands to retrieve or set printer settings. For example, a menu might display the printer’s current baud rate, while also offering other potential baud rate settings for the printer selection. In more advanced uses, WML defined menus can cause ZPL command files, stored in the printer E: memory, to be injected into the printers command engine – where they will be read in and acted upon. In this use, the ZPL command file files are known as .nrd files.

For example, a WML defined menu could call an .nrd file that contains a customized set of printer configuration commands. In this way, different profiles can be created for the printer - making it possible for the printer operator to select the appropriate configuration profile needed for the task the printer is being used in.

An important concept to consider is that the WML menu completely defines what is displayed on the printers screen. If an item is not included in the WML menu definition it will not be displayed to the user.

NOTE: The `index.wml` file must reside on the printer's `E:` drive for the WML menu to display. If the `index.wml` file is on a drive other than `E:`, then the standard front panel menus display. 3

NOTE: When a WML menu is resident on the printer, the standard menu system can be easily be accessed by holding down the Cancel and Setup/Exit buttons (on the ZM400) or the Cancel and Setup/Exit buttons (on Xi4) or the Select button (on GX) on the front panel while the printer powers up. Hold the buttons down until the PRINT READY message displays on the front panel. To return to the WML defined menu, reset the printer again.3

Supported Printers

WML is supported on the following printers, using the indicated firmware. The buttons on the printers' front panel that are used for Navigating WML defined menus are noted.

NOTE: When a WML defined menu is in use, the stripes pattern found at the top of selected printers is not displayed.

Table 43 WML-Supported Printers

Printer	Firmware	Number of "lines" available	Menu Navigation Buttons	Keys to Access Standard Menu System
105SLPlus	V53.17.15Z (or later)	5	Select + (PLUS) - (MINUS)	Hold down CANCEL & SETUP/EXIT during power-up
Xi4 series	V53.17.5Z (or later)	5	Select + (PLUS) - (MINUS)	Hold down CANCEL & SETUP/EXIT during power-up
ZE500	V53.17.15Z (or later)	5	Select + (PLUS) - (MINUS)	Hold down CANCEL & SETUP/EXIT during power-up
ZM400	V53.17.5Z (or later)	5	Select + (PLUS) - (MINUS)	Hold down CANCEL & SETUP/EXIT during power-up
ZM600	V53.17.5Z (or later)	5	NEXT/SAVE + (PLUS) - (MINUS)	Hold down CANCEL & SETUP/EXIT during power-up
G series	V53.17.5Z (or later)	4	SELECT SCROLL	Hold down SELECT during power-up

Professional Services for WML Content Creation

Zebra offers a Professional Services group that can help with the creation of WML content. To inquire about Zebra's Professional Services, contact your Zebra account representative.

WML Tags

This section lists the WML tags and tag parameters that can be used to create a menu system.

Table 1 shows the WML tags and tag parameters that can be used to create a menu system. As with other tag-based languages, such as HTML and XML, ending tags should be used to indicate the end of a structure. An example of an ending tag would be </wml>, which indicates the end of a WML script.

IMPORTANT: Using end tags is required to create well formed and functional WML scripts.

Table 44 WML Tag Descriptions

`<wml> </wml>`	indicates the beginning/end of the WML script
`<display> </display>`	indicates the beginning/end of the content to display on-screen
`<card> </card>`	indicates the beginning/end of a card
`<p> </p>`	indicates the beginning/end of a Paragraph
` `	Line break
`<a href="#menu">Menu</a>`	Hyperlink to another card
`<timer value="xx"> </timer>`	Controls display timer in 10th of a second increments
`" ontimer="#main"`	Controls action to take at timer end
`alerts="on"`	Controls display of on-screen alerts
`$(command.command)`	$ executes a SGD "get" command
`<do .....><setvar .......></do>`	Controls execution of do and setvar commands

Using WML

This section provides you with the necessary steps to prepare and transmit WML content to the printer. There are two methods to send WML content to the printer - via the FTP protocol or using the "CISDFCRC16" command. Both methods are detailed below.

IMPORTANT: The & (ampersand) character should not be used within the body of any Paragraph tag (<p>). If an ampersand is present within the body of a Paragraph tag, a WML-based menu may not function as expected. The ampersand character should NEVER be used within a paragraph tag for any of the printer's soft keys (P1, P2, etc.); doing so can render the menu inoperable.

Create a Sample index.wml File

This procedure shows how to create an index.xml file.

1. Open a text editor.

2. Type (or copy/paste) the following text:

```
<wml>
<display>
<card>
<p>Hello World!!</p>
</card>
</display>
</wml>
```

   Save this file with this name: `index.wml`.

3. To confirm these commands are correctly set, send the `getvar` command to check the settings. To do this, send these commands to the printer:

Prepare the Printer to Receive WML Content via FTP

NEED

WML files – and any .nrd files used by a WML menu structure – must be stored in the printers E: memory location. While the files are first being transmitted to the printer, they should not be processed by the printers ZPL formatting engine. This can be done by configuring the SGD settings `"ip.ftp.enable"` and `"ip.ftp.execute_file"`.

The `ip.ftp.enable"` setting allows the printer to receive content via the FTP protocol. The `“ip.ftp.execute_file”` setting controls the printers’ ability to process or not process commands received via the FTP protocol using the printers ZPL engine. By default, both settings are enabled.

Set `"ip.ftp.enable"` to `"on"` and `"ip.ftp.execute_file"` to `"off"`.

To do this, send these commands to the printer:

```
! U1 setvar "ip.ftp.enable" "on"
! U1 setvar "ip.ftp.execute_file" "off"
! U1 getvar "ip.ftp.enable"
! U1 getvar "ip.ftp.execute_file"
```

To confirm these commands are correctly set, send the `getvar` command to check the settings. To do this, send these commands to the printer:

```
! U1 getvar "ip.ftp.enable"
! U1 getvar "ip.ftp.execute_file"
```

If a terminal emulation program is being used, the following response should be returned from the printer.

```
"on""off"
```

NOTE: Only printers using the Internal 10/100 wired or Internal Wireless Plus & Wireless Plus print server can use the `! U1 setvar "ip.ftp.execute_file" "off"` command. For other print servers, use the `“CISDFCRC16”` command method detailed below.

Send WML Content to the Printer via FTP

Use these steps to send WML content to the printer via FTP.

Go to a command prompt.

At the command line prompt, type `"ftp xxx.xxx.xxx.xxx"`, where `xxx.xxx.xxx.xxx` is the IP Address of the printer. For example, if the IP Address of the printer is 10.3.5.34, the command would be:

```
ftp 10.3.5.34
```

1. Press Enter to connect to the printer.
2. Press Enter to log in to the printer.
3. At the FTP prompt, type "put index.wml" and press Enter. The index.wml file will be transferred to the printer's E: memory.
4. Type "quit" to disconnect from the printer and exit FTP.
5. Power cycle the printer.

   Once the printer completes the power cycle the display should look similar to this:

   For additional index.wml examples, see WML Examples.

NOTE: When a WML menu is resident on the printer, the standard menu system can be easily be accessed by holding down the Cancel and Setup/Exit buttons (on the ZM400) or the Cancel and Setup/Exit buttons (on Xi4) or the Select button (on GX) on the front panel while the printer powers up. Hold the buttons down until the PRINT READY message displays on the front panel. To return to the WML defined menu, reset the printer again.

IMPORTANT: When using the `"ip.ftp.execute_file"` command, be sure to reset the command back to `"on"` for use in production processes. If the setting is left in the `"off"` configuration, when label formats or firmware are sent to the printer via FTP they will not be processed as intended – and the E: memory location can quickly become full.

Resetting the ip.ftp.execute_file Setting

Use this procedure to reset the `ip.ftp.execute_file` setting.

To reset the `"ip.ftp.execute_file"` setting to the default state, send the following command to the printer.

```
! U1 setvar "ip.ftp.execute_file" "on"
```

Sending WML Content to the Printer via the CISDFCRC16 Command

Use these step to send WML content to the printer via the CISDFCRC16 command.

WML files – and any .nrd files used by a WML menu structure – must be stored in the printers E: memory location. While the files are first being transmitted to the printer, they should not be processed by the printers ZPL formatting engine. This can be done by using the CISDFCRC16 command. This command allows content to be written directly to the E: memory location, without being processed by the printers ZPL formatting engine. By using the CISDFCRC16 command, WML content can be transmitted to the printer via the Serial, USB or Parallel ports.

For additional information on the CISDFCRC16 command, see .

1. To send the sample `index.wml` shown earlier, send the following commands to the printer:

```
! CISDFCRC16
0000
INDEX.WML
0000004E
0000
<wml>
<display>
<card>
<p>Hello World!!</p>
</card>
</display>
</wml>
```

2. Power cycle the printer.

   Once the printer completes the power cycle the display should look similar to this:

   For additional index.wml examples, see WML Examples .

NOTE: When a WML menu is resident on the printer, the standard menu system can be easily be accessed by holding down the Cancel and Setup/Exit buttons (on the ZM400) or Cancel and Setup/Exit buttons (on Xi4) on the front panel while the printer powers up. Hold the buttons down until the PRINT READY message displays on the front panel. To return to the WML defined menu, reset the printer again.

Retrieving WML Content from the Printer using the file.type Command

Use this procedure to retrieve content from the printer using the file.type command.

It is possible to retrieve .wml file content from the printer using the `"file.type"` SGD command. To do this, open a terminal emulation connection to the printer and issue the command. For example, to retrieve the contents of the INDEX.WML file, use the following command:

```
! U1 setvar "file.type" "E:INDEX.WML"
```

NOTE: The `file.type` command is case sensitive – if the file is stored on the printer as `INDEX.WML`, the command must use that same case. Additionally, you should note that `.nrd` files are treated as confidential – they cannot be retrieved from the printer.

Using .nrd Files from WML Menus

This section provides information on using .nrd files.

It is possible to have a WML menu send the contents of a ZPL or SGD file to the printer to be processed. In this way, the WML menu can leverage complex command scripts in response to the user pressing a single button on the printer. One possible use for this capability would be to create a series of “profile” files that contain all the settings necessary to reconfigure the printer for different uses.

The command files are known as “.nrd” files and are stored directly on the printers E: memory location. The .nrd files can be created using a standard text editor and sent to the printer via FTP or using the CISDFCRC16 command. Files should have an “.nrd” extension.

Removing WML or .nrd Files from the Printer using the file.delete Command

Use this procedure to delete WML or .nrd filers from the printer using the file.delete command.

It is possible to remove .wml files from the printer using the file.delete SGD command. To do this, open a terminal emulation connection to the printer and issue the command. For example, to remove the `INDEX.WML` file, use the following command:

```
! U1 do "file.delete" "E:INDEX.WML"
```

WML Examples

The examples shown below “build” from a simple, display-only, WML menu to a more complex interactive example that uses .nrd files containing ZPL commands. In the initial examples, all lines are explained in detail, in the later examples only the new concepts are covered in detail.

Indenting is used in the examples below to improve readability, it is not necessary in actual use.

Example 1

This example shows a basic WML menu structure that uses only fixed text. The content below shows the WML script plus numbered callouts and a table that identify the function of each of the WML tags.

```
1 →    <wml>
2 →    <display>
3 →       <card>
4 →       <p>Hello World!!</p>
5 →       </card>
6 →    </display>
7 →    </wml>
```

1	Beginning of the WML file.
2	Beginning of the content to be displayed.
3	The <card> tag begins the definition of this menu.
4	The beings a paragraph, here displaying Hello World! The ends the paragraph.
5	The </card> tag ends the definition of this menu.
6	End of the content to be displayed.
7	End of the WML file.

In use, this WML menu looks similar to this:

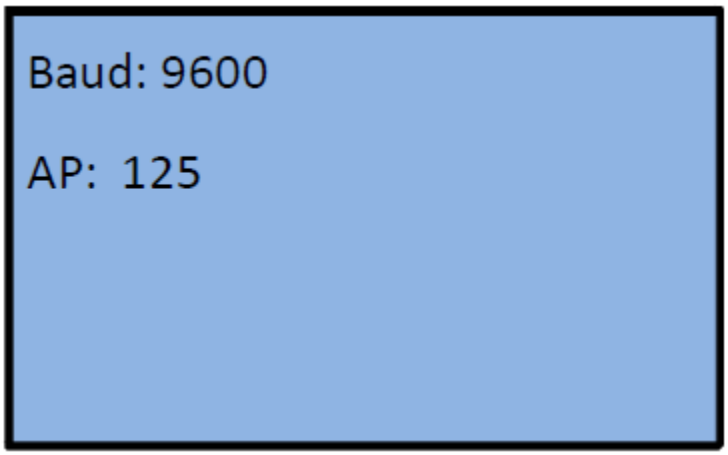
Example 2

This example demonstrates a WML menu structure that uses fixed text, plus two SGD commands to display the current printer settings for the Baud rate and ESSID settings on the printer.

```
1 →     <wml>
2 →     <display>
3 →     <card>
4 →     <p>Baud: $(comm.baud)</p>
5 →     <br/>
6 →     <p>AP: $(wlan.essid)</p>
7 →     </card>
8 →     </display>
9 →     </wml>
```

1	Beginning of the WML file.
2	Beginning of the content to be displayed.
3	The <card> tag begins the definition of this menu.
4	The begins a paragraph.‘Baud:’ displays the text Baud:$(comm.baud) retrieves and displays the printers’ current baud rate. The ends the paragraph.
5	A line break
6	The begins a paragraph.“AP:” displays the text AP:$(wlan.essid) retrieves and displays the printers’ current ESSID setting. The ends the paragraph.
7	The </card> tag ends the definition of this menu.
8	End of the content to be displayed.
9	End of the WML file

In use, this WML menu looks similar to this:



Example 3

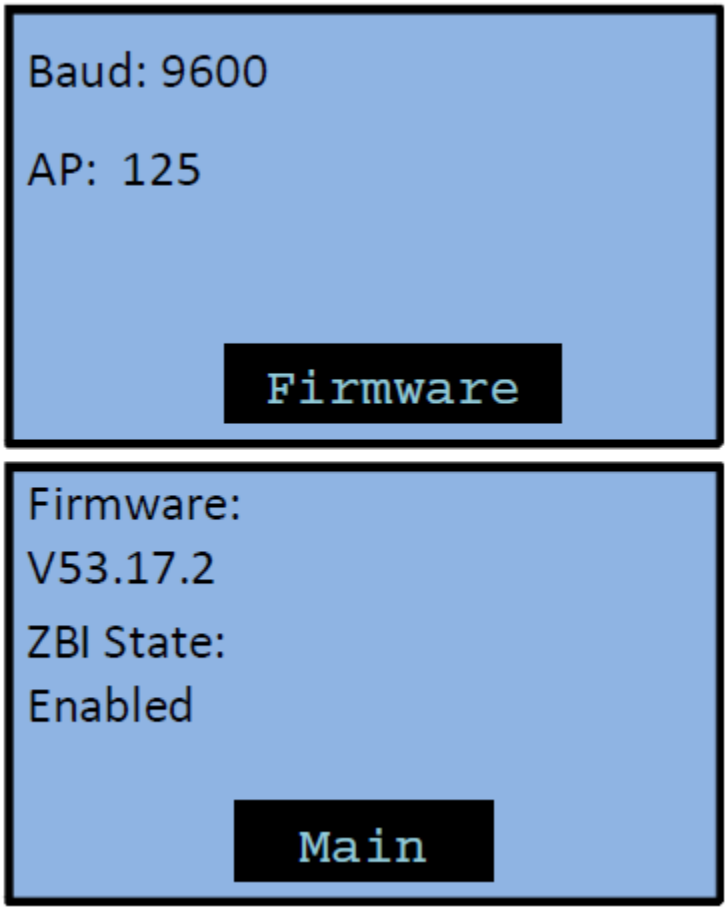
This example demonstrates a WML menu structure with two menus.

Fixed text and SGD commands are used to display the current printer settings for the Baud rate and ESSID settings on menu one and the Firmware version and ZBI State on menu two. Through use of the 'timer' setting, the menu will automatically return to a defined WML card if no buttons are pressed after a set time period. The menu is configured to allow printer alerts (such as HEAD OPEN) to be displayed.

```
<wml>
<display>
  <card id="main" title="" ontimer="#main" alerts="on">
  <timer value="50"></timer>
   <p>Baud: $(comm.baud)</p>
   <br/>
   <p>AP: $(wlan.essid)</p>
   <p> </p><br/>
   <p> </p><br/>
  <p> <a href="#system">Firmware</a></p>
  </card>
   <card id="system" title="" ontimer="#main" alerts="on">
   <timer value="50"></timer>
   <p>Firmware:</p><br/>
   <p>$(appl.name)</p><br/>
   <p>ZBI State:</p><br/>
   <p>$(zbi.key)</p><br/>
   <p> <a href="#main">Main</a></p>
   </card>
</display>
</wml>
```

3	<card id="main" – defines the card's id – "main".title="" – defines the title (not displayed on screen).ontimer="#main" – defines the WML card to display when the timer runs out.alerts="on"> – enables the alerts display feature.
4	<timer value="50"></timer> – sets the timer to 50 (in 10th of a second increments).
10	Firmware – defines a link to the "system" card.
12	<card id="system" – defines the card's id – "system".title="" – defines the title (not displayed on screen).ontimer="#main" – defines the WML card to display when the timer runs out.alerts="on"> – enables the alerts display feature.
13	<timer value="50"></timer> – sets the timer to 50 (in 10th of a second increments).
18	Main – defines a link to the "main" card.

In use, these two WML menus look similar to this:



NOTE: GX series printers can display four lines of text. If you are using a GX series printer, remove one line of text from each "card" to use this example.

Example 4

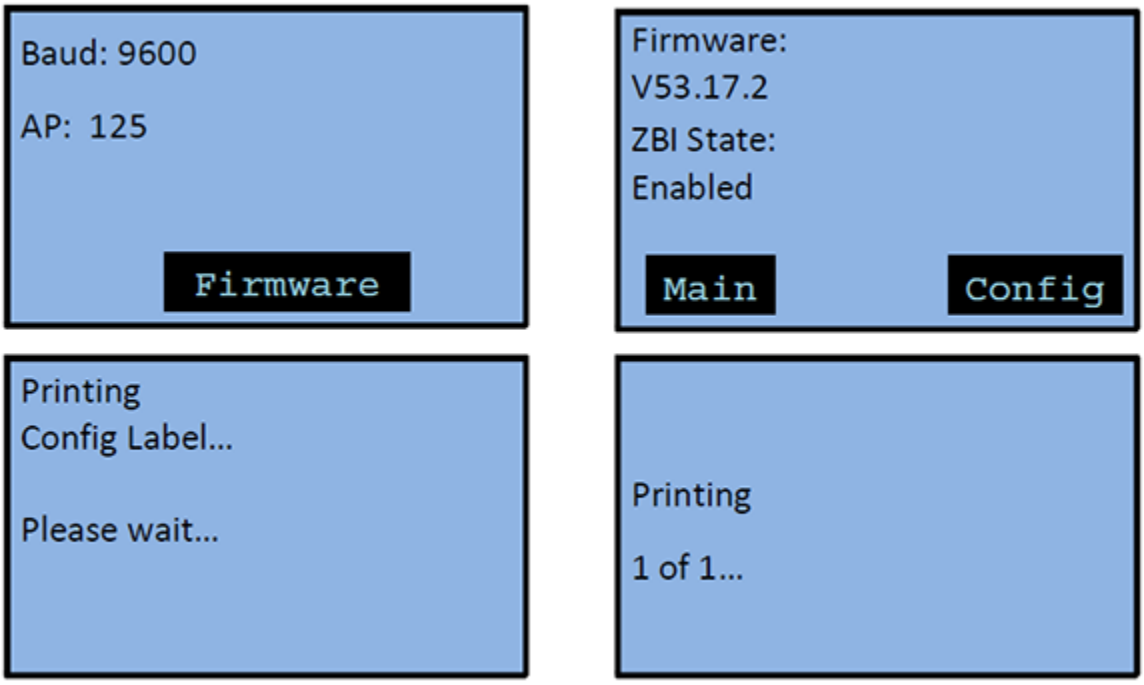
This example demonstrates a WML menu structure that creates two menu screens and a link to a command file – “config.nrd” – that contains a ZPL command that will cause the unit to print a configuration label.

```
1→      <wml>
2→      <display>
3→      <card id="main" title="" ontimer="#main" alerts="on">
4→      <timer value="50"></timer>
5→      <p>Baud: $(comm.baud)</p>
6→      <br/>
7→      <p>AP: $(wlan.essid)</p>
8→      <p> </p><br/>
9→      <p> </p><br/>
10→     <p> <a href="#system">Firmware</a></p>
11→     </card>
12→     <card id="system" title="" ontimer="#main" alerts="on">
13→     <timer value="50"></timer>
14→     <p>Firmware:</p><br/>
15→     <p>$(appl.name)</p><br/>
16→     <p>ZBI State:</p><br/>
17→     <p>$(zbi.key)</p><br/>
18→     <p><a href="#main">Main</a>
19→     <a href="#config">Config</a></p>
20→     </card>
21→     <card id="config" title="" ontimer="#main" alerts="on">
22→     <timer value="50"></timer>
23→     <p>Printing </p><br/>
24→     <p> Config Label...</p><br/>
25→     <p></p><br/>
26→     <p>Please wait...</p><br/>
27→     <setvar name="file.run" value="e:config.nrd"/>
28→     </card>
29→     </display>
30→     </wml>
```

18
19
	`<p><a href="#main">Main</a>`
	`<a href="#config">Config</a></p>`
	• Defines two links, positioned next to each other - to the "main" and "config" WML cards

27	`<setvar name="file.run" value="e:config.nrd"/>` • Defines that the SGD command `"file.run"` should be used on the `"e:config.nrd"` file. • In this instance, the `"e:config.nrd"` file contains a single ZPL command - `"~wc"`.

In use, these WML menus look similar to this:



NOTE: GX series printers can display four lines of text. If you are using a GX series printer, remove one line of text from each “card” to use this example.

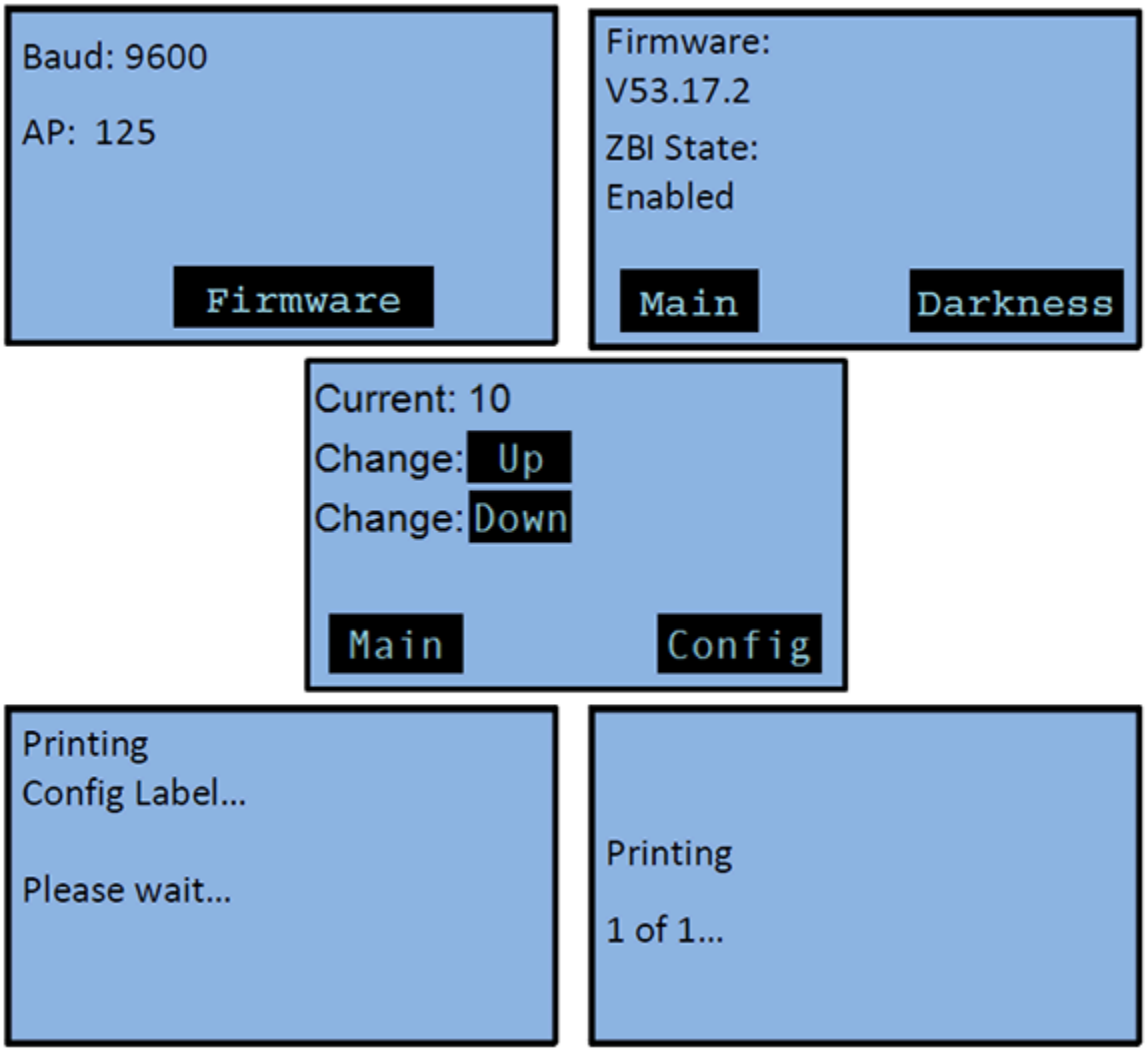
Example 5

This example demonstrates a WML menu structure with three cards. The "darkness" card leverages WML and the SGD "print.tone" command to allow the user to both view and configure a setting.

```
<wml>
<display>
<card id="main" title="" ontimer="#main" alerts="on">
<timer value="50"></timer>
<p>Baud: $(comm.baud)</p>
<br/>
<p>AP: $(wlan.essid)</p>
<p> </p><br/>
<p> </p><br/>
<p> <a href="#system">Firmware</a></p>
</card>
<card id="system" title="" ontimer="#main" alerts="on">
<timer value="50"></timer>
<p>Firmware:</p><br/>
<p>$(appl.name)</p><br/>
<p>ZBI State:</p><br/>
<p>$(zbi.key)</p><br/>
<p><a href="#main">Main</a> <a href="#darkness">Darkness</a></p>
</card>
<card id="darkness" title="" ontimer="#main" alerts="on">
<timer value="50"></timer>
<p>Current: $(print.tone)</p><br/>
<p>Change: </p><do type="accept" label="Up"><setvar
name="print.tone" value="+1.0"/></do><br/>
<p>Change: </p><do type="accept" label="Down"><setvar
name="print.tone" value="-1.0"/></do><br/>
<p> </p><br/>
<p><a href="#main">Main</a> <a href="#config">Config</a></p>
</card>
<card id="config" title="" ontimer="#main" alerts="on">
<timer value="50"></timer>
<p>Printing </p><br/>
<p> Config Label...</p><br/>
<p></p><br/>
<p>Please wait...</p><br/>
<setvar name="file.run" value="e:config.nrd"/>
</card>
</display>
</wml>
```

23	`<p>Change: </p>` • Defines the fixed text `"Change: "` `<do type="accept" label="Up"><setvar name="print.tone" value="+1.0"/></do> ` • Defines selecting the word "Up" as equal to sending the value "+1.0" for the SGD command "print.tone". In this case, this increases the setting by 1.0.
24	`<p>Change: </p>` • Defines the fixed text `"Change: "` `do type="accept" label="Down"><setvar name="print.tone" value="-1.0"/></do> ` • Defines selecting the word "Down" as equal to sending the value "-1.0" for the SGD command "print.tone". In this case, this decreases the setting by 1.0.

In use, these WML menus look similar to this:



NOTE: GX series printers can display four lines of text. If you are using a GX series printer, remove one line of text from each “card” to use this example.

Troubleshooting Scenarios

Problem Scenario	Corrective Actions
I loaded a WML menu structure on the printer, but the Factory menu structure is displaying.	• The WML files may have syntax errors. Reconfirm that the correct syntax has been used. When creating WML files it is recommended to start with a simple structure, validate that it's functional and build additional content onto the "known good" example. • Power cycle the printer and watch the start-up sequence - if a "WML ERROR" message displays during the start-up, the `index.wml` file has a syntax error that needs to be corrected. • The `index.wml` file may not have been successfully transferred to the printer. Use a terminal emulation program and the following command to retrieve the `index.wml` file to the PC for examination: `! U1 setvar "file.type" "E:INDEX.WML"` • The `index.wml` file may not be present in the E: memory location. Validate that the file is present and correctly named. • Confirm that straight quotes were used in all instances where the quote character was used (use the " character - not " or ?). SGD commands require the use of the straight quote. • Confirm that the "WML Menu Cancel" buttons, (Setup and Cancel or Cancel and Setup/Exit or Select) were not held down during start up. These actions will cause the standard menu to display.
Some characters in the menu are cut off or some lines are not displaying at all.	• Characters that extend past the width of the display are truncated, reposition the field as needed. • Check that you have not exceeded that maximum number of lines the display allows (5 lines on ZM and Xi4 series units, 4 lines on the GX series).
My WML menu structure is displaying, but one of the “cards” is not displaying or is unreachable.	• The missing "card" may not have been linked to from any of the visible "cards". Review you WML content to ensure that the correct links exist. • The WML files may have syntax errors, reconfirm that the correct syntax has been used. When creating WML files it is recommended to start with a simple structure, validate that it’s functional and build additional content onto a known good example. • If the missing card content is contained in a separate `.wml file`, confirm that the necessary `.wml` files have been transferred to the printer.

Problem Scenario	Corrective Actions
My WML menu structure uses SGD commands to display current settings, but the settings are not displaying.	• Validate that the correct syntax was used for the SGD command. • Check the manual page for the command being used. Confirm that the command is supported by the printer & firmware. Use a terminal emulation program to send just the command being used to validate it functions outside the WML menu structure. • Confirm that straight quotes were used in all instances where the quote character was used (use the " character – not “ or ˝). SGD commands require the use of the straight quote. • Characters that extend past the width of the display are truncated, reposition the field as needed. • Check that you have not exceeded that maximum number of lines the display allows (5 on ZM and Xi4 series units).
My WML menu structure used SGD commands to allow the user to alter printer settings, but the settings are not getting changed as expected.	• Validate that the value being used in the value= parameter of the <do> tag is supported by the SGD command. • Validate that the correct syntax was used for the SGD command. • Check the manual page for the command being used. Confirm that the command is supported by the printer & firmware. Use a terminal emulation program to send just the command being used to validate it functions outside the WML menu structure. • Confirm that straight quotes were used in all instances where the quote character was used (use the " character – not “ or ˝). SGD commands require the use of the straight quote.
My WML menu structure uses .nrd files to send commands to the printers ZPL or SGD engine, but the commands don’t seem to be getting sent when the user selects the on-screen link for the action.	• Confirm that the .nrd files are present in E: memory and named as expected. Resend or rename the files if necessary. • Confirm that the WML menu structure is using the correct file name(s). • Confirm that the commands in the files work as expected, independently of the WML menu or `.nrd` file. • Validate that ZPL and SGD commands have not been interlaced.
I am using the `CISDFCRC16` command to transfer files, but the files are either not being transferred to the printer or are showing up with a zero (0) byte size.	• Confirm that the Hexadecimal value used for the File Size parameter is correct. This value must be an eight digit file size specified in hexadecimal which indicates the number of bytes in the <data> section of the command. See the full manual page on the CISDFCRC16 command for additional details. • Validate that the CRC and Checksum parameters are correct (using the "0000" value for these parameters is recommended). • Confirm that the WML file name and extension are in upper case characters. • Confirm that the exclamation mark (!) was included before the command name ("`! CISDFCRC16`").

Using Weblink

Weblink is a feature of Zebra Link-OS™ printers.

Using a secure connection, the Weblink feature allows the printer to directly connect to an internet based server, for the purpose of either sending information to the server or receiving from the server. Weblink can transport data securely through a firewall.

When Should Weblink be Used?

Weblink can transport any information related to device management, transactional data and information to be processed at a later time. It can be used as part of an overall cost reduction solution that leverages web technologies.

Typically, an application called a 'servlet' is created and run on the internet based server, waiting for printers to connect and interact with the servlet. These servlet applications can provide a variety of functions – from sending operating system updates to the printer, to receiving data from the printer and, in turn, using that data to trigger events in other systems.

For example, a solution could be created that would feature the printer consuming data from a Bluetooth® scanner connected to the printer – with that scanned data then being sent from the printer to the internet-based servlet. The servlet would then seek out additional details related to the scanned data, format a document, and then send it to the printer for printing.

Configuring Weblink

When any WebLink setting (with the exception of the logging settings) is adjusted either via SNMP, SGD, or JSON it is required that the printer be reset before the new value takes effect.

The `weblink.printer_reset_required` setting will be set to `"yes"` if there are any settings that have been modified that require a printer reset.

Basic Configuration

To determine how much configuration is necessary the following questions should be considered:

1. Is the remote server the printer is attempting to connect to outside the corporate firewall?
2. Does the firewall require a username and password to access the remote server?
3. Does the printer require a proxy server to access the remote server?

4. Does the firewall permit HTTPS connections initially or does the printer need to connect via HTTP first?

   If the answer to any of these questions is 'yes', then more than the basic configuration may be necessary. Depending upon the network environment that the printer is in, accessing the remote server may only require that a few settings be set.

   The minimum requirement is that the URL for the remote server be set. For simplicity, assume that only conn1 is being used (this is the typical scenario). See also Difference Between Conn1 and Conn2.

   To configure the printer to connect to the remote server:

   i. Set `weblink.ip.conn1.location` to the URL of the remote server.

   The URL must conform to the standards described in RFC3986 (http://www.ietf.org/rfc/rfc3986.txt). For example, if the remote servlet's full URL is

   `https://www.examplecorpinc.com/zebra/weblink/`

   Configure the location setting as follows:

```
! U1 setvar "weblink.ip.conn1.location"
             "https://www.examplecorpinc.com/zebra/weblink/"
```

   ii. Reset the printer.

   When the printer has an IP address, it will attempt to connect to the remote server. In the event that the remote server does not indicate that the printer has connected, logging may need to be enabled in order to determine the failure.

When a Proxy Server is Part of the Network Configuration

If a proxy server must be used to access the remote server, the printer's proxy setting must be set to connect to the server.

There are typically four properties associated with a proxy server:

- The proxy server scheme: HTTP or HTTPS
- The proxy server address
- The proxy server port (optional)
- The username and password for the proxy (optional)

To supply the address of the proxy server (assuming a default port and no username/password), configure the proxy setting as follows:

```
! U1 setvar "weblink.ip.conn1.proxy" "https://my.internal.proxy/"
```

In this scenario the proxy address is `my.internal.proxy` and the scheme is HTTPS. The default port (1080) will be used. No username or password will be used to authenticate with the proxy.

To specify an alternate port configure the proxy as follows:

```
! U1 setvar "weblink.ip.conn1.proxy" "https://my.internal.proxy:3128/"
```

To specify a username and password configure the proxy as follows:

```
! U1 setvar "weblink.ip.conn1.proxy" "https://user:pass@my.internal.proxy/"
```

The proxy username, password, and the rest of the URL must follow the rules specified in RFC3986 (http://www.ietf.org/rfc/rfc3986.txt).

When HTTP Authentication is Necessary

Use this configuration when, for example, a firewall requires a username and/or password.

It may be necessary to specify a username and password to various routers and servers along the path to the remote server. Typically when using a browser to access the server the authentication request will be presented in the form of a dialog window that asks for the username and password.

Since the printer's connection to the remote server is headless and non-interactive, the Weblink configuration allows a user to enter in a server name/username/password triplet. The triplet will be used in the event that the printer is presented with an authentication request (for example, this typically is requested via the `HTTP/1.1 401 Unauthorized` request).

To specify authentication credentials, issue the following:

```
! U1 setvar "weblink.ip.conn1.authentication.add" "servername.com username password"
```

In this scenario the server requesting authentication is servername.com. The username and password to be supplied are 'username' and 'password'. The server name can be either a DNS name or an IP address. The username and password cannot be retrieved from SGD, SNMP, or JSON once added. Only the server name will be returned.

More than one set of authentication triplets can be added. The printer will only use the credentials as they are needed. In other words, the printer will only use the credentials for servername.com if it receives a `HTTP/1.1 401 Unauthorized` request from servername.com.

To see what authentication triplets are specified issue:

```
! U1 getvar "weblink.ip.conn1.authentication.entries"
```

To remove authentication credentials issue the following:

```
! U1 setvar "weblink.ip.conn1.authentication.remove" "servername.com"
```

Additional Firewall Configuration

Some firewalls do not allow the first connection attempt for a device to be HTTPS or require new connections periodically to keep the initial connections intact. The weblink test branch was provided to address issues that typically arise because the printer is an unattended device.

To configure the printer to attempt an HTTP connection anytime the HTTPS connection drops :

```
! U1 setvar "weblink.ip.conn1.test.location" "http://www.zebra.com/apps/linktest"
! U1 setvar "weblink.ip.conn1.test.test_on" "failure"
```

The weblink.ip.conn1.test.location can be any valid HTTP address. The default uses a link provided by Zebra that exists for no other purpose than to help developers test their connections to the internet. Setting `weblink.ip.conn1.test.test_on` to interval or both will force the printer to attempt a connection to the URL in location every `weblink.ip.conn1.test.retry_interval` seconds (default is 900 seconds/15 minutes).

To configure the printer to try an HTTP connection periodically, independent of the HTTPS success:

```
! U1 setvar "weblink.ip.conn1.test.location" "http://www.zebra.com/apps/
linktest"
! U1 setvar "weblink.ip.conn1.test.test_on" "interval"
! U1 setvar "weblink.ip.conn1.test.retry_interval" "900"
```

Difference Between Conn1 and Conn2

The printer has the ability to connect to two different servers.

Connection 1 (conn1) and Connection 2 (conn2) are identical in every way in terms of their configuration. It is expected that conn2 will typically be left unmodified unless a user has an alternate server that they wish to use to configure the printer.

A typical scenario in which both connections would be used is if a user wishes to have the printer connect to both a configuration server and a data source.

Enable Logging

If your printer has trouble connecting, you may wish to enable logging.

By default logging is not enabled in order to reduce the amount of memory consumed when the Weblink feature is enabled. It is recommended that once the Weblink feature is configured properly and is performing as expected that the logging be disabled or that a very small (less than 100) number of logging entries be permitted.

To enable logging, `weblink.logging.max_entries` needs to be modified. By default it is set to 0, which indicates that no messages are logged. When attempting to troubleshoot connection issues it is recommended that `max_entries` be set to at least 100 entries. Setting `max_entries` to 100 means that the 100 newest logging entries will be present in `weblink.logging.entries` as older entries are discarded once the maximum number of entries is reached.

```
! U1 setvar "weblink.logging.max_entries" "100"
```

The logging settings are atypical to the Weblink settings as they do not require the printer to be reset before taking effect. This does not mean that previous logging message that would have been logged will appear when the `max_entries` setting is changed from 0 to a greater value. It means that any new logging messages will be logged from that point forward.

Issue the following command to clear any log entries currently in the weblink.logging.entries buffer.

```
! U1 do "weblink.logging.clear" ""
```

Navigating the Log Output

The log will contain useful information, even in the scenario where the printer successfully connects to the remote server. This section explains how to read the log and highlights some of the key entries that will help to determine if the connection was successful.

A typical log entry looks as follows:

```
[12-04-2012 14:57:10.625] [conn1.1] Attempting connection to https://
www.examplecorpinc.com/zebra/weblink/
```

The first column is the date and time that the event occurred. The format of the date and time matches the format of `rtc.date` and `rtc.time`. The time, however, also includes the milliseconds to aid in troubleshooting network latency concerns.

NOTE: For printers that do not have a battery to store the Real Time Clock (RTC) value, the date will be restored to the default value upon a power cycle. The default value depends upon how the `rtc.date` SGD is managed. If it has never been set then it will default to the firmware build date (the value in `appl.date`). Otherwise, the value in `rtc.date` will default to the value that it was last set to. This does not mean the value of the `rtc.date` when it was power cycled. It means that when a user sets `rtc.date` that becomes the new default value.

If the printer has a battery then the `rtc.date` is never default and continues to track the date as expected.

The second column indicates the connection name and channel that the entries are associated with. The connection name will match the weblink branch that was configured with the respective URL (for example, conn1 or conn2). The channel number indicates which channel on the respective connection the entries corresponds to.

NOTE: Channels are additional connections that are requested by the server when the server needs to perform a specific operation that cannot be done on the channel(s) currently open. Typically only the RAW channel is open which operates similar to the RAW TCP port. It is typical to see two channels opened, the main channel and the RAW channel.

The third column is the actual message that contains information about what occurred in the printer at the corresponding time in column one. In the above example the printer was initiating the connection to the URL specified in `weblink.ip.conn1.location`.

Review the section titled SSL/TLS Certificate Errors to understand what it means when certain logging messages/errors appear in the log.

SSL/TLS Certificate Errors

Secure connections to the remote server present the opportunity for several errors when attempting to connect.

The errors typically involve the certificates used when connecting via SSL or TLS. This section highlights some of the most common issues involving the certificates.

Error	Cause / Solution
"SSL certificate problem: self signed certificate in certificate chain"	One of the situations that prevent a successful connection is not having the correct Certificate Authority certificates installed on the remote server. Zebra requires that the Zebra Root Certificate Authority and the Zebra Subordinate Certificate Authority be installed on the remote server. This error typically indicates that only one of the Zebra Certificate Authority certificates is installed on the remote server. **IMPORTANT:** When using certificate files, the time on the printer must be set correctly for the websocket connection to succeed, as the time is used in the certificate validation.
"SSL certificate problem: unable to get local issuer certificate"	One of the situations that prevent a successful connection is not having the correct Certificate Authority certificates installed on the remote server. Zebra requires that the Zebra Root Certificate Authority and the Zebra Subordinate Certificate Authority be installed on the remote server. This error typically indicates that neither of the Zebra Certificate Authority certificates are installed on the remote server. **IMPORTANT:** When using certificate files, the time on the printer must be set correctly for the websocket connection to succeed, as the time is used in the certificate validation.
"SSL certificate problem: certificate has expired"	This error indicates that the remote server's certificate has expired. This is typically an indication that the printer's date and/or time are incorrect as the Zebra certificates are typically issued for long durations. Check that `rtc.date` and `rtc.time` are set correctly. **NOTE:** For printers that do not have a battery to store the Real Time Clock (RTC) value, the date will be restored to the default value upon a power cycle. The default value depends upon how the `rtc.date` SGD is managed. If it has never been set then it will default to the firmware build date (the value in `appl.date`). Otherwise, the value in `rtc.date` will default to the value that it was last set to. This does not mean the value of the `rtc.date` when it was power cycled. It means that when a user sets `rtc.date` that becomes the new default value. If the printer has a battery then the `rtc.date` is never default and continues to track the date as expected. **IMPORTANT:** When using certificate files, the time on the printer must be set correctly for the websocket connection to succeed, as the time is used in the certificate validation.

Error	Cause / Solution
`"SSL certificate problem: certificate is not yet valid"`	This error indicates that the remote server's certificate was incorrectly issued or that the printer's date and/or time are incorrect. Check that the printer's date and time (`rtc.date` and `rtc.time`) are set correctly and that the certificate's start and expiration date are valid. **NOTE:** For printers that do not have a battery to store the Real Time Clock (RTC) value, the date will be restored to the default value upon a power cycle. The default value depends upon how the `rtc.date` SGD is managed. If it has never been set then it will default to the firmware build date (the value in `appl.date`). Otherwise, the value in `rtc.date` will default to the value that it was last set to. This does not mean the value of the `rtc.date` when it was power cycled. It means that when a user sets `rtc.date` that becomes the new default value. If the printer has a battery then the `rtc.date` is never default and continues to track the date as expected.

`"subjectAltName does not match 1.2.3.4"`
 	
`"SSL certificate subject name 'examplecorpinc.com' does not match target host name '1.2.3.4'"`
	Part of the certificate validation process involves making sure that the remote server is who it claims to be. A certificate can be created to validate against several aliases/DNS names. Typically the certificate will not contain the IP address of the server as IP addresses are subject to change. When specifying the remote server's URL via `weblink.ip.conn1.location` be sure to specify one of the DNS aliases listed in the certificate. The valid names will be listed either under the Common Name (CN) field and/or the subjectAltName (SAN or Subject Alternate Name) field within the certificate. For example, the certificate may have the CN set to `'examplecorpinc'` and the SAN set to `'examplecorpinc.com'` or `'alias.for.examplecorpinc.com'`. Any of the CN or SAN names can be used, but, as the IP address is not listed in the CN or SAN it cannot. It is not recommended that the IP address be part of the SAN if a DNS name is available to avoid connection issues that may arise due to subnet change or DHCP lease expirations, etc.
	
	IMPORTANT: When using certificate files, the time on the printer must be set correctly for the websocket connection to succeed, as the time is used in the certificate validation.

`"Unknown SSL protocol error in connection to ..."`	When this message is seen it means that the remote server's SSL/TLS configuration is incorrect. Refer to Troubleshooting to ensure the server and printer are both configured correctly.
I do not see any of these errors, but the printer still does not connect.	Refer to Troubleshooting to ensure the server and printer are both configured correctly.

Other Typical Errors

While SSL/TLS connection errors are the most common, there are issues that can arise that prevent a successful connection.

This section highlights the most common issues.

Error	Cause / Solution
"Read failed with an unexpected error"	This message typically indicates that connection to the remote server was lost. The connection can either be lost due to the server powering off or resetting, the firewall or proxy server shutting down the connection, or because the remote server gracefully requests that the connection be discontinued. **NOTE:** After 60 seconds of inactivity on the connection the printer will attempt to contact the server via a TCP Keepalive. If the connection is still present the server will respond and the connection will remain open. After 10 successive failed attempts to contact the remote the printer will assume the connection is severed and close the connection. The printer will resume it's attempt to connect to the remote server so that when the server comes back online the printer will re-establish communication.
"Failed to connect (SP = #, CI = #, UW = #, AC = #, PC = #)"	If this error is seen one or more of the '#' values will be set to 0. This is an indication of an incorrect configuration of the remote server. Ensure that the remote server is setup according to the Servlet configuration in the Zebra Link-OS SDK documentation.This typically indicates an incorrect version of the remote Application Server (for example, Apache/Tomcat version may be incorrect). If this issue persists contact Zebra Technical Support.. Provide the output of the following command (ensure that logging is enabled and that this error appears within the entries).! U1 getvar "weblink"

Troubleshooting

Whenever troubleshooting a connection issue, the following questions should be answered to ensure the configuration is correct.

1. Is the printer connected correctly via Wireless or Ethernet?
2. Does the printer have a valid IP address?
3. Can I ping the printer's IP address from a device on the same network as the printer?
4. Is the remote server URL in `weblink.ip.conn1.location` correct and does it point to the remote server that is configured for weblink functionality?
5. Can you connect to the location defined in the `weblink.ip.conn1.location` setting via a browser?
6. Is the remote server I am attempting to connect to outside the corporate firewall?
7. Can the URL specified in `weblink.ip.conn1.test.location` be accessed?

   If this is the case, talk with your administrator about altering restrictions for accessing HTTPS connections.
8. Does the firewall require a username and password to access the remote server?
9. Do I require a proxy server to access the remote server?

10. Is the proxy server port the default (1080) or another port (for example, 3128 for the Linux Squid proxy)?

NOTE: If using the Linux Proxy Server Squid, and you are having trouble connecting, note that it may be configured to:

- disallow POST messages
- only operate in HTTP/1.0 mode 3
- disallow SSL connections.

Refer to your Linux Squid documentation for complete details.

11. Does the firewall permit HTTPS connections initially or do I need to connect via HTTP first?
12. Is the remote server configured to use a supported version of TLS?

    Zebra currently supports TLS 1.2 and recommends using the most recent version whenever possible. Earlier versions may not be supported in the future.
13. Are the Zebra Certificate Authority Certificates correctly installed on the remote server?
14. Was the server's certificate issued by Zebra and is it signed by the Zebra Certificate Authority?
15. Has the server's certificate expired?
16. Is the printer's date and time within the issue and expired period of the server's certificate?
17. Does the value in `weblink.ip.conn1.location` match either the Common Name or one of the names listed in the Subject Alternate Name of the remote server's certificate?
18. Is the proxy server configured correctly and does the respective proxy server allow HTTPS connections via the HTTP CONNECT method?
19. Are there any HTTP authentication attempts when trying to connect that fail?
20. Are there any HTTP/1.1 4xx messages in the log?

    If your connection issues persist and the solutions in this document do not help, contact Zebra Tech Support and provide the output of the following command. Ensure that logging is enabled and that the error(s) appear within the entries)

    ! U1 getvar "weblink"

HTTP Messages

List of HTTP messages.

Message	Cause / Solution
`HTTP/1.1 100 Continue`	This indicates that the server and printer have begun communicating and is often seen in place of `HTTP/1.1 200 OK`.
`HTTP/1.1 101 Switching Protocols`	This indicates that the basic connection to the server worked and the protocol is being switched to a more efficient protocol for data transfer.
`HTTP/1.1 200 OK`	This indicates that an `HTTP GET` or `HTTP POST` was successful.
`HTTP/1.1 30x Moved/ Redirect/etc`	This indicates that the URL specified has moved or that the firewall redirected the printer to another location (typically this is done to authenticate a user in a transparent proxy configuration).

Message	Cause / Solution
`HTTP/1.1 401 Unauthorized`	This indicates that the printer either needs to authenticate with the server or failed to authenticate with the remote server (or server/ router along the route to the server).
`HTTP/1.1 403 Forbidden`	This typically means that the authentication was provided and valid, however, the user does not have access to the requested resource.
`HTTP/1.1 404 Not Found`	This indicates that the remote URL provided points to an invalid location on the server. This does indicate, however, that the server name is valid. Just the path after the domain name is invalid.

HTTP POST Alerts

Link-OS printers can issue alerts to a web server that is listening for HTTP POST requests. The advantage of an HTTP POST alert over the other destinations available (for example, TCP, UDP, SNMP) is that HTTP is firewall friendly.

Configuring Alerts Where the Alert Destination is HTTP POST

Any setting in the `alerts.http` branch that is set will take effect for any HTTP POST alerts that occur from that point forward. A printer reset is not required for the settings to take effect.

If the server is configured to accept and process HTTP POST messages either via a CGI script or a server-side script such as PHP or ASP then the alert can be forwarded to that server from the printer. The printer will send the alert using the multipart/form-data Content-Type. This allows any type of data, including binary data, to be sent via the POST.

The POST will support two variables within the body of the POST:

- alertMsg – This is the alert details and the content follows the format of a standard alert when it is issued over one of the other alert destinations (for example, serial, USB, TCP, etc.).
- uniqueId – The unique id of the printer. This matches the value in `device.unique_id`.

  The HTTP POST request will look as follows (sent when the printer was paused)

```
POST /http_post/alert.php HTTP/1.1
Host: 10.3.4.58
Accept: */*
Connection: close
Content-Length: 281
Expect: 100-continue
Content-Type: multipart/form-data;
 boundary=----------------------------350c75835f46

------------------------------350c75835f46
Content-Disposition: form-data; name="alertMsg"

ALERT%3A%20PRINTER%20PAUSED
------------------------------350c75835f46
Content-Disposition: form-data; name="uniqueId"

XXQLJ120900310
------------------------------350c75835f46--
```

IMPORTANT: The message is using HTTP/1.1 and therefore HTTP/1.1 header fields. This is important because some older proxy servers do not handle these fields gracefully and may block the POST message.

It is important to note that the message is using HTTP/1.1 and therefore HTTP/1.1 header fields. This is important because some older proxy servers do not handle these fields gracefully and may block the POST message.

How to Parse via PHP

The following example shows how to parse the POST message.

It does not, however, show how to use this information on other pages, store the results in a database, report this to another device on the domain, etc. The response in this example will be sent back to the printer, but it will be ignored by the printer. If you wish to see the response you can use a packet sniffing tool such as Wireshark.

```
<?php

    $alertMsg = urldecode($_POST["alertMsg"]);
    if (preg_match("/(\w+(\s+\w+)?):\s+(((SGD SET)\s+([\w\d\.\_]+)\s+\->\s+(.
+))|([\w|\s]+))/", $alertMsg, $matches)== 1) {

        $alertType = $matches[1];

        if ($matches[5] === "SGD SET") {
            $alertCondition = $matches[5];
            $alertSgdName = htmlspecialchars($matches[6]);
            $alertSgdData = htmlspecialchars($matches[7]);
        } else {
            $alertCondition = $matches[3];
        }
    }

    echo "<H1>Alert Received</H1><br/>\r\n";
    echo "<h2>Original Message = $alertMsg</h2><br/>\r\n";
    echo "<h2>Alert Type = $alertType</h2><br/>\r\n";
    echo "<h2>Alert Condition = $alertCondition</h2><br/>\r\n";
    echo "<h2>SGD Name = $alertSgdName</h2><br/>\r\n";
    echo"<h2>SGD Value = $alertSgdData</h2><br/>\r\n";
?>
```

Basic Configuration

To determine how much configuration is necessary, consider the following questions:

- Is the remote server that the printer is attempting to connect to outside of the corporate firewall?
- Does the firewall require a username and password to access the remote server?
- Does the printer require a proxy server to access the remote server?

If the answer to any of these questions is ‘yes’, then more than the basic configuration may be necessary. Depending upon the network environment that the printer is in access the remote server may only require that a few settings be set.

To configure an alert to be sent via HTTP POST to a remote server, issue the following command:

```
! U1 setvar "alerts.add" "PRINTER PAUSED,HTTP POST,Y,Y,
    http://www.examplecorpinc.com/alerts.php,0,N,"
```

The above command will issue an HTTP POST alert to the remote server (http://www.examplecorpinc.com/alerts.php) when the printer is paused or un-paused.

- The first parameter indicates the condition to monitor. A list of available alert conditions can be viewed by issuing:

```
! U1 getvar "alerts.conditions"
```

- The second parameter indicates the alert destination. For the purposes of this section HTTP-POST is the preferred destination. A list of available alert destinations can be viewed by issuing:

```
! U1 getvar "alerts.destinations"
```

- The third and fourth parameters are ‘Send on Set’ and ‘Send on Clear’, respectively. They can be either "Y" for monitor the alert or "N" for don’t monitor the alert. If both are set to "N" then the alert will not be added or it will be deleted if the alert already existed. To view which alerts already exist issue:

```
! U1 getvar "alerts.configured"
```

- The fifth parameter holds the URL for the server that will be sent the HTTP POST. It holds a maximum of 255 characters for the URL and it must conform to the URI standards described in RFC3986 (http://www.ietf.org/rfc/rfc3986.txt).
- The sixth parameter should be set to 0 for HTTP POST alerts.
- The seventh parameter and eighth parameter will not be covered in this section and should be set as indicated in the description above. See the SGD documentation for details on these two parameters.

When a Proxy Server is Part of the Network Configuration

If a proxy server must be used to access the remote server, the printer’s proxy setting must be set to connect to the server.

There are typically four properties associated with a proxy server:

- The proxy server scheme: HTTP or HTTPS
- The proxy server address
- The proxy server port (optional)
- The username and password for the proxy (optional)

To supply the address of the proxy server (assuming a default port and no username/password), configure the proxy setting as follows:

```
! U1 setvar "weblink.ip.conn1.proxy" "https://my.internal.proxy/"
```

In this scenario the proxy address is `my.internal.proxy` and the scheme is HTTPS. The default port (1080) will be used. No username or password will be used to authenticate with the proxy.

To specify an alternate port configure the proxy as follows:

```
! U1 setvar "weblink.ip.conn1.proxy" "https://my.internal.proxy:3128/"
```

To specify a username and password configure the proxy as follows:

```
! U1 setvar "weblink.ip.conn1.proxy" "https://user:pass@my.internal.proxy/"
```

The proxy username, password, and the rest of the URL must follow the rules specified in RFC3986 (http://www.ietf.org/rfc/rfc3986.txt).

When HTTP Authentication is Necessary

Use this configuration when, for example, a firewall requires a username and/or password.

It may be necessary to specify a username and password to various routers and servers along the path to the remote server. Typically when using a browser to access the server the authentication request will be presented in the form of a dialog window that asks for the username and password.

Since the printer's connection to the remote server is headless and non-interactive, the Weblink configuration allows a user to enter in a server name/username/password triplet. The triplet will be used in the event that the printer is presented with an authentication request (for example, this typically is requested via the `HTTP/1.1 401 Unauthorized` request).

To specify authentication credentials, issue the following:

```
! U1 setvar "weblink.ip.conn1.authentication.add" "servername.com username password"
```

In this scenario the server requesting authentication is servername.com. The username and password to be supplied are 'username' and 'password'. The server name can be either a DNS name or an IP address. The username and password cannot be retrieved from SGD, SNMP, or JSON once added. Only the server name will be returned.

More than one set of authentication triplets can be added. The printer will only use the credentials as they are needed. In other words, the printer will only use the credentials for servername.com if it receives a `HTTP/1.1 401 Unauthorized` request from servername.com.

To see what authentication triplets are specified issue:

```
! U1 getvar "weblink.ip.conn1.authentication.entries"
```

To remove authentication credentials issue the following:

```
! U1 setvar "weblink.ip.conn1.authentication.remove" "servername.com"
```

Enabling Logging

If your printer has trouble connecting, you may wish to enable logging.

By default, logging is not enabled in order to reduce the amount of memory consumed when the HTTP alert feature is enabled. It is recommended that once the alert HTTP feature is configured properly and is performing as expected that the logging be disabled or that a very small (less than 100) number of logging entries be permitted.

To enable logging, `alerts.http.logging.max_entries` needs to be modified. By default it is set to 0, which indicates that no messages are logged. When attempting to troubleshoot connection issues it is recommended that max_entries be set to at least 100 entries. Setting max_entries to 100 means that the 100 newest logging entries will be present in alerts.http.logging.entries as older entries are discarded once the maximum number of entries is reached.

```
! U1 setvar "alerts.http.logging.max_entries" "100"
```

The logging settings do not require the printer to be reset before taking effect. This does not mean that previous logging message that would have been logged will appear when the max_entries setting is changed from 0 to a greater value. It means that any new logging messages will be logged from that point forward.

Issue the following command to clear any log entries currently in the alerts.http.logging.entries buffer.

```
! U1 do "alerts.http.logging.clear" ""
```

Navigating the Log Output

The log can contain much information, even in the scenario where the printer successfully connects to the remote server. This section explains how to read the log and highlights some of the key entries that will help to determine if the connection was successful.

A typical log entry looks as follows:

```
[12-04-2012 14:57:10.625] [http] Attempting connection to http://
www.examplecorpinc.com/alerts.php
```

The first column is the date and time that the event occurred. The format of the date and time matches the format of `rtc.date` and `rtc.time`. The time, however, also includes the milliseconds to aid in troubleshooting network latency concerns.

The second column indicates the connection name, which is currently always set to `'http'`.

The third column is the actual message that contains information about what occurred in the printer at the corresponding time in column one. In the above example the printer was attempting to POST the alert to the connection to the URL specified in the configured alert.

Review Understanding Errors in the Alerts HTTP Log to understand what it means when certain logging messages/errors appear in the log.

Understanding Errors in the Alerts HTTP Log

Errors in the Alerts HTTP log.

Error	Cause / Solution
`Couldn't connect to host`	This could mean any number of things occurred that prevented the printer from connecting. This message is always present when the connection to the remote failed and is typically accompanied by an HTTP Response Code. See HTTP Messages for the possible HTTP Response Codes and their meaning.If this issue persists contact Zebra Technical Support and provide the output of the following command (ensure that logging is enabled and that this error appears within the entries). `! U1 getvar "alerts"`

Troubleshooting

Whenever troubleshooting a connection issue, the following questions should be answered to ensure the configuration is correct.

1. Is the printer connected correctly via Wireless or Ethernet?
2. Does the printer have a valid IP address?
3. Can I ping the printer's IP address from a device on the same network as the printer?
4. Is the remote server URL in `weblink.ip.conn1.location` correct and does it point to the remote server that is configured for weblink functionality?
5. Can you connect to the location defined in the `weblink.ip.conn1.location` setting via a browser?
6. Is the remote server I am attempting to connect to outside the corporate firewall?
7. Can the URL specified in `weblink.ip.conn1.test.location` be accessed?

   If this is the case, talk with your administrator about altering restrictions for accessing HTTPS connections.
8. Does the firewall require a username and password to access the remote server?
9. Do I require a proxy server to access the remote server?
10. Is the proxy server port the default (1080) or another port (for example, 3128 for the Linux Squid proxy)?

NOTE: If using the Linux Proxy Server Squid, and you are having trouble connecting, note that it may be configured to:

- disallow POST messages
- only operate in HTTP/1.0 mode 3
- disallow SSL connections.

Refer to your Linux Squid documentation for complete details.

11. Does the firewall permit HTTPS connections initially or do I need to connect via HTTP first?

12. Is the remote server configured to use a supported version of TLS?

    Zebra currently supports TLS 1.2 and recommends using the most recent version whenever possible. Earlier versions may not be supported in the future.

13. Are the Zebra Certificate Authority Certificates correctly installed on the remote server?
14. Was the server's certificate issued by Zebra and is it signed by the Zebra Certificate Authority?
15. Has the server's certificate expired?
16. Is the printer's date and time within the issue and expired period of the server's certificate?
17. Does the value in `weblink.ip.conn1.location` match either the Common Name or one of the names listed in the Subject Alternate Name of the remote server's certificate?
18. Is the proxy server configured correctly and does the respective proxy server allow HTTPS connections via the HTTP CONNECT method?
19. Are there any HTTP authentication attempts when trying to connect that fail?
20. Are there any HTTP/1.1 4xx messages in the log?

    If your connection issues persist and the solutions in this document do not help, contact Zebra Tech Support and provide the output of the following command. Ensure that logging is enabled and that the error(s) appear within the entries)

    ! U1 getvar "weblink"

HTTP Messages

List of HTTP messages.

Message	Cause / Solution
`HTTP/1.1 100 Continue`	This indicates that the server and printer have begun communicating and is often seen in place of `HTTP/1.1 200 OK`.
`HTTP/1.1 200 OK`	This indicates that the `HTTP POST` was successful.
`HTTP/1.1 30x Moved/ Redirect/etc`	This indicates that the URL specified has moved or that the firewall redirected the printer to another location (typically this is done to authenticate a user in a transparent proxy configuration).
`HTTP/1.1 401 Unauthorized`	This indicates that the printer either needs to authenticate with the server or failed to authenticate with the remote server (or server/ router along the route to the server).
`HTTP/1.1 403 Forbidden`	This typically means that the authentication was provided and valid, however, the user does not have access to the requested resource.
`HTTP/1.1 404 Not Found`	This indicates that the remote URL provided points to an invalid location on the server. This does indicate, however, that the server name is valid. Just the path after the domain name is invalid.

Advanced Techniques

This section presents information and commands for using advanced techniques, such as special effects, serialized data fields, control commands, program delimiters, communications, and memory cards.

Special Effects for Print Fields

This section lists some special effect for print fields.

- Reverse Printing a Field - The ^FR (Field Reverse Print) command allows a field to appear as white over black or black over white. When printing a field, the ^FR command indicates that it will print the field the opposite of its background color.
- Reverse Printing a Label - The ^LR (Label Reverse Print) command reverses the printing of all fields in the label format. It allows a field to appear as white over black or black over white. ^LR functions like ^FR, but it applies to all fields in a label. The ^LR command remains active until turned off.
- Printing a Mirror Image - The ^PM (Print Mirror Image of Label) command prints the entire printable area of the label as a mirror image. This command flips the image from left to right.
- Printing a Label Inverted 180 Degrees - The ^P0 (Print Orientation) command inverts the label format 180 degrees. In essence, the label is printed upside down.

Serialized Data

The ^SN (Serialization Data) command allows the printer to index data fields by a selected increment or decrement value (that is, make the data fields increase or decrease by a specified value) each time a label is printed.

This can be performed on up to 100 to 150 fields in a given format and can be performed on both alphanumeric and bar code fields. A maximum of 12 of the right-most integers are subject to indexing. The first integer found when scanning from right to left starts the indexing portion of the data field.

If the alphanumeric field to be indexed ends with an alpha character, the data will be scanned, character-by-character, from right to left until a numeric character is encountered. Serialization will take place using the value of the first number found.

Variable Data

To increase throughput, you can set up a program that uses variable data fields. Then, instead of formatting the whole label each time a label is printed, the printer will have to format only the changed data field. To use this capability, you must use the ^MC and ^FV commands.

Stored Formats

You can create formats and save them in the printers memory. A stored format can then be recalled and merged with downloaded data to form a complete label. This process saves data transmission time but not formatting time.

To create a format, complete these steps:

1. Design the label.
2. Replace variable data fields with field numbers.
3. Allocate space for the size of the field.
4. Give the format a name.
5. Save the format on the printer to a memory location (R, E, B, A).

You can store multiple formats on the printer, limited only by available memory. If you try to save a format that would overload memory you can confirm that the format has been successfully stored on the printer by printing the LIST FORMATS from the front panel, or by using the ^HW command to return the directory listing to the host. For details see, ^HW.

If the power is turned off, all stored formats in volatile memory (R:) will be lost.

Initialize/Erase Stored Formats

Stored formats can be selectively erased using the ^ID command.

Download Format Command

The ^DF (Download Format) command saves the ZPL II format commands as text strings to be later merged using ^XF with variable data. The format to be stored may contain Field Number (^FN) commands to be referenced when recalled.

While use of stored formats will reduce transmission time, no formatting time is saved since this command saves the ZPL II as text strings which need to be formatted at print time.

Field Number Command

The ^FN (Field Number) command is used to number the data fields. This command is used in both Store Format and Recall Format operations.

When storing a format, the ^FN command is used where you would normally use the ^FD (Field Data) command. When recalling the stored format, use ^FN in conjunction with the ^FD(Field Data) command.

Recall Stored Format Command

The ^XF (Recall Format) command recalls a stored format to be merged with variable data. There can be multiple ^XF commands and they can be located anywhere in the label format.

When recalling a stored format and merging data utilizing the ^FN (Field Number) function, the calling format must contain the ^FN command to merge the data properly.

While the use of stored formats will reduce transmission time, no formatting time is saved because the format being recalled was saved as text strings that need to be formatted at print time.

These are examples of using the stored format:

Working with Stored Format commands involves designing and saving a stored format, then recalling and merging the format with some variable data.

The following is an example of how to use the various Stored Format commands. First, enter the following format and send it to the printer. Notice that no label is printed. (DATA Indicator went On and Off.)

```
^XA^DFFORMAT^FS
^LH30,30
^BY2,3,100
^FO120,100^CFD^FN1^FS
^FO120,160^B3^FN2^FS
^XZ
```

Second, enter the following format and send it to the printer. The label shown will be printed.

```
^XA^XFFORMAT^FS
^FN1^FDLABEL ONE^FS
^FN2^FDAAA001^FS
^XZ
```

Control Commands

Control commands may be sent from the host at any time to elicit an immediate response from the printer. Control commands may be sent in a group or separately.

A control command is acted upon when received to perform a variety of actions, such as:

- clearing the memory
- physical action
- a combination of the above such as feeding a label and calculating and storing its length.

The basic format for using all of the control commands is:

~(2-letter command)

For example: ~DG

Test and Setup Commands

The following commands, presented in alphabetical order, are used to test various elements of the printer and its status.

Table 45 Test and Setup Commands

Command	Function
~HM (Memory Status)	Sending this command to the printer immediately returns a memory status message to host. Use this command whenever you need to know the status of the memory.
~HS (Host Status)	Sending this command to the printer immediately returns a three-line printer status message to the host. Use this command whenever you need to know the status of the printer.
~JR (Power On Reset)	This command resets all of the printer's internal software, performs a power-on self-test, clears the buffer and DRAM, and resets communication parameters and default values. ~JR performs the same function as a manual power-on reset.
~JN (Head Test Fatal)	This command resets the printhead element error override, acting as a toggle for ~J0. The printer then goes into fault status (turns head indicator on steadily) if any subsequent execution of the printing element test detects bad printing elements. This command is only functional on certain printer platforms.
~J0 (Head Test Non-Fatal)	This command overrides a failure of head element status check and allows printing to continue. The override is canceled when the printer is turned off or receives a ~JR or ~JN command. The printhead test will not produce an error if the ~J0 override is active. This command is only functional on certain printer platforms.
^JT (Head Test Interval)	This command lets you change the printhead test interval from 100 to any desired interval. The printer automatically performs an internal printhead element test, which occurs every 100 labels. This takes place during formatting which minimizes a delay in printing. Therefore, the test may be performed while the printer is in PAUSE. This command is only functional on certain printer platforms.
~WC (Print Configuration Label)	The ~WC command is used to generate a printer configuration label. The printer configuration label contains information about the printer setup, such as sensor type, network ID, ZPL mode, firmware version, and descriptive data on the R:, E:, B:, and A: devices.
~HQ (Host Query)	The ~HQ command group causes the printer to send information back to the host.

Calibration and Media Feed Commands

The following commands, presented in alphabetical order, are used to perform various media and ribbon calibrations and also set the media feed mode for the printer.

Command	Function
~JC (Set Media Sensor Calibration)	Forces a label length measurement and recalibrates the media and ribbon sensors. **NOTE:** In continuous mode, only the media and ribbon sensors are recalibrated.
~JG (Graphing Sensor Calibration)	Forces a label length measurement, recalibrates the media and ribbon sensors, and prints a graph (media sensor profile) of the sensor values.
~JL (Set Label Length)	Sets the label length. Depending on the size of the label, the printer will feed one or more blank labels.
^MF (Media Feed)	Dictates what happens to the media at power up and after an error is cleared.

Cancel/Clear Commands

The following command controls the contents of the Zebra input buffer.

Command	Function
~JA (Cancel All)	Cancels all format commands in the buffer. It also cancels any batches that may be printing. The printer stops printing after the current label (if one is printing) is finished printing. All internal buffers are cleared of data. The DATA LED turns off.

Printer Control Commands

The following commands control various printer operations:

Command	Function
^PF (Slew Given Number of Dot Rows)	Causes the printer to slew labels (move labels at a high speed without printing) a specified number of dot rows, at the bottom of the label. This allows faster printing when the bottom portion of a label is blank.
~PH or ^PH (Slew to Home Position)	Causes the printer to feed one blank label. • The ~PH command feeds one label after the format currently being printing is done or when the printer is placed in pause. • The ^PH command feeds one blank label after the format it is in prints.
~PP (Programmable Pause)	Stops printing after the current label is printed (if one is printing) and places the printer in the Pause mode.The operation is identical to pressing the PAUSE button on the front panel of the printer. The printer will remain paused until the PAUSE button is pressed or a ~PS command is sent to the printer.

Command	Function
^PP (Programmable Pause)	This command pauses the printer after the format it is in prints. Because this command is not executed immediately, several labels may be printed before the printer is paused.The operation is identical to pressing the PAUSE button on the front panel of the printer. The printer will remain paused until the PAUSE button is pressed or a ~PS command is sent to the printer.
^PQ (Print Quantity)	This command gives control over several printing operations. It controls the number of labels to print, the number of labels printed before the printer pauses, and the number of replications of each serial number.
^PR (Print Rate)	Determines the media speed during printing and the slew speed (feeding a blank label). The printer will operate with the selected speeds until the setting is resent in a subsequent format or the printer is turned off. **Limitations of Higher Print Speeds** Print speed is application specific. Because print quality is affected by media and ribbon, printing speeds, and printer operating modes, it is very important to run tests for your applications. • With high print speeds, use thermal transfer mode only. • Horizontal bar codes with a minimum x dimension of 5 mil may be printed at print speeds of 2 in. (51mm) per second. • Rotated bar codes are limited to a minimum x dimension of 10 mil (modulus 2) at higher print speeds. At x dimension of 5 mil (modulus 1), they may be printed at 2 in. per second. • At high print speeds, Font A at a magnification of 1 is not recommended; all other fonts are acceptable.
~PS (Print Start)	Causes a printer in the Pause mode to resume printing. The operation is identical to pressing the PAUSE button on the front panel of the printer when the printer is already in the Pause mode.

Set Dots/Millimeter

The following commands change the number of dots per millimeter.

Command	Function
^JM (Set Dots/Millimeter)	Changes the number of dots printed per millimeter. Depending on the printhead, normal dots per millimeter on a Zebra printer are the following: • 24 dots/mm (609.6 dots/inch) • 12 dots/mm (304.8 dots/inch) • 8 dots/mm (203.2 dots/inch) • 6 dots/mm (152.4 dots/inch) In some applications, these high densities are not required. For these applications, a lower density of 4 dots/mm (102 dots/inch) or 3 dots/mm (77 dots/inch) can be selected.If used, this command must be entered before the first ^FS command.

Host Status Commands

The following commands control the host device.

Table 46 Host Status Commands

Command	Function
~HI (Host Identification)	This command is designed to be sent from the Host to the Zebra printer to find out the type of Zebra printer. Upon receipt, the Zebra printer will respond to the Host with a character string that gives information about the printer such as the version of firmware, dots per inch, memory, and printer options.
^SP (Start Print)	This command allows a label to start printing at a specified point before the entire label has been completely formatted. On extremely complex labels, this command can increase the overall throughput of the printer. The command works as follows: you specify the dot row at which the ^SP command is to take affect. This then creates a label 'segment.' Once the ^SP command is processed, all information in that segment will be printed. During the printing process, all of the commands after the ^SP will continue to be received and processed by the printer. If the segment after the ^SP command (or the remainder of the label) is ready for printing, media motion does not stop. If the next segment is not ready, the printer will stop "mid-label" and wait for the next segment to be completed. Precise positioning of the ^SP command is somewhat of a trial-and-error process as it depends primarily on print speed and label complexity. The ^SP command can be effectively used to determine the worst-case print quality. You can determine if using the ^SP command is appropriate for the particular application by using the following procedure. If you send the label format up to the first ^SP command and then wait for printing to stop before sending the next segment, the printed label will be a sample of the worst case print quality. It will also drop any field that is out of order.
~WC (Print Configuration Label)	This command is used to generate a printer configuration label. This command only works when the printer is idle.
~WL Print Network Configuration Label	This command is used to generate a network configuration label. This command only works when the printer is idle.

Changing Delimiters and Command Prefixes

For some applications, you may need to change the ZPL II default delimiter (,) the format command default prefix (^), and/or the control command default prefix (~). Any ASCII character may be set as the delimiter.

IMPORTANT: The delimiters used in the incoming ZPL script must match the delimiters set on the printer. If you change the delimiters on the printer, any ZPL script that uses the default delimiters will not work.

You might change these characters if you are using a hand-held terminal that does not have a comma to enter the ZPL II commands, if you are working with a mainframe that has trouble processing the caret, or if you find some other character(s) easier to use.

Reasons to set an alternate delimiter include, but are not limited to:

- you are using a hand-held terminal that does not have a comma to enter the ZPL II commands;
- you are working with a host system that does not easily output the default delimiter (for example, AS/400)
- you find some other character(s) easier to use.

Communication Diagnostics Commands

Zebra printers support communication diagnostics through both hardware and software control. You can use these diagnostics to troubleshoot programs.

Command	Function
~JD (Enable Communications Diagnostics)	Initiates a diagnostic mode that produces an ASCII printout (using current label length and full width of printer) of all characters received by the printer. This printout includes the ASCII Characters, the HEX value and any communication errors.
~JE (Disable Diagnostics)	Cancels the diagnostic mode and returns the printer to normal label printing.

Graphic Commands

In addition to text and bar codes, multiple types of graphics can be printed on a Zebra printer:

- boxes and lines (^GB), circles (^GC), diagonal lines (^GD), and ellipses (^GE)
- ZPL II label formats saved as graphics images
- graphic images in Hexadecimal format
- graphic symbols (^GS)

Table 47 Boxes, Lines, Circles, Diagonal Lines, and Ellipses

Command	Function
^GB (Graphic Box)	The ^GB command is used to draw boxes and lines as part of a label format. Boxes and lines are used to highlight important information, divide labels into distinct areas, or to improve the appearance of the label. The same format command is used for drawing either boxes or lines.
^GC (Graphic Circle)	The ^GC command produces a circle on the printed label. The command parameters specify the diameter (width) of the circle, outline thickness, and color. Thickness extends inward from the outline.
^GD (Graphic Diagonal Line)	The ^GC command produces a circle on the printed label. The command parameters specify the diameter (width) of the circle, outline thickness, and color. Thickness extends inward from the outline.
^GE (Graphic Ellipse)	The ^GE command produces an ellipse in the label format.
^GS (Graphic Symbol)	The ^GS command enables you to generate the registered trademark, copyright symbol, and other symbols.

These label formats can also be stored as graphic images and data can be merged with them at print time. Additionally, ZPL II will permit the printing of graphic images from other sources that have been created in (or converted to) hexadecimal (HEX) format. Such graphic images can come from a variety of sources, including CAD programs, draw and paint programs, and scanned images.

Image Move

The ^IM (Image Move) command performs a direct move of an image from a storage area into the bitmap. The command is identical to the Recall Graphic command except that there are no sizing parameters.

Working with Label Formats as Graphics

The ^IS (Image Save) and ^IL (Image Load) commands are used to save a ZPL label format (including text and/or bar codes) in the printer's DRAM, FLASH, or PCMCIA as a special graphic image.

The ^IS (Image Save) and ^IL (Image Load) commands are used to save a ZPL label format (including text and/or bar codes) in the printer's DRAM, FLASH, PCMCIA, or battery backed up SRAM, as a special graphic image. This increases the throughput of a series of similar but not identical labels.

Instead of formatting each individual label completely, store the constant fields as an image (known as creating a template). Then, in subsequent label formats, commands are issued to recall that graphic image format and merge it with variable data.

Working with Hex Graphic Images

ZPL II can be used to save graphic images in HEX format in DRAM, FLASH, or PCMCIA, depending on the type of memory installed in your printer.

ZPL II can be used to save graphic images in HEX format in DRAM, FLASH, PCMCIA, or battery backed up SRAM, depending on the type of memory installed in your printer. The image might be created using a CAD program, a draw or paint program, or a scanner. These images can then be printed on the label. Graphic images may be created using a program that creates files in the .PCX format. These files must then be converted to ZPL II graphic format .GRF (pure hexadecimal data without headers or other extraneous information) for use as part of a label format.

You can use ZebraDesigner or ZebraNet Bridge Enterprise to convert the .PCX graphic format into the pure hexadecimal .GRF graphic format. Hexadecimal data may also be directly input as part of a ZPL II program. Manually preparing a string of HEX code is possible but usually impractical.

Alternative Data Compression Scheme for ~DG and ~DB Commands

There is an alternative data compression scheme recognized by the Zebra printer. This scheme further reduces the actual number of data bytes and the amount of time required to download graphic images and bitmapped fonts with the ~DG and ~DB commands.

The following represent the repeat counts 1, 2, 3, 4, 5, ..., 19 on a subsequent Hexadecimal value. Values start with G since 0 through 9 and A through F are already used for HEX values.)

G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	1

These numbers represent the repeat counts 20, 40, 60, 80,....400 on a subsequent hexadecimal value.

g	h	I	j	k	l	m	n	o	p
20	40	60	80	100	120	140	160	180	200
r	s	t	u	v	w	x	y	z	
240	260	280	300	320	340	360	380	400	

Sending M6 to the printer is identical to sending the following hexadecimal data:

```
6666666
```

The M has the value of 7. Therefore M6 sends seven (7) hexadecimal 6's.

Sending hB to the printer is identical to sending the following hexadecimal data:

```
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
```

The h has a value of 40. Therefore, hB sends 40 Hexadecimal B's.

Repeat Values

Several repeat values can be used together to achieve any desired value.

Several repeat values can be used together to achieve any value desired. vMB or MvB will send 327 hexadecimal B's to the printer.

- a comma (,) fills the line, to the right, with zeros (0) until the specified line byte is filled.
- an exclamation mark (!) fills the line, to the right, with ones (1) until the specified line byte is filled.
- a colon (:) denotes repetition of the previous line.

Recalling a Hexadecimal Graphic Image

The ^XG (Recall Graphic) command is used to recall one or more graphic images for printing. This command is used in a label format to merge pictures such as company logos and piece parts, with text data to form a complete label.

An image may be recalled and resized as many times per format as needed. Other images and data may be added to the format.

Reducing Download Time of Graphic Images

There is a method of reducing the actual number of data bytes sent to the printer when using the ~DG command.

If the HEX string ends in an even number of zeros (0's), a single comma (,) can be substituted for ALL of the zeros. If the HEX string ends in an odd number of zeros, one zero and a single comma is required. The exclamation mark (!) and the colon (:) described under Repeat Values can also be used.

NOTE: The text rows in your editor may not be the same as the dot rows used by ZPL II. The editor may word wrap or truncate the dot rows. ZPL II ignores the end of a text line (carriage returns and line feed characters).

Transferring Object Between Storage Devices

The ^TO (Transfer Object) command is used to copy an object or group of objects from one storage device to another. It is quite similar to the copy function used in personal computers.

Source and destination devices must be supplied and must be different and valid for the action specified. Invalid parameters will cause the command to be ignored.

There are no defaults associated with this command. However, the asterisk (*) may be used as a wild card for Object names and extensions. For instance, ZEBRA.* or *.GRF would be acceptable forms for use with ^TO command.

The Asterisk (*) can be used to transfer multiple object files (except *.FNT) from the DRAM to the Memory Card. For example, you have several object files that contain logos. These files are named LOGO1.GRF, LOGO2.GRF, and LOGO3.GRF.

You want to transfer all of these files to the Memory Card using the name NEW instead of LOGO. By placing an Asterisk (*) after both LOGO and NEW in the transfer command, you can copy all of these files with one command. The format for this would be as follows:

```
^XA
^TOR:LOGO*.GRF,B:NEW*.GRF
^XZ
```

NOTE: If, during a multiple transfer, a file is too big to be stored on the Memory Card, it will be skipped. All remaining files will be checked to see if they can be stored. Those that can be stored, will be stored.

Deleting Graphics from Memory

The ^ID (Item Delete) command deletes objects, images, fonts, and formats from storage areas selectively or in groups. This command can be used within a printing format to delete objects just prior to saving new ones or can be in a stand-alone type format simply to delete objects.

The object name and extension support the use of the asterisk (*) as a wildcard. This allows for easy deletion of selected groups of objects.

The following are various examples of using the ^ID command.

To delete just stored formats from DRAM:

```
^XA^IDR:*.ZPL^XZ
```

To delete formats and images named SAMPLE from DRAM regardless of the extension:

```
^XA^IDR:SAMPLE.*^XZ
```

To delete the image SAMPLE1.GRF prior to storing SAMPLE2.GRF:

```
^XA
^FO25,25^AD,18,10^FDDelete^FS
^FO25,45^AD,18,10^FDthen Save^FS
^IDR:SAMPLE1.GRF^FS
^ISR:SAMPLE2.GRF^FS
^XZ
```

To delete everything from DRAM:

```
^XA^IDR:*.*^XZ
```

Defining and Using the AUTOEXEC.ZPL Function

An AUTOEXEC.ZPL file function is supported by the printer. It functions in much the same way as the AUTOEXEC.BAT file in MS-DOS.

The AUTOEXEC.ZPL file function can be used for setting up various parameters at the time the printer is powered up (such as ^COY, ^LL, ^CWf). The function can also be recalled at any time after power up.

This file must initially be in the extra EPROM, FLASH, or PCMCIA memory. When the printer is powered on, it looks to the extra memory site for the stored format called AUTOEXEC.ZPL. If found, the contents of the file are automatically executed as a stored format.

This is an example of an `autoexe.zpl` file:

```
^XA
^DFE:AUTOEXEC.ZPL^FS
^SEE:JIS.DAT^FS
^CW1,E:ANMDJ.TTF^FS
^XZ
```

Memory, Flash Cards, and Font Cards

Zebra printers come with a variety of memory device, including DRAM, EPROM, PCMCIA, Flash, socket Flash, and battery backed-up RAM.

NOTE: Not all memory options are available on all printers.

Most Zebra printers allow you to print a printer configuration label, which will show the letter designation assigned to your printer memory options. For printer models that do not support this feature, use the table below to see how the memory IDs are assigned. Memory IDs default to these values when the printer is reset to factory defaults.

Table 48 Letter Designations for Different Memory Options

Memory Option	Default Letter Designation
EPROM	E:
PCMCIA	B:
Flash	E:

Table 48 Letter Designations for Different Memory Options (Continued)

Memory Option	Default Letter Designation
DRAM	R:
Battery backed-up RAM	B: or E:
Socket Flash	B:
Compact Flash	A:

A few ZPL II commands directly affect the types of memory available to Zebra printers. These commands are ~JB, ^JB and ~HM

Table 49 Commands that Affect Available Memory Types

Command	Function
~JB (Reset Battery Dead)	This command is sent to the printer if either of these conditions exist: • If the B: memory card is intentionally cleared (reinitialized). • If the battery supplying power to the Battery Powered Font Card fails and is replaced. (A bad battery would show a “battery dead” condition on the printer configuration label.) **NOTE:** If you replace the battery but do not send this command to the printer, the Battery Powered Font Card will not function.
^JB (Initialize Flash Memory)	This command is used to initialize the two types of Flash Memory available in the Zebra printers. **NOTE:** Link-OS printers use an automatic memory management system that eliminates the need to manually initialize the Flash Memory system.
~HM (Host Memory Status)	Sending this command to the printer immediately returns a memory status message to the host. Use this command whenever you need to know the status of the memory. When the Host Memory Status Command, ~HM, is sent to the Zebra printer, a line of data containing three numbers is sent back to the Host. The following is an example: `1024,0780,0780` • The first value is the total amount of RAM (Random Access Memory) installed in the printer. This number is in Kilobytes. • The second value is the maximum amount of RAM available to the user. This number is in Kilobytes. • The third value is the amount of RAM currently available to the user. This number is in Kilobytes.

Shortcuts and Alternate Schemes for Writing ZPL II Scripts

ZPL II programming scripts can be written in a variety of ways. There are, however, more efficient ways to write a ZPL II script depending on the application and the commands used. The following are certain ways to write the same ZPL II script, each yielding the same result.

The Code 39 bar code shows the ZPL II script written like this:

```
^XA^FO100,75^BY3
^B3N,N,100,Y,N
^FD123ABC^XZ
```

Since it is only one field, however, the entire command can be written as a one line entry:

```
^XA^FO100,75^BY3^B3N,N,100,Y,N^FD123ABC^XZ
```

Finally, this script can be further simplified by writing it on one line, using the comma (,) delimiter to reduce the default parameters in the ^B3 command and eliminating the default parameters at the end of the ^B3 command:

```
^XA^FO100,75^BY3^B3,,100^FD123ABC^XZ
```

You might write your ZPL II scripts in any way that makes sense to you. Some programmers prefer to write out each format command and field on a line by line basis like this:

```
^XA
^PR2^FS
^LL935^FS
^LH30,30^FS
^FO20,10^AF^FDZEBRA^FS
^FO20,60^B3,,40^FDAA001^FS
^FO20,180^AF^SNSERIAL NUMBER 00000000111,1,Y^FS
^PQ10^FS
^XZ
```

Although this script will print with no problems, it contains unnecessary ^FS (Field Separator) commands which have been placed after the format commands. Some programmers feel it is required to place a ^FS command at the end of each line, but the ^FS command is only needed to separate specific fields. Therefore, the script would transmit more quickly written like this:

```
^XA
^PR2
^LL935
^LH30,30
^FO20,10^AF^FDZEBRA^FS
^FO20,60^B3,,40^FDAA001^FS
^FO20,180^AF^SNSERIAL NUMBER 00000000111,1,Y^FS
^PQ10
^XZ
```

Other programmers prefer to keep the format commands on one line as an organizational preference, like this:

```
^XA^PR2^LL935^LH30,30
^FO20,10^AF^FDZEBRA^FS
^FO20,60^B3,,40^FDAA001^FS
^FO20,180^AF^SNSERIAL NUMBER 00000000111,1,Y^FS
^PQ10^XZ
```

The label will print out the same so you should develop a scripting pattern that suits your own organizational style but one which is efficient and is concerned with keeping transmission times to a minimum.

Font Shortcuts

There are times when you might include a specific font into your script and use it repeatedly within different fields.

The following is an example of one way to write this script:

```
^XA
^FO120,108^A0N,89^FDA Guide to^FS
^FO120,207^A0N,89^FDZPL II^FS
^FO120,306^A0N,89^FDProgramming^FS
^FO120,405^A0N,89^FDLanguage^FS
^XZ
```

Notice that the ^FS command is used on the second to last line to close the field. Actually, it is unnecessary because the ^XZ will accomplish the same thing, so we can remove it from our script. Also, since the font and font size are not changing within the fields, this script can be simplified for quicker transmission by removing the unnecessary font entries and listing the font information once using the ^CF command (see ^CF):

```
^XA
^CF0,89
^FO120,108^FDA Guide to^FS
^FO120,207^FDZPL II^FS
^FO120,306^FDProgramming^FS
^FO120,405^FDLanguage
^XZ
```

This script can be made even more efficient by including the ^FB command to identify the left origin of the text which occurs at the same place each time. For details, see ^FB:

```
^XA
^CF0,89
^FO120,108
^FB800,6
^FDA Guide to\&ZPL II\&Programming\&Language
^XZ
```

NOTE: The entries "\&" within the text indicate a carriage return/line feed as allowed by the ^FB command. For details, see ^FB.

If you wanted to change the font type or size within the script, however, you would need to include the specific font parameters within the field where the change occurs. In this case, you would not want to use the ^FB command because the change in font size (in our example below) will affect the y-axis (up and down) position of the text.

You can still use the ^CF command, but you will need to include the specific font information on the line where the change in the field occurs:

```
^XA
^CF0,89
^FO120,108^FDA Guide to^FS
^FO120,207^FDZPL II^FS
^FO120,306^A0N,110^FDProgramming^FS
^FO120,426^FDLanguage
^XZ
```

PDFium License

PDFium license information

Copyright 2014 PDFium Authors. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met.

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of Google Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS

AS IS AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT

LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR

A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT

OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,

SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT

LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,

DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY

THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT

(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE

OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.