



Programming Guide

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



© **2009 ZIH Corp.** The copyrights in this manual and the software and/or firmware in the printer described therein are owned by ZIH Corp. and Zebra's licensors. Unauthorized reproduction of this manual or the software and/or firmware in the printer may result in imprisonment of up to one year and fines of up to \$10,000 (17 U.S.C.506). Copyright violators may be subject to civil liability.

This product may contain ZPL[®], ZPL II[®], and ZebraLink[™] programs; Element Energy Equalizer[™] Circuit; E^{3™}; and Monotype Imaging fonts. Software © ZIH Corp. All rights reserved worldwide.

ZebraLink, Element Energy Equalizer, E³ and all product names and numbers are trademarks, and Zebra, the Zebra head graphic, ZPL and ZPL II are registered trademarks of ZIH Corp. All rights reserved worldwide.

All other brand names, product names, or trademarks belong to their respective holders. For additional trademark information, please see "Trademarks" on the product CD.

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 Corporation.

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

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

Limitation of Liability In no event shall Zebra Technologies Corporation 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 Corporation 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



Functional List of ZPL Commands	21
Alphabetical List of ZBI Commands	25
Alphabetical List of Set/Get/Do Commands	27
About This Document	31
Who Should Use This Document	32
How This Document Is Organized	32
Contacts	33
Document Conventions	34
Introduction	37
ZPL Commands	39
Basic ZPL Exercises and Examples	41
Before you begin	41
^A Scalable/Bitmapped Font	51
^A@ Use Font Name to Call Font	53
^B0 Aztec Bar Code Parameters	55
^B1 Code 11 Bar Code	57
^B2 Interleaved 2 of 5 Bar Code	59
^B3 Code 39 Bar Code	61
^B4 Code 49 Bar Code	65
^B5 Planet Code bar code	69
^B7 PDF417 Bar Code	70
^B8 EAN-8 Bar Code	74
^B9 UPC-E Bar Code	76
^BA Code 93 Bar Code	78

^BB	CODABLOCK Bar Code	82
^BC	Code 128 Bar Code (Subsets A, B, and C)	86
^BD	UPS MaxiCode Bar Code	98
^BE	EAN-13 Bar Code	101
^BF	MicroPDF417 Bar Code	103
^BI	Industrial 2 of 5 Bar Codes	106
^BJ	Standard 2 of 5 Bar Code	108
^BK	ANSI Codabar Bar Code	110
^BL	LOGMARS Bar Code	112
^BM	MSI Bar Code	114
^B0	Aztec Bar Code Parameters	116
^BP	Plessey Bar Code	118
^BQ	QR Code Bar Code	120
^BR	GS1 Databar (formerly Reduced Space Symbology)	128
^BS	UPC/EAN Extensions	130
^BT	TLC39 Bar Code	133
^BU	UPC-A Bar Code	135
^BX	Data Matrix Bar Code	137
^BY	Bar Code Field Default	142
^BZ	POSTAL Bar Code	144
^CC	~CC Change Caret	146
^CD	~CD Change Delimiter	147
^CF	Change Alphanumeric Default Font	148
^CI	Change International Font/Encoding	149
^CM	Change Memory Letter Designation	153
^CN	Cut Now	154
^CO	Cache On	155
^CP	Remove Label	157
^CT	~CT Change Tilde	158
^CV	Code Validation	159
^CW	Font Identifier	161
~DB	Download Bitmap Font	163
~DE	Download Encoding	165
^DF	Download Format	167
~DG	Download Graphics	168
~DN	Abort Download Graphic	171
~DS	Download Intellifont (Scalable Font)	172
~DT	Download Bounded TrueType Font	173
~DU	Download Unbounded TrueType Font	174
~DY	Download Objects	175

~EG Erase Download Graphics	179
^FB Field Block	180
^FC Field Clock	183
^FD Field Data	184
^FH Field Hexadecimal Indicator	185
^FL Font Linking	187
^FM Multiple Field Origin Locations	190
^FN Field Number	193
^FO Field Origin	194
^FP Field Parameter	195
^FR Field Reverse Print	196
^FS Field Separator	197
^FT Field Typeset	198
^FV Field Variable	200
^FW Field Orientation	201
^FX Comment	203
^GB Graphic Box	204
^GC Graphic Circle	206
^GD Graphic Diagonal Line	207
^GE Graphic Ellipse	208
^GF Graphic Field	209
^GS Graphic Symbol	211
~HB Battery Status	213
~HD Head Diagnostic	214
^HF Host Format	215
^HG Host Graphic	216
^HH Configuration Label Return	217
~HI Host Identification	218
~HM Host RAM Status	219
~HQ Host Query	220
~HQ Examples	223
~HS Host Status Return	228
^HT Host Linked Fonts List	232
~HU Return ZebraNet Alert Configuration	233
^HV Host Verification	234
^HW Host Directory List	235
^HY Upload Graphics	237
^HZ Display Description Information	238
^ID Object Delete	240
^IL Image Load	242

^IM Image Move	244
^IS Image Save	245
~JA Cancel All	247
^JB Initialize Flash Memory	248
~JB Reset Optional Memory	249
~JC Set Media Sensor Calibration	250
~JD Enable Communications Diagnostics	251
~JE Disable Diagnostics	252
~JF Set Battery Condition	253
~JG Graphing Sensor Calibration	254
^JH Early Warning Settings	255
^JI Start ZBI (Zebra BASIC Interpreter)	258
~JI Start ZBI (Zebra BASIC Interpreter)	260
^JJ Set Auxiliary Port	261
~JL Set Label Length	263
^JM Set Dots per Millimeter	264
~JN Head Test Fatal	265
~JO Head Test Non-Fatal	266
~JP Pause and Cancel Format	267
~JQ Terminate Zebra BASIC Interpreter	268
~JR Power On Reset	269
^JS Sensor Select	270
~JS Change Backfeed Sequence	271
^JT Head Test Interval	273
^JU Configuration Update	274
^JW Set Ribbon Tension	275
~JX Cancel Current Partially Input Format	276
^JZ Reprint After Error	277
~KB Kill Battery (Battery Discharge Mode)	278
^KD Select Date and Time Format (for Real Time Clock)	279
^KL Define Language	280
^KN Define Printer Name	281
^KP Define Password	283
^KV Kiosk Values	284
^LF List Font Links	286
^LH Label Home	287
^LL Label Length	288
^LR Label Reverse Print	289
^LS Label Shift	290
^LT Label Top	291

^MA Set Maintenance Alerts	292
^MC Map Clear	294
^MD Media Darkness	295
^MF Media Feed	296
^MI Set Maintenance Information Message	297
^ML Maximum Label Length	298
^MM Print Mode	299
^MN Media Tracking	301
^MP Mode Protection	302
^MT Media Type	304
^MU Set Units of Measurement	305
^MW Modify Head Cold Warning	307
^NC Select the Primary Network Device	308
~NC Network Connect	309
^ND Change Network Settings	310
^NI Network ID Number	312
~NR Set All Network Printers Transparent	313
^NS Change Wired Networking Settings	314
~NT Set Currently Connected Printer Transparent	316
^PA Advanced Text Properties	317
^PF Slew Given Number of Dot Rows	318
^PH ~PH Slew to Home Position	319
~PL Present Length Addition	320
^PM Printing Mirror Image of Label	321
^PN Present Now	322
^PO Print Orientation	323
^PP ~PP Programmable Pause	324
^PQ Print Quantity	325
^PR Print Rate	326
~PR Applicator Reprint	328
~PS Print Start	329
^PW Print Width	330
~R0 Reset Advanced Counter	331
^SC Set Serial Communications	332
~SD Set Darkness	333
^SE Select Encoding Table	334
^SF Serialization Field (with a Standard ^FD String)	335
^SI Set Sensor Intensity	340
^SL Set Mode and Language (for Real-Time Clock)	341
^SN Serialization Data	343

^S0 Set Offset (for Real-Time Clock)	346
^SP Start Print	347
^SQ Halt ZebraNet Alert	349
^SR Set Printhead Resistance	351
^SS Set Media Sensors	352
^ST Set Date and Time (for Real-Time Clock)	354
^SX Set ZebraNet Alert	355
^SZ Set ZPL	357
~TA Tear-off Adjust Position	358
^TB Text Blocks	359
^TO Transfer Object	360
~WC Print Configuration Label	362
^WD Print Directory Label	363
~WQ Write Query	365
~WQ Examples	368
^XA Start Format	373
^XB Suppress Backfeed	374
^XF Recall Format	375
^XG Recall Graphic	376
^XS Set Dynamic Media Calibration	377
^XZ End Format	378
^ZZ Printer Sleep	379
ZPL RFID Commands	381
RFID Command Overview	382
Printer and Firmware Compatibility	383
^HL or ~HL Return RFID Data Log to Host	385
^HR Calibrate RFID Transponder Position	386
^RA Read AFI or DSFID Byte	390
^RB Define EPC Data Structure	392
^RE Enable/Disable E.A.S. Bit	394
^RF Read or Write RFID Format	395
^RI Get RFID Tag ID	398
^RM Enable RFID Motion	399
^RN Detect Multiple RFID Tags in Encoding Field	400
^RQ Quick Write EPC Data and Passwords	401
^RR Specify RFID Retries for a Block	403
^RS Set Up RFID Parameters	404
^RT Read RFID Tag	411
~RV Report RFID Encoding Results	414

^RW Set RF Power Levels for Read and Write	415
^RZ Set RFID Tag Password and Lock Tag	417
^WF Encode AFI or DSFID Byte	420
^WT Write (Encode) Tag	422
^WV Verify RFID Encoding Operation	424
ZPL Wireless Commands	425
^KC Set Client Identifier (Option 61)	426
^NB Search for Wired Print Server during Network Boot	427
^NN Set SNMP	428
^NP Set Primary/Secondary Device	429
^NT Set SMTP	430
^NW Set Web Authentication Timeout Value	431
^WA Set Antenna Parameters	432
^WE Set WEP Mode	433
^WI Change Wireless Network Settings	435
^WL Set LEAP Parameters	437
~WL Print Network Configuration Label	438
^WP Set Wireless Password	440
^WR Set Transmit Rate	441
~WR Reset Wireless Radio Card and Print Server	442
^WS Set Wireless Radio Card Values	443
^WX Configure Wireless Securities	445
Supporting Parameters for Different Security Types	449
ZBI Commands	455
Command and Function Reference Format	456
Function Rules	456
Introduction to Zebra Basic Interpreter (ZBI)	458
What is ZBI and why is it for me?	458
Printers, ZBI Keys, & ZBI Versions	458
Section Organization	460
Writing ZBI Programs	460
Editing Commands	461
Running and Debugging Commands	469

Base Types and Expressions	482
Variable Names	482
Variable Declarations	483
Constants	483
Arrays	483
Assignment	484
Numeric Expressions	486
String Concatenation (&)	488
Sub-strings	488
Boolean Expressions	489
Combined Boolean Expressions	491
Control and Flow	492
Input and Output	500
Available Ports	500
Creating Connections	500
Reading and Writing	508
Port Usage Examples	518
File System	525
Runtime Access	526
Comma Separated Values (CSV)	532
Accessing Comma Separated Value (CSV) and Text File Functions	532
Events	538
Available Events	538
ZBI Key Names	540
Systems	549
Applicator Functions	555
String Functions	561
Math Functions	579
Array Functions	588
Time and Date Functions	597
Set/Get/Do Interactions	602
Example Programs	605
Array Program	605
CSV Program	606
DPI Conversion Program	611
Email Program	614
Extraction 1 Program	616
Extraction 2 Program	617
Front Panel Control	620
Recall Program	622
Scale Program	624

SGD Printer Commands	625
Overview	626
setvar Command	626
getvar Command	626
do Command	627
Command Structure	628
How to Send Multiple SGD Commands	628
appl.option_board_version	629
appl.bootblock	630
appl.name	631
CISDFCRC16	632
Download Files	632
cutter.clean_cutter	635
device.download_connection_timeout	636
device.friendly_name	637
device.frontpanel.key_press	638
device.frontpanel.line1	639
device.frontpanel.line2	640
device.frontpanel.xml	641
device.jobs_print	642
device.languages	643
device.orientation	644
device.pnp_option	645
device.reset	646
device.restore_defaults	647
device.unique_id	648
device.uptime	649
device.user_p1	650
device.user_p2	651
device.xml.enable	652
display.text	653
file.delete	654
file.dir	655
file.type	656
file.run	657
head.latch	658
interface.network.active.gateway	659
interface.network.active.ip_addr	660
interface.network.active.mac_addr	661
interface.network.active.mac_raw	662
interface.network.active.netmask	663
interface.network.active.protocol	664
media.cut_now	665
media.darkness_mode	666

media.media_low.external	667
media.media_low.warning	668
media.present.cut_amount	669
media.present.eject	670
media.present.length_addition	671
media.present.loop_length	672
media.present.loop_length_max	673
media.present.cut_margin	674
media.present.present_timeout	675
media.present.present_type	676
media.printmode	677
media.speed	679
odometer.headclean	680
odometer.headnew	681
odometer.label_dot_length	682
odometer.media_marker_count1	683
odometer.media_marker_count2	684
odometer.retracts_count	685
odometer.total_print_length	686
print.tone	687
rfid.error.response	688
rfid.position.program	689
rfid.reader_1.antenna_port	690
rfid.reader_1.power.read	691
rfid.reader_1.power.single_power	692
rfid.reader_1.power.write	693
zbi.control.add_breakpoint	694
zbi.control.break	695
zbi.control.clear_breakpoints	696
zbi.control.delete_breakpoint	697
zbi.control.line_number	698
zbi.control.restart	699
zbi.control.run	700
zbi.control.step	701
zbi.control.terminate	702
zbi.control.variable_name	703
zbi.control.variable_value	704
zbi.key	705
zbi.last_error	706
zbi.reseller_key	707
zbi.revision	708
zbi.running_program_name	709
zbi.start_info.execute	710
zbi.start_info.file_name	711
zbi.start_info.memory_alloc	712
zbi.state	713

SGD Wired Commands	715
Overview	716
setvar Command	716
getvar Command	716
do Command	717
Command Structure	718
How to Send Multiple SGD Commands	718
external_wired.check	719
external_wired.ip.addr	720
external_wired.ip.arp_interval	721
external_wired.ip.default_addr_enable	722
external_wired.ip.dhcp.cid_all	723
external_wired.ip.dhcp.cid_enable	724
external_wired.ip.dhcp.cid_prefix	725
external_wired.ip.dhcp.cid_suffix	726
external_wired.ip.dhcp.cid_type	727
external_wired.ip.gateway	728
external_wired.ip.netmask	729
external_wired.ip.port	730
external_wired.ip.protocol	731
external_wired.ip.timeout.enable	732
external_wired.ip.timeout.value	733
external_wired.ip.v6.addr	734
external_wired.ip.v6.gateway	735
external_wired.ip.v6.prefix_length	736
external_wired.mac_addr	737
external_wired.mac_raw	738
internal_wired.auto_switchover	739
internal_wired.ip.addr	741
internal_wired.ip.arp_interval	742
internal_wired.ip.default_addr_enable	743
internal_wired.ip.dhcp.cache_ip	744
internal_wired.ip.dhcp.cid_all	745
internal_wired.ip.dhcp.cid_enable	746
internal_wired.ip.dhcp.cid_prefix	747
internal_wired.ip.dhcp.cid_suffix	748
internal_wired.ip.dhcp.cid_type	749
internal_wired.ip.dhcp.option12	754
internal_wired.ip.dhcp.option12_format	755
internal_wired.ip.gateway	760
internal_wired.ip.netmask	761
internal_wired.ip.port	762
internal_wired.ip.protocol	763
internal_wired.ip.timeout.enable	764

internal_wired.ip.timeout.value	765
internal_wired.mac_addr	766
internal_wired.mac_raw	767
SGD Wireless Commands	769
Overview	770
setvar Command	770
getvar Command	770
do Command	771
Command Structure	772
How to Send Multiple SGD Commands	772
bluetooth.address	773
bluetooth.afh_map	774
bluetooth.afh_map_curr	775
bluetooth.afh_mode	776
bluetooth.authentication	777
bluetooth.bluetooth_pin	778
bluetooth.date	779
bluetooth.discoverable	780
bluetooth.enable	781
bluetooth.friendly_name	782
bluetooth.local_name	783
bluetooth.radio_auto_baud	784
bluetooth.radio_version	785
bluetooth.short_address	786
bluetooth.version	787
card.mac_addr	788
card.inserted	789
comm.type	790
ip.active_network	791
ip.addr	792
ip.arp_interval	793
ip.bootp.enable	794
ip.dhcp.cache_ip	795
ip.dhcp.cid_all	796
ip.dhcp.cid_enable	797
ip.dhcp.cid_prefix	798
ip.dhcp.cid_suffix	799
ip.dhcp.lease.length	800
ip.dhcp.cid_type	801
ip.dhcp.enable	802
ip.dhcp.lease.last_attempt	803
ip.dhcp.lease.server	804
ip.dhcp.lease.time_left	805
ip.dhcp.option12	806

ip.dhcp.option12_format	807
ip.dhcp.option12_value	808
ip.dhcp.request_timeout	809
ip.dhcp.requests_per_session	810
ip.dhcp.session_interval	811
ip.dns.domain	812
ip.dns.servers	813
ip.ftp.enable	814
ip.ftp.execute_file	815
ip.gateway	816
ip.http.enable	817
ip.lpd.enable	818
ip.mac_raw	819
ip.mirror.auto	820
ip.mirror.error_retry	821
ip.mirror.feedback.auto	822
ip.mirror.feedback.freq	823
ip.mirror.feedback.odometer	824
ip.mirror.feedback.path	825
ip.mirror.fetch	826
ip.mirror.freq	827
ip.mirror.freq_hours	828
ip.mirror.last_error	829
ip.mirror.last_time	830
ip.mirror.password	831
ip.mirror.path	832
ip.mirror.server	833
ip.mirror.success	834
ip.mirror.success_time	835
ip.mirror.username	836
ip.mirror.version	837
ip.netmask	838
ip.pop3.enable	839
ip.pop3.password	840
ip.pop3.poll	841
ip.pop3.server_addr	842
ip.pop3.username	843
ip.port	844
ip.primary_network	845
ip.smtp.domain	846
ip.smtp.enable	847
ip.smtp.server_addr	848
ip.snmp.get_community_name	849
ip.snmp.set_community_name	850
ip.snmp.enable	851

ip.telnet.enable	852
ip.tcp.enable	853
ip.udp.enable	854
wlan.adhocautomode	855
wlan.adhocchannel	856
wlan.associated	857
wlan.bssid	858
wlan.channel	859
wlan.channel_mask	860
wlan.current_tx_rate	861
wlan.essid	862
wlan.firmware_version	863
wlan.ip.addr	864
wlan.ip.arp_interval	865
wlan.ip.default_addr_enable	866
wlan.ip.dhcp.cache_ip	867
wlan.ip.dhcp.cid_all	868
wlan.ip.dhcp.cid_enable	869
wlan.ip.dhcp.cid_prefix	870
wlan.ip.dhcp.cid_suffix	871
wlan.ip.dhcp.cid_type	872
wlan.ip.dhcp.lease.last_attempt	873
wlan.ip.dhcp.lease.length	874
wlan.ip.dhcp.lease.server	875
wlan.ip.dhcp.lease.time_left	876
wlan.ip.dhcp.option12	877
wlan.ip.dhcp.option12_format	878
wlan.ip.dhcp.option12_value	879
wlan.ip.dhcp.request_timeout	880
wlan.ip.dhcp.requests_per_session	881
wlan.ip.dhcp.session_interval	882
wlan.ip.gateway	883
wlan.ip.netmask	884
wlan.ip.port	885
wlan.ip.protocol	886
wlan.ip.timeout.enable	887
wlan.ip.timeout.value	888
wlan.keep_alive.enable	889
wlan.keep_alive.timeout	890
wlan.kerberos.kdc	891
wlan.kerberos.password	892
wlan.kerberos.realm	893
wlan.kerberos.username	894
wlan.mac_addr	895
wlan.mac_raw	896

wlan.operating_mode	897
wlan.password	898
wlan.preamble	899
wlan.private_key_password	900
wlan.roam.interchannel_delay	901
wlan.roam.interval	902
wlan.roam.max_chan_scan_time	903
wlan.roam.signal	904
wlan.security	905
Supporting SGD Commands for Different Security Types	907
wlan.signal_noise	914
wlan.signal_quality	915
wlan.signal_strength	916
wlan.station_name	917
wlan.tx_power	918
wlan.tx_rate	919
wlan.username	920
wlan.wep.auth_type	921
wlan.wep.index	922
wlan.wep.key1	923
wlan.wep.key2	924
wlan.wep.key3	925
wlan.wep.key4	926
wlan.wpa.psk	927
wlan.wep.key_format	928
Zebra Code Pages	929
Zebra Code Page 850 — Latin Character Set	930
Zebra Code Page 1250 (Scalable/Downloaded TTF) —Central and Eastern European Latin Character Set	932
.....	933
Zebra Code Page 1250 — (Bitmap Fonts) Central and Eastern European Latin Character Set	934
Zebra Code Page 1252 — Latin Character Set	936
Zebra Code Page 1253 — Modern Greek Character Set	938
Zebra Code Page 1254 — Turkish Character Set	940
Zebra Code Page 1255 — Hebrew Character Set	942
ASCII	945
ASCII Code Chart	946
Fonts and Bar Codes	947
Standard Printer Fonts	948
Proportional and Fixed Spacing	950

Scalable Versus Bitmapped Fonts	952
Scalable Fonts	952
Bitmapped Fonts	952
Font Matrices	954
Bar Codes	956
Basic Format for Bar Codes	956
Bar Code Field Instructions	957
Bar Code Command Groups	959
Mod 10 and Mod 43 Check Digits	961
Mod 10 Check Digit	962
Mod 43 Check Digit	963
ZB64 Encoding and Compression	965
Introduction to B64 and Z64	966
B64 and Z64 Encoding	968
Field Interactions	971
Normal Orientation	972
Rotated Orientation	973
Bottom Up Orientation	974
Inverted Orientation	975
Real Time Clock	977
Overview	978
Control Panel Programming	979
Real Time Clock Parameters	979
RTC Date	980
RTC Time	981
RTC General Information	982
First Day of the Week Affects Calendar Week	984
Time and Date Precision	987
ZPL II Samples	989
ZBI Character Set	993
Character Set	994
SGD Command Support	997
Printer and Firmware Compatibility	998
Printer Type	998

Features	1017
Firmware x.17	1018
Firmware x.16	1019
Firmware x.15	1021
Wireless Securities	1021
ZPL and SGD Commands	1022
SNMP	1022
Set / Get / Do Support	1022
XML-Enabled Printing	1022
Firmware x.14	1023
Global Printing Solution	1024
Mirror	1027
Mirror Overview	1028
Benefits	1028
Professional Services for Mirror Configuration	1030
Requirements	1030
How Mirror Works	1032
Mirror Process Summary	1032
Mirror Process Details	1033
Creating ZPL Files for Use in the "<update-root>/files" Directory	1034
One Format per File	1034
Character Substitution	1035
Removing the ^XA and ^XZ commands	1036
Removing the ^DF command	1036
Example Files	1037
File Naming Recommendations	1037
Command Use Recommendations	1037
Configuration	1038
Mirror FTP Server Configuration	1038
Mirror Printer Configuration	1038
The Feedback.get File	1039
Example Feedback.get file	1040
How to Set Up and Use Mirror	1042
Scenario One	1042
Scenario Two	1043
Troubleshooting	1044

Wireless Markup Language (WML)	1047
WML Overview	1048
WML Details	1048
Supported Printers	1049
Professional Services for WML Content Creation	1050
WML Tags	1050
Using WML	1050
Send WML Content to the Printer via FTP	1052
Resetting the "ip.ftp.execute_file setting	1052
Sending WML Content to the Printer via the CISDFCRC16 Command:	1053
Retrieving WML Content from the Printer using the file.type Command:	1054
Using .nrd Files from WML Menus	1054
Removing WML or .nrd Files from the Printer using the file.delete Command:	1054
WML Examples	1055
Example 1	1055
Example 2	1056
Example 3	1057
Example 4	1059
Example 5	1060
Troubleshooting Scenarios	1063
Glossary	1065
Index	1069

Functional List of ZPL Commands



Abort Download Graphic	171	Define Language	280
Advanced Text Properties	317	Define Password	283
ANSI Codabar Bar Code	110	Define Printer Name	281
Applicator Reprint	328	Detect Multiple RFID Tags in Encoding Field .	400
Aztec Bar Code Parameters	116	Disable Diagnostics	252
Aztec Bar Code Parameters	55	Display Description Information	238
Bar Code Field Default	142	Download Bitmap Font	163
Battery Status	213	Download Bounded TrueType Font	173
Cache On	155	Download Encoding	165
Calibrate RFID Transponder Position	386	Download Files	632
Cancel All	247	Download Format	167
Cancel Current Partially Input Format	276	Download Graphics	168
Change Alphanumeric Default Font	148	Download Intellifont (Scalable Font)	172
Change Backfeed Sequence	271	Download Objects	175
Change Caret	146	Download Unbounded TrueType Font	174
Change Delimiter	147	EAN-13 Bar Code	101
Change International Font/Encoding	149	EAN-8 Bar Code	74
Change Memory Letter Designation	153	Early Warning Settings	255
Change Network Settings	310	Enable Communications Diagnostics	251
Change Tilde	158	Enable RFID Motion	399
Change Wired Networking Settings	314	Enable/Disable E.A.S. Bit	394
Change Wireless Network Settings	435	Encode AFI or DSFID Byte	420
CODABLOCK Bar Code	82	End Format	378
Code 11 Bar Code	57	Erase Download Graphics	179
Code 128 Bar Code (Subsets A, B, and C) .	86	Field Block	180
Code 39 Bar Code	61	Field Clock	183
Code 49 Bar Code	65	Field Data	184
Code 93 Bar Code	78	Field Hexadecimal Indicator	185
Code Validation	159	Field Number	193
Comment	203	Field Orientation	201
Configuration Label Return	217	Field Origin	194
Configuration Update	274	Field Parameter	195
Configure Wireless Securities	445	Field Reverse Print	196
Cut Now	154	Field Separator	197
Data Matrix Bar Code	137	Field Typeset	198
Define EPC Data Structure	392	Field Variable	200

Font Identifier	161	Object Delete	240
Font Linking	187	Pause and Cancel Format	267
Get RFID Tag ID	398	PDF417 Bar Code	70
Graphic Box	204	Planet Code bar code	69
Graphic Circle	206	Plessey Bar Code	118
Graphic Diagonal Line	207	POSTAL Bar Code	144
Graphic Ellipse	208	Power On Reset	269
Graphic Field	209	Present Length Addition	320
Graphic Symbol	211	Present Now	322
Graphing Sensor Calibration	254	Print Configuration Label	362
GS1 Databar (formerly Reduced Space Symbol- gy)	128	Print Directory Label	363
Halt ZebraNet Alert	349	Print Mode	299
Head Diagnostic	214	Print Network Configuration Label	438
Head Test Fatal	265	Print Orientation	323
Head Test Interval	273	Print Quantity	325
Head Test Non-Fatal	266	Print Rate	326
Host Directory List	235	Print Start	329
Host Format	215	Print Width	330
Host Graphic	216	Printer Sleep	379
Host Identification	218	Printing Mirror Image of Label	321
Host Linked Fonts List	232	Programmable Pause	324
Host Query	220	QR Code Bar Code	120
Host RAM Status	219	Quick Write EPC Data and Passwords	401
Host Status Return	228	Read AFI or DSFID Byte	390
Host Verification	234	Read or Write RFID Format	395
Image Load	242	Read RFID Tag	411
Image Move	244	Recall Format	375
Image Save	245	Recall Graphic	376
Industrial 2 of 5 Bar Codes	106	Remove Label	157
Initialize Flash Memory	248	Report RFID Encoding Results	414
Interleaved 2 of 5 Bar Code	59	Reprint After Error	277
Kill Battery (Battery Discharge Mode)	278	Reset Advanced Counter	331
Kiosk Values	284	Reset Optional Memory	249
Label Home	287	Reset Wireless Radio Card and Print Server	442
Label Length	288	Return RFID Data Log to Host	385
Label Reverse Print	289	Return ZebraNet Alert Configuration	233
Label Shift	290	Scalable/Bitmapped Font	51
Label Top	291	Search for Wired Print Server during Network Boot	427
List Font Links	286	Select Date and Time Format (for Real Time Clock)	279
LOGMARS Bar Code	112	Select Encoding Table	334
Map Clear	294	Select the Primary Network Device	308
Maximum Label Length	298	Sensor Select	270
Media Darkness	295	Serialization Data	343
Media Feed	296	Serialization Field (with a Standard ^FD String)	335
Media Tracking	301	Set All Network Printers Transparent	313
Media Type	304	Set Antenna Parameters	432
MicroPDF417 Bar Code	103	Set Auxiliary Port	261
Mode Protection	302	Set Battery Condition	253
Modify Head Cold Warning	307	Set Client Identifier (Option 61)	426
MSI Bar Code	114	Set Currently Connected Printer Transparent	316
Multiple Field Origin Locations	190	Set Darkness	333
Network Connect	309		
Network ID Number	312		

Set Date and Time (for Real-Time Clock) . . .	354
Set Dots per Millimeter	264
Set Dynamic Media Calibration	377
Set Label Length	263
Set LEAP Parameters	437
Set Maintenance Alerts	292
Set Maintenance Information Message	297
Set Media Sensor Calibration	250
Set Media Sensors	352
Set Mode and Language (for Real-Time Clock)	341
Set Offset (for Real-Time Clock)	346
Set Primary/Secondary Device	429
Set Printhead Resistance	351
Set RF Power Levels for Read and Write	415
Set RFID Tag Password and Lock Tag	417
Set Ribbon Tension	275
Set Sensor Intensity	340
Set Serial Communications	332
Set SMTP	430
Set SNMP	428
Set Transmit Rate	441
Set Units of Measurement	305
Set Up RFID Parameters	404
Set Web Authentication Timeout Value	431
Set WEP Mode	433
Set Wireless Password	440
Set Wireless Radio Card Values	443
Set ZebraNet Alert	355
Set ZPL	357
Slew Given Number of Dot Rows	318
Slew to Home Position	319
Specify RFID Retries for a Block	403
Standard 2 of 5 Bar Code	108
Start Format	373
Start Print	347
Start ZBI (Zebra BASIC Interpreter)	258
Start ZBI (Zebra BASIC Interpreter)	260
Suppress Backfeed	374
Tear-off Adjust Position	358
Terminate Zebra BASIC Interpreter	268
Text Blocks	359
TLC39 Bar Code	133
Transfer Object	360
UPC/EAN Extensions	130
UPC-A Bar Code	135
UPC-E Bar Code	76
Upload Graphics	237
UPS MaxiCode Bar Code	98
Use Font Name to Call Font	53
Verify RFID Encoding Operation	424
Write (Encode) Tag	422
Write Query	365



Notes •



Alphabetical List of ZBI Commands

! (EXCLAMATION MARK)	464	IF Statements	493
ACCEPT	508	INBYTE	514
ADDBREAK	479	INPUT	510
AUTONUM	466	INSERTROW	591
AUXPORT_GETPIN	559	INTTOHEX\$	586
AUXPORT_RELEASEPIN	560	ISERROR	550
AUXPORT_SETPIN	558	ISWARNING	551
AUXPORT_STEALPIN	556	LCASE\$	562
BREAK	478	LEN	578
CHR\$	563	LET	486
CLIENTSOCKET	507	LIST	465
CLOSE	503	LOAD	529
CLRERR	554	LTRIM\$	564
COLUMNSIZE	594	MAX	581
CSVLOAD	533	MAXNUM	583
CSVSTORE	535	MIN	582
CTRL-C	472	MOD	584
DATAREADY	504	NEW	462
DATE	600	ON ERROR	555
DATE\$	598	OPEN	502
DEBUG	476	ORD	576
DELBREAK	480	OUTBYTE	513
DELETE	531	POS	577
DELROW	592	PRINT	512
DIR	530	READ	515
DO Loops	494	REDIM	589
ECHO	468	REGISTEREVENT	543
END	499	REM	463
EXIT	498	RENUM	467
EXTRACT\$	574	REPEAT\$	565
FIND	595	RESTART	473
FOR Loops	495	ROWSIZE	593
GETVAR\$	604	RTRIM\$	566
GOTO/GOSUB	496	RUN	471
HANDLEEVENT	546	SEARCHTO\$	517
HEXTPOINT	587	SERVERCLOSE	506

SERVERTOCKET	505
SETERR	553
SETVAR	603
SLEEP	552
SPLIT	567
SPLITCOUNT	572
STEP	475
STORE	528
STR\$	580
SUB	497
TIME	601
TIME\$	599
TRACE	477
TRIGGEREVENT	548
TXTLOAD	536
TXTSTORE	537
UCASE\$	573
UNREGISTEREVENT	545
VAL	585
WRITE	516
ZPL	481

Alphabetical List of Set/Get/Do Commands



appl.bootblock	630	external_wired.check	719
appl.name	631	external_wired.ip.addr	720
appl.option_board_version	629	external_wired.ip.arp_interval	721
bluetooth.address	773	external_wired.ip.default_addr_enable	722
bluetooth.afh_map	774	external_wired.ip.dhcp.cid_all	723
bluetooth.afh_map_curr	775	external_wired.ip.dhcp.cid_enable	724
bluetooth.afh_mode	776	external_wired.ip.dhcp.cid_prefix	725
bluetooth.authentication	777	external_wired.ip.dhcp.cid_suffix	726
bluetooth.bluetooth_pin	778	external_wired.ip.dhcp.cid_type	727
bluetooth.date	779	external_wired.ip.gateway	728
bluetooth.discoverable	780	external_wired.ip.netmask	729
bluetooth.enable	781	external_wired.ip.port	730
bluetooth.friendly_name	782	external_wired.ip.protocol	731
bluetooth.local_name	783	external_wired.ip.timeout.enable	732
bluetooth.radio_auto_baud	784	external_wired.ip.timeout.value	733
bluetooth.radio_version	785	external_wired.ip.v6.addr	734
bluetooth.short_address	786	external_wired.ip.v6.gateway	735
bluetooth.version	787	external_wired.ip.v6.prefix_length	736
card.inserted	789	external_wired.mac_addr	737
card.mac_addr	788	external_wired.mac_raw	738
CISDFCRC16	632	file.delete	654
comm.type	790	file.dir	655
cutter.clean_cutter	635	file.run	657
device.download_connection_timeout	636	file.type	656
device.friendly_name	637	head.latch	658
device.frontpanel.key_press	638	interface.network.active.gateway	659
device.frontpanel.line1	639	interface.network.active.ip_addr	660
device.frontpanel.line2	640	interface.network.active.mac_addr	661
device.frontpanel.xml	641	interface.network.active.mac_raw	662
device.jobs_print	642	interface.network.active.netmask	663
device.languages	643	interface.network.active.protocol	664
device.orientation	644	internal_wired.auto_switchover	739
device.pnp_option	645	internal_wired.ip.addr	741
device.reset	646	internal_wired.ip.arp_interval	742
device.restore_defaults	647	internal_wired.ip.default_addr_enable	743
device.unique_id	648	internal_wired.ip.dhcp.cache_ip	744
device.uptime	649	internal_wired.ip.dhcp.cid_all	745
device.user_p1	650	internal_wired.ip.dhcp.cid_enable	746
device.user_p2	651	internal_wired.ip.dhcp.cid_prefix	747
device.xml.enable	652	internal_wired.ip.dhcp.cid_suffix	748
display.text	653	internal_wired.ip.dhcp.cid_type	749

internal_wired.ip.dhcp.lease.last_attempt	750	ip.mirror.success_time	835
internal_wired.ip.dhcp.lease.length	751	ip.mirror.username	836
internal_wired.ip.dhcp.lease.server	752	ip.mirror.version	837
internal_wired.ip.dhcp.lease.time_left	753	ip.netmask	838
internal_wired.ip.dhcp.option12	754	ip.pop3.enable	839
internal_wired.ip.dhcp.option12_format	755	ip.pop3.password	840
internal_wired.ip.dhcp.option12_value	756	ip.pop3.poll	841
internal_wired.ip.dhcp.requests_per_session	758	ip.pop3.server_addr	842
internal_wired.ip.dhcp.request_timeout	757	ip.pop3.username	843
internal_wired.ip.dhcp.session_interval	759	ip.port	844
internal_wired.ip.gateway	760	ip.primary_network	845
internal_wired.ip.netmask	761	ip.smtp.domain	846
internal_wired.ip.port	762	ip.smtp.enable	847
internal_wired.ip.protocol	763	ip.smtp.server_addr	848
internal_wired.ip.timeout.enable	764	ip.snmp.enable	851
internal_wired.ip.timeout.value	765	ip.snmp.get_community_name	849
internal_wired.mac_addr	766	ip.snmp.set_community_name	850
internal_wired.mac_raw	767	ip.tcp.enable	853
ip.active_network	791	ip.telnet.enable	852
ip.addr	792	ip.udp.enable	854
ip.arp_interval	793	media.cut_now	665
ip.bootp.enable	794	media.darkness_mode	666
ip.dhcp.cache_ip	795	media.media_low.external	667
ip.dhcp.cid_all	796	media.media_low.warning	668
ip.dhcp.cid_enable	797	media.present.cut_amount	669
ip.dhcp.cid_prefix	798	media.present.cut_margin	674
ip.dhcp.cid_suffix	799	media.present.eject	670
ip.dhcp.cid_type	801	media.present.length_addition	671
ip.dhcp.enable	802	media.present.loop_length	672
ip.dhcp.lease.last_attempt	803	media.present.loop_length_max	673
ip.dhcp.lease.length	800	media.present.present_timeout	675
ip.dhcp.lease.server	804	media.present.present_type	676
ip.dhcp.lease.time_left	805	media.printmode	677
ip.dhcp.option12	806	media.speed	679
ip.dhcp.option12_format	807	odometer.headclean	680
ip.dhcp.option12_value	808	odometer.headnew	681
ip.dhcp.requests_per_session	810	odometer.label_dot_length	682
ip.dhcp.request_timeout	809	odometer.media_marker_count1	683
ip.dhcp.session_interval	811	odometer.media_marker_count2	684
ip.dns.domain	812	odometer.retracts_count	685
ip.dns.servers	813	odometer.total_print_length	686
ip.ftp.enable	814	print.tone	687
ip.ftp.execute_file	815	rfid.error.response	688
ip.gateway	816	rfid.position.program	689
ip.http.enable	817	rfid.reader_1.antenna_port	690
ip.lpd.enable	818	rfid.reader_1.power.read	691
ip.mac_raw	819	rfid.reader_1.power.single_power	692
ip.mirror.auto	820	rfid.reader_1.power.write	693
ip.mirror.error_retry	821	wlan.adhocautomode	855
ip.mirror.feedback.auto	822	wlan.adhocchannel	856
ip.mirror.feedback.freq	823	wlan.associated	857
ip.mirror.feedback.odometer	824	wlan.bssid	858
ip.mirror.feedback.path	825	wlan.channel	859
ip.mirror.fetch	826	wlan.channel_mask	860
ip.mirror.freq	827	wlan.current_tx_rate	861
ip.mirror.freq_hours	828	wlan.essid	862
ip.mirror.last_error	829	wlan.firmware_version	863
ip.mirror.last_time	830	wlan.ip.addr	864
ip.mirror.password	831	wlan.ip.arp_interval	865
ip.mirror.path	832	wlan.ip.default_addr_enable	866
ip.mirror.server	833	wlan.ip.dhcp.cache_ip	867
ip.mirror.success	834	wlan.ip.dhcp.cid_all	868

wlan.ip.dhcp.cid_enable	869	zbi.control.variable_value	704
wlan.ip.dhcp.cid_prefix	870	zbi.key	705
wlan.ip.dhcp.cid_suffix	871	zbi.last_error	706
wlan.ip.dhcp.cid_type	872	zbi.reseller_key	707
wlan.ip.dhcp.lease.last_attempt	873	zbi.revision	708
wlan.ip.dhcp.lease.length	874	zbi.running_program_name	709
wlan.ip.dhcp.lease.server	875	zbi.start_info.execute	710
wlan.ip.dhcp.lease.time_left	876	zbi.start_info.file_name	711
wlan.ip.dhcp.option12	877	zbi.start_info.memory_alloc	712
wlan.ip.dhcp.option12_format	878	zbi.state	713
wlan.ip.dhcp.option12_value	879		
wlan.ip.dhcp.requests_per_session	881		
wlan.ip.dhcp.request_timeout	880		
wlan.ip.dhcp.session_interval	882		
wlan.ip.gateway	883		
wlan.ip.netmask	884		
wlan.ip.port	885		
wlan.ip.protocol	886		
wlan.ip.timeout.enable	887		
wlan.ip.timeout.value	888		
wlan.keep_alive.enable	889		
wlan.keep_alive.timeout	890		
wlan.kerberos.kdc	891		
wlan.kerberos.password	892		
wlan.kerberos.realm	893		
wlan.kerberos.username	894		
wlan.mac_addr	895		
wlan.mac_raw	896		
wlan.operating_mode	897		
wlan.password	898		
wlan.preamble	899		
wlan.private_key_password	900		
wlan.roam.interchannel_delay	901		
wlan.roam.interval	902		
wlan.roam.max_chan_scan_time	903		
wlan.roam.signal	904		
wlan.security	905		
wlan.signal_noise	914		
wlan.signal_quality	915		
wlan.signal_strength	916		
wlan.station_name	917		
wlan.tx_power	918		
wlan.tx_rate	919		
wlan.username	920		
wlan.wep.auth_type	921		
wlan.wep.index	922		
wlan.wep.key1	923		
wlan.wep.key2	924		
wlan.wep.key3	925		
wlan.wep.key4	926		
wlan.wep.key_format	928		
wlan.wpa.psk	927		
zbi.control.add_breakpoint	694		
zbi.control.break	695		
zbi.control.clear_breakpoints	696		
zbi.control.delete_breakpoint	697		
zbi.control.line_number	698		
zbi.control.restart	699		
zbi.control.run	700		
zbi.control.step	701		
zbi.control.terminate	702		
zbi.control.variable_name	703		



Notes • _____

About This Document



This section provides you with contact information, document structure and organization, and additional reference documents.

Contents

Who Should Use This Document	32
How This Document Is Organized	32
Contacts	33
Document Conventions	34

Who Should Use This Document

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

How This Document Is Organized

The Programming Guide is set up as follows:

Section	Description
<i>Introduction</i>	Provides a high-level overview about this guide and Zebra Programming Language (ZPL).
<i>ZPL Commands</i>	Provides an alphabetical, detailed description of each ZPL command.
<i>ZPL RFID Commands</i>	Provides an alphabetical, detailed description of each ZPL RFID command, and some examples of how to use them.
<i>ZPL Wireless Commands</i>	Provides new and modified ZPL commands for the wireless print server.
<i>ZBI Commands</i>	Provides details about the Zebra Basic Interpreter, its commands, descriptions, formats, and parameters.
<i>SGD Printer Commands</i>	Provides a high-level overview of printer setting Set / Get / Do (SGD) commands.
<i>SGD Wired Commands</i>	Provides a high-level overview of the wired Set / Get / Do (SGD) commands.
<i>SGD Wireless Commands</i>	Provides a high-level overview of the wireless Set / Get / Do (SGD) commands.
Appendixes	The appendixes include: <i>Zebra Code Pages</i> <i>ASCII</i> <i>Fonts and Bar Codes</i> <i>Mod 10 and Mod 43 Check Digits</i> <i>ZB64 Encoding and Compression</i> <i>Field Interactions</i> <i>Real Time Clock</i> <i>ZBI Character Set</i> <i>SGD Command Support</i> <i>Mirror</i> <i>Wireless Markup Language (WML)</i> <i>Features</i>
<i>Glossary</i>	Glossary of terms.

Contacts

Technical Support via the Internet is available 24 hours per day, 365 days per year.

Web Site: www.zebra.com

E-mail Back Technical Library:

E-mail address: emb@zebra.com

Subject line: Emaillist

Self Service Knowledge Base: www.zebra.com/knowledgebase

Online Case Registration: www.zebra.com/techrequest

Which Department Do You Need?	The Americas	Europe, Middle East, and Africa	Asia Pacific and India
Regional Headquarters	Zebra Technologies International, LLC 333 Corporate Woods Parkway Vernon Hills, IL 60061-3109 U.S.A. T: +1 847 793 2600 Toll-free +1 800 423 0422 F: +1 847 913 8766	Zebra Technologies Europe Limited Dukes Meadow Millboard Road Bourne End Buckinghamshire, SL8 5XF United Kingdom T: +44 (0) 1628 556000 F: +44 (0) 1628 556001	Zebra Technologies Asia Pacific Pte. Ltd. 120 Robinson Road #06-01 Parakou Building Singapore 068913 T: + 65 6858 0722 F: +65 6885 0838
Technical Support For questions on the operation of Zebra equipment and software, please call your distributor. For additional assistance, contact us. <i>Please have your model and serial numbers available.</i>	T: +1 877 ASK ZEBRA (275 9327) F: +1 847 913 2578 Hardware: ts1@zebra.com Software: ts3@zebra.com Kiosk printers: T: +1 866 322 5202 E: kiosksupport@zebra.com	T: +44 (0) 1628 556039 F: +44 (0) 1628 556003 E: Tseurope@zebra.com	T: +65 6858 0722 F: +65 6885 0838 E: China: tschina@zebra.com All other areas: tsasiapacific@zebra.com
Repair Service Department For back-to-base service and repair.	T: +1 877 ASK ZEBRA (275 9327) F: +1 847 821 1797 E: repair@zebra.com To request a repair in the U.S., go to www.zebra.com/repair .	T: +44 (0) 1772 693069 F: +44 (0) 1772 693046 New requests: ukrma@zebra.com Status updates: repairupdate@zebra.com	T: +65 6858 0722 F: +65 6885 0838 E: China: tschina@zebra.com All other areas: tsasiapacific@zebra.com
Technical Training Department For Zebra product training courses.	T: +1 847 793 6868 T: +1 847 793 6864 F: +1 847 913 2578 E: ttamerica@zebra.com	T: +44 (0) 1628 556000 F: +44 (0) 1628 556001 E: Eurtraining@zebra.com	T: + 65 6858 0722 F: +65 6885 0838 E: China: tschina@zebra.com All other areas: tsasiapacific@zebra.com
Inquiry Department For product literature and distributor and dealer information.	T: +1 877 ASK ZEBRA (275 9327) E: inquiry4@zebra.com	T: +44 (0) 1628 556037 F: +44 (0) 1628 556005 E: mseurope@zebra.com	E: China: GCmarketing@zebra.com All other areas: APACChannelmarketing@zebra.com
Customer Service Department (US) Internal Sales Department (UK) For printers, parts, media, and ribbon, please call your distributor or contact us.	T: +1 877 ASK ZEBRA (275 9327) E: clientcare@zebra.com	T: +44 (0) 1628 556032 F: +44 (0) 1628 556001 E: cseurope@zebra.com	T: +65 6858 0722 F: +65 6885 0836 E: China: order-csr@zebra.com All other areas: csasiapacific@zebra.com
Key: T: Telephone F: Facsimile E: E-mail			

Document Conventions

The following conventions are used throughout this document to convey certain information.

Alternate Color (online only) Cross-references contain hot links to other sections in this guide. If you are viewing this guide online in .pdf format, you can click the cross-reference ([blue text](#)) to jump directly to its location.

Command Line Examples Command line examples appear in `Courier New` font. For example, type `ZTOOLS` to get to the Post-Install scripts in the `bin` directory.

Files and Directories File names and directories appear in `Courier New` font. For example, the `Zebra<version number>.tar` file and the `/root` directory.

Icons Used



Identifies features that are available in printers with firmware version Vx.17.x or later.



Identifies features that are available in printers with firmware version V60.16, V53.16, or later.



Identifies features that are available in printers with firmware version V60.15.x, V50.15.x, or later.



Identifies features that are available in printers with firmware version V60.14, V50.14, or later.



Identifies features that are available in printers with firmware version V60.13.x, V50.13.x, or earlier.



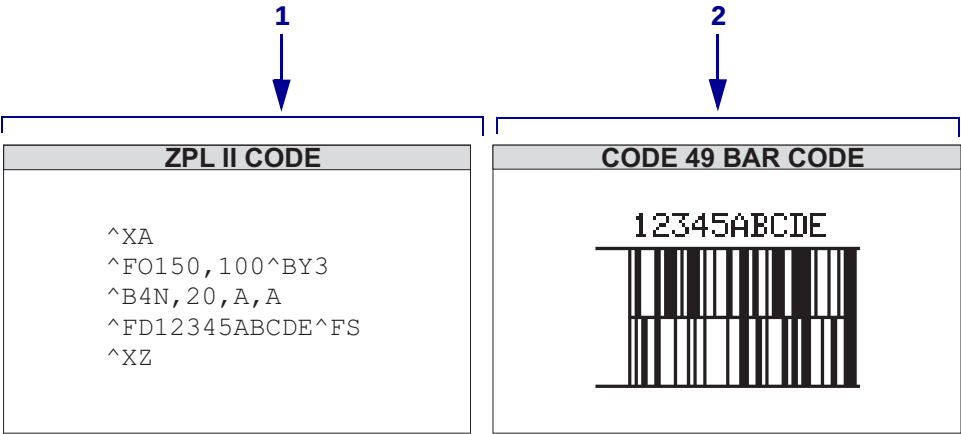
Important • Advises you of information that is essential to complete a task.



Note • Indicates neutral or positive information that emphasizes or supplements important points of the main text.

Illustration Callouts Callouts are used when an illustration contains information that needs to be labeled and described. A table that contains the labels and descriptions follows the graphic. [Figure 1](#) provides an example.

Figure 1 • Sample Figure with Callouts



1	ZPL Code
2	Generated Label



Notes • _____



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.

Firmware You can get the printer's firmware version by printing out a configuration label. For instructions to do so, see your printer's user guide.



Note • For firmware upgrades go to: www.zebra.com.



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.

If there are any terms used in this guide that you need clarification on, please see the [Glossary on page 1065](#).



Notes • _____



ZPL Commands

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

Description This heading provides an explanation of how the command is used, what it is capable of, and any defining characteristics it has.

Format 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: ^B80, h, f, g. It is arranged with the caret symbol (^), the command code (B8), and the parameters and are replaced with supported values.

Parameters If a command has values that can be defined to make its function more specific, these are outlined as parameters. Parameters typically have *Accepted Values* and *Default Values*.

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

h = bar code height (in dots)

Accepted Values: 1 to 32000

Default Value: value set by ^BY

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



Example • 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  
^F050,50  
^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.

Comment 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.



Example • An example comment is: **This command works only when the printer is idle, or This command is ignored if a value exceeds the parameter limits.**

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 (CANCEL test).
- ☐ 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.

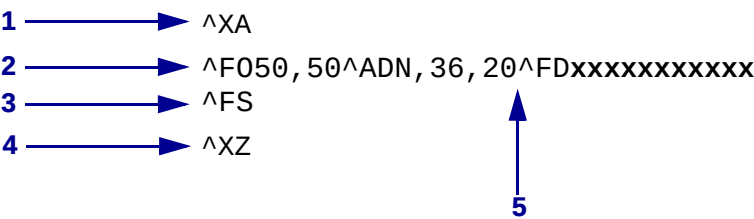
Exercise 1 • This exercise shows you how to specify a location for an entered name.

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



Important • Your name goes where you see **xxxxxxxxxxxx** in the second line of code.

3. Send this format to the printer:



1	Every format starts with the ^XA command
2	^F0 (field origin) command
3	^FS (field separator) command
4	Every format ends with the ^XZ command
5	^FD (field data) command

4. When the label prints correctly, alter the first number after the ^F0x . See how that change affects the print position. Alter the second number after the ^F050, 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.



Example • 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 Bar Codes on page 947](#).
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
^F050, 50^ADN, 36, 20^FDxxxxxxxxxxxx^FS
^F0 (fill in the rest)
^F0 (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

Special Effects commands

The Graphic Box or ^GB command or is used in some of the speical effects commands.

Reverse Printing a Field

1. Write the following format and send to the printer:

```
^XA
^PR1
^F0100,100
^GB70,70,70,,3^FS
^F0200,100
^GB70,70,70,,3^FS
^F0300,100
^GB70,70,70,,3^FS
^F0400,100
^GB70,70,70,,3^FS
^F0107,110^CF0,70,93
^FR^FDREVERSE^FS
^XZ
```

2. To see the effects, remove:

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

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

Label Reverse Print

1. Write the following format and send to the printer:

```
^XA^LRY
^F0100,50
^GB195,203,195^FS
^F0180,110^CFG
^FDLABEL^FS
^F0130,170
^FDREVERSE^FS
^XZ
```

2. To see the effects, remove:

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

Mirror Image of Label

1. Write the following format and send to the printer:
`^XA^PMY`
`^F0100,100`
`^CFG`
`^FDMIRROR^FS`
`^F0100,160`
`^FDIMAGE^FS`
`^XZ`
2. To see the effects, in the first line of code change `^PMY` to `^PMN`.

Print Orientation

1. Write the following format and send to the printer:
`^XA^CFD`
`^POI`
`^LH330,10`
`^F050,50`
`^FDZEBRA TECHNOLOGIES^FS`
`^F050,75`
`^FDVernon Hills, IL^FS`
`^XZ`
2. To see the effects, in the second line of code change `^POI` to `^PON`.

Exercise 2 • Boxes and lines

1. Use the address format from [Exercise 1](#).
2. Add this new line to your existing format:
`^F050,200^GB200,200,2^FS`
 This prints a box one wide by one inch long and the thickness of the line is 2 dots.
3. Reposition and resize the square so that it goes around the name and address uniformly.
4. Print a line by adding:
`^F050,300^GB400,0,4,^FS`
 This prints a horizontal line two inches wide by 4 dots thick.
5. Print a vertical line using this code:
`^F0100,50^GB0,400,4^FS`

Exercise 3 • Bar codes — ^B3 code 39 bar code

1. Write the following format and send to the printer:
`^XA`
`^F050,50^B3N,N,100,Y,N^FD123456^FS`
`^XZ`
2. Try changing each of the parameters in the ^B3 string so you can see the effects.
`^B3o,e,h,f,g`
`^BY`



Important • For valid parameter choices, see [^B3 on page 61](#).

3. Insert the ^BY command just before the ^B3 to see how the narrow bar width can be altered.
`^F050,50^BY2^B3..etc ^BYx`, acceptable values for x are 1 through 10
4. Alter the ratio of the narrow to wide bar.
`^F050,50^BY2,3^B3..etc ^BY2,x` acceptable values for x are 2.1 through 3 in .1 increments
5. Print out a ^B3 bar code with the interpretation line on top of the bar code and the bar code rotated 90 degrees.
6. Add a ^PQ just before the ^XZ to print several labels.
`^PQ4`
`^XZ`
`^PR` Print rate (in inches per second)
7. 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

1. Send this format to the printer:

```
^XA
^F0100,100^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

2. 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
^F0100,100^ADN,36,20^FDABCD1000EFGH^SF%%d%%,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 on page 335](#).

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



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

1. Send this format to the printer:

```
^XA
^F020,30^GB750,1100,4^FS
^F020,30^GB750,200,4^FS
^F020,30^GB750,400,4^FS
^F020,30^GB750,700,4^FS
^F020,226^GB325,204,4^FS
^F030,40^ADN,36,20^FDShip to:^FS
^F030,260^ADN,18,10^FDPart number #^FS
^F0360,260^ADN,18,10^FDDescription:^FS
^F030,750^ADN,36,20^FDFrom:^FS
^ISR:SAMPLE.GRF^FS
^XZ
```

2. Send this format:

```
^XA
^ILR:SAMPLE.GRF^FS
^F0150,125^ADN,36,20^FDAcme Printing^FS
^F060,330^ADN,36,20^FD14042^FS
^F0400,330^ADN,36,20^FDScrew^FS
^F070,480^BY4^B3N,,200^FD12345678^FS
^F0150,800^ADN,36,20^FDMacks Fabricating^FS
^XZ
```

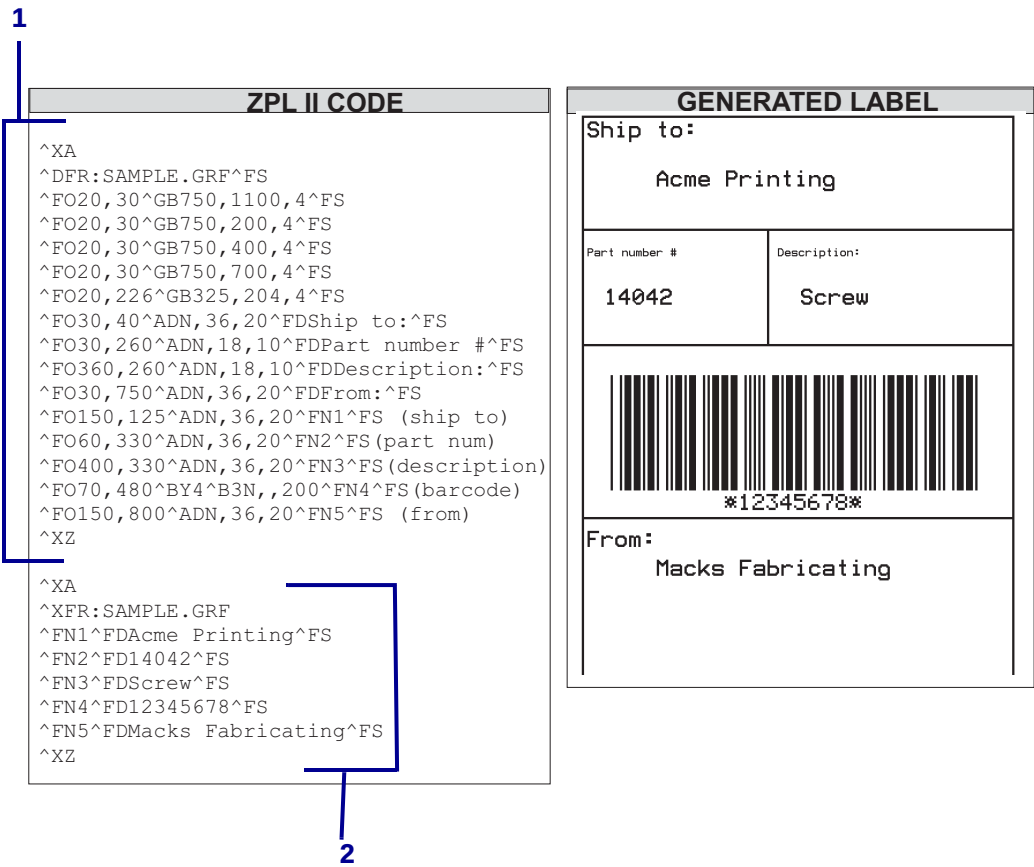
In this way the template only needs to be sent one time to the printer's memory. Subsequent formats can be sent recalling the template and merging variable data into the template. In this exercise, the file was saved in the printers R: memory, which is volatile.

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 2 • Download and Recall Format



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

Exercise 7 • Asian and Unicode Encodings



This exercise works in 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
<pre>^XA^CW1,B:ANMDK.TTF ^SEB:UHANGUL.DAT^CI26 ^FO100,100^A1,50,50^FDASCII 한국어 ^FS ^XZ</pre>	ASCII한국어



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

ZPL II CODE
<pre>^XA^CW1,ANMDJ.TTF^CI28 ^FO100,50^A1,30,30^FDENGLISH/日本語/한국어/简体中文/繁体中文^FS ^XZ</pre>
GENERATED LABEL
ENGLISH/日本語/한국어/简体中文/繁体中文

^A

Scalable/Bitmapped Font

Description 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.

Format ^Afo,h,w



Important • Parameter **f** is required. If **f** is omitted it defaults to the last value of the ^CF command.

This table identifies the parameters for this format:

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

Scalable Font Command

➔ **Example** • This is an example of a scalable font command:

ZPL II CODE	GENERATED LABEL
<pre>^XA ^FO50,50 ^A0,32,25 ^FDZEBRA^FS ^FO50,150 ^A0,32,25 ^FDPROGRAMMING^FS ^FO50,250 ^A0,32,25^FDLANGUAGE^FS ^XZ</pre>	ZEBRA PROGRAMMING LANGUAGE

Bitmap Font Command

➔ **Example** • This is an example of a bitmap font command:

ZPL II CODE	GENERATED LABEL
<pre>^XA ^FO50,50 ^ADN,36,20 ^FDZEBRA^FS ^FO50,150 ^ADN,36,20 ^FDPROGRAMMING^FS ^FO50,250 ^ADN,36,20^FDLANGUAGE^FS ^XZ</pre>	ZEBRA PROGRAMMING LANGUAGE

For reference, see [Standard Printer Fonts on page 948](#), [Zebra Code Page 850 — Latin Character Set on page 930](#), [Fonts and Bar Codes on page 947](#), and [ASCII on page 945](#).

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.



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 971](#).



Use Font Name to Call Font

Description 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@.

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

This table identifies the parameters for this format:

Parameters	Details
o = field orientation	<i>Accepted Values:</i> - N = normal - R = rotates 90 degrees (clockwise) - I = inverted 180 degrees - B = read from bottom up, 270 degrees <i>Default Value:</i> N or the last ^FW value
h = character height (in dots)	<i>Default Value:</i> 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)	<i>Default Value:</i> 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	<i>Accepted Values:</i> R:, E:, B:, and A: <i>Default Value:</i> R:
o = font name	<i>Accepted Values:</i> any valid font <i>Default Value:</i> 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.

Parameters	Details
x = extension .14↑ .TTE is only supported in firmware version V60.14.x, V50.14.x, or later.	Accepted Values: .FNT = font .TTF = TrueType Font .TTE = TrueType Extension

➔ **Example** • This example is followed by a table that identifies the called out lines of code:

1

2

3

4

5

6

7

8

ZPL II CODE

^XA

^A@N,50,50,B:CYRI_UB.FNT

^FO100,100

^FDZebra Printer Fonts^FS

^A@N,40,40

^FO100,150

^FDThis uses B:CYRI_UB.FNT^FS

^XZ

GENERATED LABEL

Zebra Printer Fonts

This uses B:CYRI_UB.FNT

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, <i>Zebra Printer Fonts</i> 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, <i>This uses the B:CYRI_UB.FNT</i> on the label.
8	Ends the label format.

For reference, see [Zebra Code Page 850 — Latin Character Set](#) on page 930, [Fonts and Bar Codes](#) on page 947, and [ASCII](#) on page 945.

^B0

Aztec Bar Code Parameters

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



Note • The Aztec bar code works with firmware version V60.13.0.11A and V50.13.2 or later.

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

This table identifies the parameters for this format:

Parameters	Details
a = orientation	<i>Accepted Values:</i> N = normal R = rotated I = inverted 180 degrees B = read from bottom up, 270 degrees <i>Default Value:</i> current ^FW value
b = magnification factor	<i>Accepted Values:</i> 1 to 10 <i>Default Value:</i> 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	<i>Accepted Values:</i> Y = if data contains ECICs N = if data does not contain ECICs <i>Default Value:</i> N
d = error control and symbol size/type indicator	<i>Accepted Values:</i> 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” <i>Default Value:</i> 0
e = menu symbol indicator	<i>Accepted Values:</i> Y = if this symbol is to be a menu (bar code reader initialization) symbol N = if it is not a menu symbol <i>Default Value:</i> N

Parameters	Details
f = number of symbols for structured append	<i>Accepted Values:</i> 1 through 26 <i>Default Value:</i> 1
g = optional ID field for structured append	The ID field is a text string with 24-character maximum <i>Default Value:</i> no ID



Example • This is an example of the ^B0 command:

ZPL II CODE	GENERATED LABEL
<pre> ^XA ^B0R,7,N,0,N,1,0 ^FD 7. This is testing label 7^FS ^XZ </pre>	

^B1

Code 11 Bar Code

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

- ^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.

Format ^B1o,e,h,f,g



Important • If additional information about the Code 11 bar code is required, go to www.aimglobal.org.

This table identifies the parameters for this format:

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

➔ **Example •** This is an example of the Code 11 bar code:

ZPL II CODE	CODE 11 BAR CODE
<pre>^XA ^FO100,100^BY3 ^B1N,N,150,Y,N ^FD123456^FS ^XZ</pre>	

CODE 11 BAR CODE CHARACTERS											
0	1	2	3	4	5	6	7	8	9	-	
Internal Start/Stop Character: △											
<i>When used as a stop character:</i>											
△ is used with 1 check digit											
△ is used with 2 check digits											

^B2

Interleaved 2 of 5 Bar Code

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

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.

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



Important • If additional information about the Interleaved 2 of 5 bar code is required, go to www.aimglobal.org.

This table identifies the parameters for this format:

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



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

ZPL II CODE

^XA

^FO100,100^BY3

^B2N,150,Y,N,N

^FD123456^FS

^XZ

INTERLEAVED 2 OF 5 BAR CODE



123456

INTERLEAVED 2 OF 5 BAR CODE CHARACTERS

0123456789

Start/Stop (internal)

Comments 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 962](#).

^B3

Code 39 Bar Code

Description The Code 39 bar code is the standard for many industries, including the U.S. 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.

Each character in a Code 39 bar code 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.

Format ^B3o, e, h, f, g



Important • If additional information about the Code 39 bar code is required, go to www.aimglobal.org.

This table identifies the parameters for this format:

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

Parameters	Details
g = print interpretation line above code	<i>Accepted Values:</i> Y = yes N = no <i>Default Value:</i> N



Example • This is an example of a Code 39 bar code:

ZPL II CODE	CODE 39 BAR CODE																																																																
^XA ^FO100,100^BY3 ^B3N,N,100,Y,N ^FD123ABC^FS ^XZ	 *123ABC*																																																																
CODE 39 BAR CODE CHARACTERS																																																																	
<table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td></td></tr><tr><td>A</td><td>B</td><td>C</td><td>D</td><td>E</td><td>F</td><td>G</td><td>H</td><td>I</td><td>J</td><td>K</td><td>L</td><td>M</td><td>N</td><td>O</td><td>P</td><td>Q</td><td>R</td><td>S</td><td>T</td><td>U</td><td>V</td><td>W</td><td>X</td><td>Y</td><td>Z</td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>-</td><td>.</td><td>\$</td><td>/</td><td>+</td><td>%</td><td>Space</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>			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												
	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 bar code. 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 bar code:

ZPL II CODE	GENERATED LABELS
^XA ^FO20,20 ^B3N,N,100,Y ^FDTEST+\$M\$J-\$^FS ^XZ	 *TEST+\$M\$J-\$*

Full ASCII Mode for Code 39

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

Table 1 • Code 39 Full ASCII Mode

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

Table 2 • Code 39 Full ASCII Mode

ASCII	Code 39	ASCII	Code 39
@	%V	'	%W
A	A	a	+A
B	B	b	+B
C	C	c	+C
D	D	d	+D
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, %X

^B4

Code 49 Bar Code

Description 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.

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.

Format ^B4o,h,f,m



Important • For additional information about the Code 49 bar code is required, go to www.aimglobal.org.

This table identifies the parameters for this format:

Parameters	Details
o = orientation	<i>Accepted Values:</i> - N = normal - R = rotated 90 degrees (clockwise) - I = inverted 180 degrees - B = read from bottom up, 270 degrees <i>Default Value:</i> current ^FW value
h = height multiplier of individual rows	<i>Accepted Values:</i> 1 to height of label <i>Default Value:</i> 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	<i>Accepted Values:</i> - N = no line printed - A = print interpretation line above code - B = print interpretation line below code <i>Default Value:</i> N When the field data exceeds two rows, expect the interpretation line to extend beyond the right edge of the bar code symbol.

Parameters	Details
m = starting mode	<i>Accepted Values:</i> 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. <i>Default Value: A</i>



Example • This is an example of a Code 49 bar code:

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

Table 3 • Code 49 CharacterSet

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.



Example • 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

Planet Code bar code

Description The ^B5 command is supported in all printers as a resident bar code.



Note • Accepted bar code characters are 0 - 9.

Format ^B5o,h,f,g

This table identifies the parameters for this format:

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



Example • This is an example of a Planet Code bar code:

ZPL II CODE	GENERATED LABEL
<pre>^XA ^FO150,100^BY3 ^B5N,100,Y,N ^FD12345678901^FS ^XZ</pre>	

^B7

PDF417 Bar Code

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

The bar code 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 bar code is also capable of using the structured append option (^FM), which allows you to extend the field data limitations by printing multiple bar codes. For more information on using structured append, see [^FM on page 190](#).

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

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

This table identifies the parameters for this format:

Parameters	Details
o = orientation	<i>Accepted Values:</i> N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from bottom up, 270 degrees <i>Default Value:</i> current ^FW value
h = bar code height for individual rows (in dots)	<i>Accepted Values:</i> 1 to height of label <i>Default Value:</i> 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 bar code height, divided by the number of rows, equals the height of the individual rows in dots, where the overall bar code height is defined by the ^BY command. 1 is not a recommended value.
s = security level	<i>Accepted Values:</i> 1 to 8 (error detection and correction) <i>Default Value:</i> 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.

Parameters	Details
c = number of data columns to encode	<i>Accepted Values:</i> 1 to 30 <i>Default Value:</i> 1 : 2 (row-to-column aspect ratio) You can specify the number of code-word columns giving control over the width of the symbol.
r = number of rows to encode	<i>Accepted Values:</i> 3 to 90 <i>Default Value:</i> 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	<i>Accepted Values:</i> N = no truncation Y = perform truncation <i>Default Value:</i> N

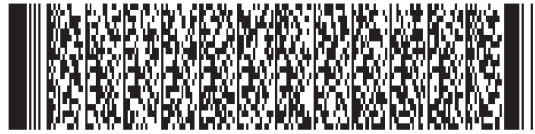


Example • This is an example of a PDF417 bar code:

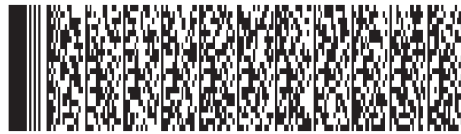
ZPL II CODE	PDF417 BAR CODE
<pre>^XA ^BY2,3 ^FO10,10^B7N,5,5,,83,N ^FDZebra Technologies Corporation strives to be the expert supplier of innovative solutions to speciality demand labeling and ticketing problems of business and government. We will attract and retain the best people who will understand 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</pre>	



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



PDF417 without Truncation being selected



PDF417 with Truncation being selected



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

ZPL II CODE	GENERATED LABEL
<pre>^XA ^FO50,50^BY3,3.0^B7N,8,5,7,21,N ^FH_ ^FD[]>_1E06_1DP12345678_1DQ160 _1D1JUN123456789A2B4C6D8E_1D20LA6-987 _1D21L54321_ZES_1D15KG1155 _1DBSC151208_1D7Q10GT_1E_04^FS ^XZ</pre>	

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 bar code.
- The truncation feature can be used in situations where label damage is not likely. The right row indicators and stop pattern is reduced to a single module bar width. The difference between a non truncated and a truncated bar code 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)

Accepted Values: 2 to 10

Default Value: 2

r = ratio

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

h = height of bars (in dots)

Accepted Values: 1 to 32000

Default Value: 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 930](#), [^CC ~CC on page 146](#), and [^CT ~CT on page 158](#).

- 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

EAN-8 Bar Code

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

- ^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.

Format ^B8o, h, f, g



Important • If additional information about the EAN-8 bar code is required, go to www.aimglobal.org.

This table identifies the parameters for this format:

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



Example • This is an example of an EAN-8 bar code:

ZPL II CODE	EAN-8 BAR CODE								
^XA ^FO100,100^BY3 ^B8N,100,Y,N ^FD1234567^FS ^XZ	 1234 5670								
EAN-8 BAR CODE CHARACTERS									
0	1	2	3	4	5	6	7	8	9

^B9

UPC-E Bar Code

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

Format ^B9, h, f, g, e



Important • If additional information about the UPC-E bar code is required, go to www.aimglobal.org.

This table identifies the parameters for this format:

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

Parameters	Details
e = print check digit	<i>Accepted Values:</i> N = no Y = yes <i>Default Value:</i> Y



Example • This is an example of a UPC-E bar code:

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

Code 93 Bar Code

Description The ^BA command creates a variable length, continuous symbology. The Code 93 bar code 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.

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

Each character in the Code 93 bar code 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.

Format ^BAo, h, f, g, e



Important • If additional information about the Code 93 bar code is required, go to www.aimglobal.org.

This table identifies the parameters for this format:

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

Parameters	Details
g = print interpretation line above code	<i>Accepted Values:</i> N = no Y = yes <i>Default Value:</i> N
e = print check digit	<i>Accepted Values:</i> N = no Y = yes <i>Default Value:</i> N



Example • This is an example of a Code 93 bar code:

ZPL II CODE	CODE 93 BAR CODE																																																												
<pre>^XA ^FO100,75^BY3 ^BAN,100,Y,N,N ^FD12345ABCDE^FS ^XZ</pre>	 012345ABCDE0																																																												
CODE 93 BAR CODE CHARACTERS																																																													
<table><tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td></td></tr><tr><td>A</td><td>B</td><td>C</td><td>D</td><td>E</td><td>F</td><td>G</td><td>H</td><td>I</td><td>J</td><td>K</td><td>L</td></tr><tr><td>M</td><td>N</td><td>O</td><td>P</td><td>Q</td><td>R</td><td>S</td><td>T</td><td>U</td><td>V</td><td>W</td><td>X</td></tr><tr><td>Y</td><td>Z</td><td>-</td><td>.</td><td>\$</td><td>/</td><td>+</td><td>%</td><td>&</td><td>'</td><td>(</td><td>)</td></tr><tr><td colspan="12">SPACE</td></tr></table>			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											
	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. For more details, see [Table 4 on page 80](#).

Full ASCII Mode for Code 93

Code 93 can generate the full 128-character ASCII set using paired characters as shown in [Table 4 on page 80](#).

Table 4 • Code 93 Full ASCII Mode

ASCII	Code 93	ASCII	Code 93
NUL	'U	SP	Space
SOH	&A	!	(A
STX	&B	"	(B
ETX	&C	#	(C
EOT	&D	\$	(D
ENQ	&E	%	(E
ACK	&F	&	(F
BEL	&G	'	(G
BS	&H	(	(H
HT	&I	)	(I
LF	&J	*	(J
VT	&K	++	++
FF	&L	,	(L
CR	&M	-	-
SO	&N	.	.
SI	&O	/	/
DLE	&P	0	O
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

Table 5 • Code 93 Full ASCII Mode

ASCII	Code 93	ASCII	Code 93
@	'V	'	'W
A	A	a	)A
B	B	b	)B
C	C	c	)C
D	D	d	)D
E	E	e	)E
F	F	f	)F
G	G	g	)G
H	H	h	)H
I	I	l	)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

^BB

CODABLOCK Bar Code

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

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.

Format ^BB*o, h, s, c, r, m*



Important • If additional information about the CODABLOCK bar code is required, go to www.aimglobal.org.

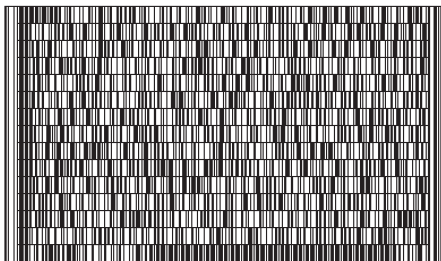
This table identifies the parameters for this format:

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

Parameters	Details
r = number of rows to encode	<i>Accepted Values:</i> 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 both parameters are specified, the amount of field data must be less than the product of the specified parameters. If the field data exceeds the value of the product, either no symbol or an error code is printed (if ^CV is active). • 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	<i>Accepted Values:</i> 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. <i>Default Value:</i> F



Example • This is an example of a CODABLOCK bar code:

ZPL II CODE	CODABLOCK BAR CODE
<pre> ^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 </pre>	

Special Considerations for the ^BY Command When Using ^BB

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

w = module width (in dots)

Accepted Values: 2 to 10 (CODABLOCK A only)

Default Value: 2

r = ratio

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

h = height of bars (in dots)

Accepted Values: 1 to 32, 32000

Default Value: 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 bar code 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 <nu l> character can be inserted using of the ^FH command.

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

For any other character above 84 hex, either no bar code 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 <nu l> character can be inserted using of the ^FH command.

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

^BC

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

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

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.

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



Important • If additional information about the Code 128 bar code is required, go to www.aimglobal.org.

This table identifies the parameters for this format:

Parameters	Details
o = orientation	<i>Accepted Values:</i> N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from bottom up, 270 degrees <i>Default Value:</i> current ^FW value
h = bar code height (in dots)	<i>Accepted Values:</i> 1 to 32000 <i>Default Value:</i> value set by ^BY
f = print interpretation line	<i>Accepted Values:</i> Y (yes) or N (no) <i>Default Value:</i> Y The interpretation line can be printed in any font by placing the font command before the bar code command.
g = print interpretation line above code	<i>Accepted Values:</i> Y (yes) or N (no) <i>Default Value:</i> N
e = UCC check digit	<i>Accepted Values:</i> 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. <i>Default Value:</i> N

Parameters	Details
m = mode	<i>Accepted Values:</i> - 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 bar code. 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, calculate it, and print it. Automatically sizes the human readable. <i>Default Value:</i> N



Example • This is an example of a Code 128 bar code:

ZPL II CODE	CODE 128 BAR CODE
<pre>^XA ^FO100,100^BY3 ^BCN,100,Y,N,N ^FD123456^FS ^XZ</pre>	 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 bar code.
- The desired Start Code can be placed at the beginning of the field data. If no Start Code is entered, Subset B are used.

To change subsets within a bar code, 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.

Table 6 shows the Code 128 Invocation Codes and Start Characters for the three subsets.

Table 6 • 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 Code A	(Numeric Pairs give Alpha/Numerics)	
>:	104	Start Code B	(Normal Alpha/Numeric)	
>;	105	Start Code C	(All numeric (00 - 99))	

Table 7 shows the character sets for Code 128:

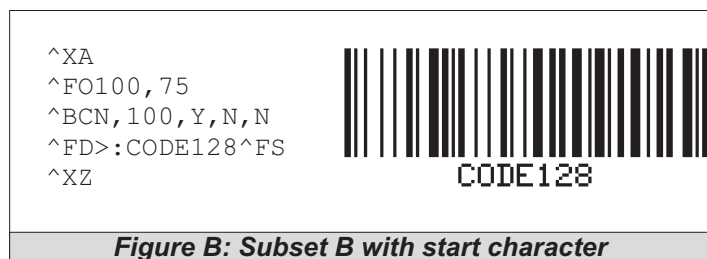
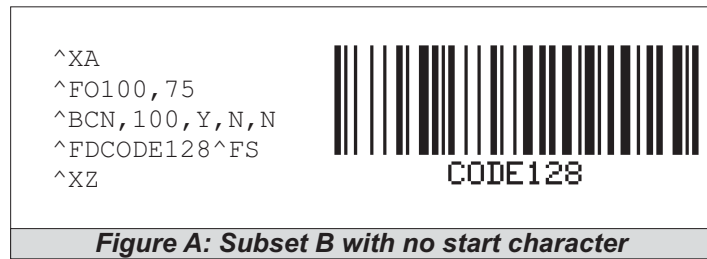
Table 7 • Code 128 Character Sets

Value	Code A	Code B	Code C
0	SP	SP	00
1	!	!	01
2	"	"	02
3	#	#	03
4	\$	\$	04
5	%	%	05
6	&	&	06
7	'	'	07
8	(	(	08
9	)	)	09
10	*	*	10
11	+	+	11
12	,	,	12
13	-	-	13
14	.	.	14
15	/	/	15
16	0	0	16
17	1	1	17
18	2	2	18
19	3	3	19
20	4	4	20
21	5	5	21
22	6	6	22
23	7	7	23
24	8	8	24
25	9	9	25
26	:	:	26
27	;	;	27
28	<	<	28
29	=	=	29
30	>	>	30
31	?	?	31
32	@	@	32
33	A	A	33
34	B	B	34
35	C	C	35
36	D	D	36
37	E	E	37
38	F	F	38
39	G	G	39
40	H	H	40
41	I	I	41
42	J	J	42
43	K	K	43
44	L	L	44
45	M	M	45
46	N	N	46
47	O	O	47
48	P	P	48
49	Q	Q	49
50	R	R	50
51	S	S	51
52	T	T	52

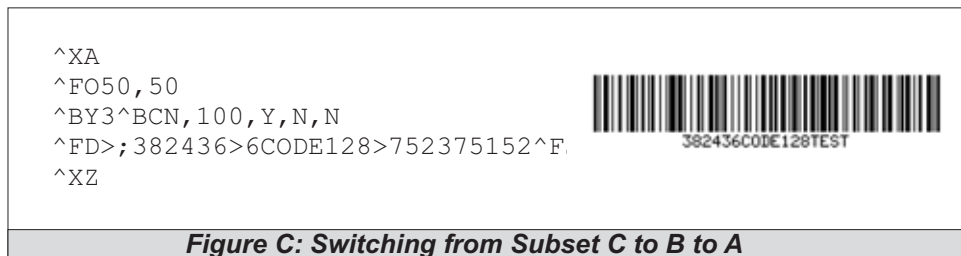
Value	Code A	Code B	Code C
53	U	U	53
54	V	V	54
55	W	W	55
56	X	X	56
57	Y	Y	57
58	Z	Z	58
59	[	[	59
60	\	\	60
61	]	]	61
62	^	^	62
63	~	~	63
64	NUL	.	64
65	SOH	a	65
66	STX	b	66
67	ETX	c	67
68	EOT	d	68
69	ENQ	e	69
70	ACK	f	70
71	BEL	g	71
72	BS	h	72
73	HT	i	73
74	LF	j	74
75	VT	k	75
76	FF	l	76
77	CR	m	77
78	SO	n	78
79	SI	o	79
80	DLE	p	80
81	DC1	q	81
82	DC2	r	82
83	DC3	s	83
84	DC4	t	84
85	NAK	u	85
86	SYN	v	86
87	ETB	w	87
88	CAN	x	88
89	EM	y	89
90	SUB	z	90
91	ESC	{	91
92	FS		92
93	GS	}	93
94	RS	~	94
95	US	DEL	95
96	FNC3	FNC3	96
97	FNC2	FNC2	97
98	SHIFT	SHIFT	98
99	Code C	Code C	99
100	Code B	FNC4	Code B
101	FNC4	Code A	Code A
102	FNC1	FNC1	FNC1
103		START (Code A)	
104		START (Code B)	
105		START (Code C)	



Example • Figures A and B are examples of identical bar codes, and Figure C is an example of switching from Subset C to B to A, as follows:



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.



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 bar code to be printed with no rotation (N) and a height of 100 dots. An interpretation line is printed (Y) below the bar code (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 bar code.

^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:

^ > ~



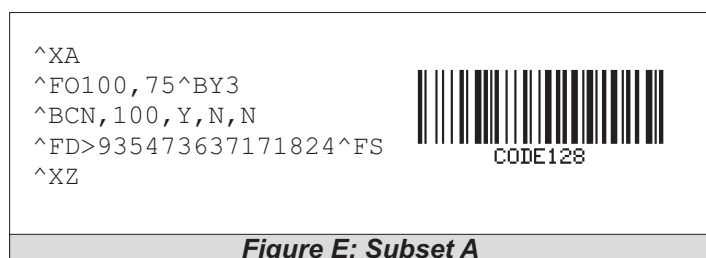
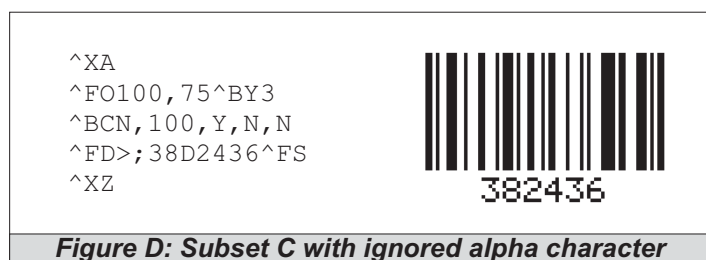
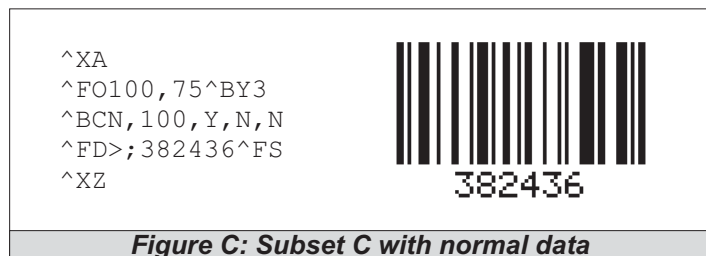
Example • Code 128 – Subsets A and C

Code 128, Subsets A and C are programmed in pairs of digits, 00 to 99, in the field data string. For details, see [Table 6 on page 88](#).

In Subset A, each pair of digits results in a single character being encoded in the bar code; 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.

Figure C and Figure D below are examples of Subset C. Notice that the bar codes are identical. In the program code for Figure D, the *D* is ignored and the 2 is paired with the 4.



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 to 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 sales 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 bar code 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 bar code. The code is also very reliable. Code 128 symbols use two independent self-checking features which improves printing and scanning reliability.

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

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

UCC/EAN, 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 8, UCC Application Identifier Table on page 95](#). It could be 00 which is the SSCC code. The customer needs to let us know what application identifiers are used for their bar code so we can help them.

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

Using N for the mode (m) parameter

➔ **Example** • This example shows with application identifier 00 structure:

ZPL II CODE	N FOR THE M PARAMETER
<pre>^XA ^FO90,200^BY4 ^BCN,256,Y,N,Y,N ^FD>;>80012345123451234512^FS ^XZ</pre>	

- >;>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 mode set to N

Using U for the mode (m) parameter

➔ **Example** • The example shows the application identifier 00 format:

ZPL II CODE	U FOR THE M PARAMETER
<pre>^XA ^FO90,200 ^BY4^BC,256,Y,N,,U ^FD0012345123451234512^FS ^XZ</pre>	

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

➔ **Example** • This example shows application identifier 00 format ((x.11.x or later):

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

(0 at end of field data is a bogus character that is inserted as a place holder 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 bogus check digit place holder.
- Check digit is automatically inserted. The printer will automatically calculate the check digit and put it into the bar code and interpretation line.
- The interpretation line will also show the parentheses and spaces but will strip them out from the actual bar code.

Printing the Interpretation Line

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

ZPL II CODE	INTERPRETATION LINE
<pre>^XA ^FO50,200 ^A0N,40,30^BCN,150,Y,N,Y ^FD>;>80012345123451234512^FS ^XZ</pre>	

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.

ZPL II CODE	FIRMWARE OLDER THAN X.10.X
<pre> ^XA ^FO25,25 ^BCN,150,N,N,Y ^FD>;>80012345123451234512^FS ^FO100,190 ^A0N,50,40 ^FD(00) 1 2345123 451234512 0^FS ^XZ </pre>	

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.

Table 8 • UCC Application Identifier Table

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 alpha numerics
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 alpha numerics
HIBCC Quantity, Date, Batch and Link	22	up to 29 alpha numerics
Lot Number	23 ^a	up to 19 alpha numerics

a. Plus one digit for length indication.

b. Plus one digit for decimal point indication.

Table 8 • UCC Application Identifier Table (Continued)

Data Content	AI	Plus The Following Data Structure
Quantity Each	30	
Net Weight (Kilograms)	310 ^b	exactly 6 digits
Length, Meters	311 ^b	exactly 6 digits
Width or Diameter (Meters)	312 ^b	exactly 6 digits
Depths (Meters)	313 ^b	exactly 6 digits
Area (Sq. Meters)	314 ^b	exactly 6 digits
Volume (Liters)	315 ^b	exactly 6 digits
Volume (Cubic Meters)	316 ^b	exactly 6 digits
Net Weight (Pounds)	320 ^b	exactly 6 digits
Customer PO Number	400	up to 29 alpha numerics
Ship To (Deliver To) Location Code using EAN 13 or DUNS Number with leading zeros	410	exactly 13 digits
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 alpha numerics
Ship To (Deliver To) Postal Code with 3-digit ISO Country Code Prefix	421	3 digits plus up to 9 alpha numerics
Roll Products - width, length, core diameter, direction and splices	8001	exactly 14 digits
Electronic Serial number for cellular mobile phone	8002	up to 20 alpha numerics

a. Plus one digit for length indication.

b. Plus one digit for decimal point indication.



Note • Table 8 is a partial table showing the application identifiers. For more current and complete information, search the Internet for **UCC Application Identifier**.

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 AI's so that the scanners reading the code sees the FNC1 and knows that an AI follows.

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

```
^XA
^BY2,2.5,193
^F033,400
^BCN,,N,N,N,A
^FD>;>80204017773003486100008535>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
^F033,200
^BCN,,N,N,N,U
^FD>;>80204017773003486>8100008535>8910001>837252^FS
^FT33,455^A0N,30,30^FD(02)04017773003486(10)0008535(9
1)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
^F0218,343
^BCB,,Y,N,N,D
^FD(91)0005886>8(10)0000410549>8(99)05^FS
^XZ
```

D* — When trying to print the last Application Identifier with an odd number of characters, a problem existed when printing EAN128 bar codes using Mode D. The problem was fixed in firmware version V60.13.0.6.

^BD

UPS MaxiCode Bar Code

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

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 bar code. However, the ^CV command can be activated.

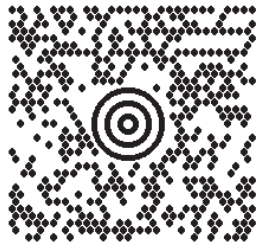
Format ^BDm, n, t

This table identifies the parameters for this format:

Parameters	Details
m = mode	<i>Accepted Values:</i> 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 <i>Default Value:</i> 2
n = symbol number	<i>Accepted Values:</i> 1 to 8 can be added in a structured document <i>Default Value:</i> 1
t = total number of symbols	<i>Accepted Values:</i> 1 to 8, representing the total number of symbols in this sequence <i>Default Value:</i> 1



Example • This is an example of UPS MAXICODE - MODE 2 bar code:

ZPL II CODE	UPS MAXICODE - MODE 2
<pre>^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^FMode2^FS ^XZ</pre>	

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 U.S. 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>

This table identifies the parameters for this format:

Parameters	Details
<hpm> = high priority message (applicable only in Modes 2 and 3)	<i>Accepted Values: 0 to 9, except where noted</i> U.S. Style Postal Code (Mode 2) <hpm> = aaabbbcccccdddd 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 cccccc = 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 **<lpn>** 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 **<lpn>** 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

EAN-13 Bar Code

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

The EAN-13 bar code 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.

Format ^BEo, h, f, g



Note • Use Interleaved 2 of 5 for UCC and EAN 14.



Important • If additional information about the EAN-13 bar code is required, go to www.aimglobal.org.

This table identifies the parameters for this format:

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

➔ **Example •** This is an example of an EAN-13 bar code:

ZPL II CODE	EAN-13 BAR CODE
<pre>^XA ^FO100,100^BY3 ^BEN,100,Y,N ^FD12345678^FS ^XZ</pre>	 0 000123 456784
EAN-13 BAR CODE CHARACTERS	
0123456789	

Comments The EAN-13 bar code uses the Mod 10 check-digit scheme for error checking. For more information on Mod 10, see [Mod 10 Check Digit on page 962](#).

^BF

MicroPDF417 Bar Code

Description 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 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

This table identifies the parameters for this format:

Parameters	Details
o = orientation	<i>Accepted Values:</i> N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from bottom up, 270 degrees <i>Default Value:</i> current ^FW value
h = bar code height (in dots)	<i>Accepted Values:</i> 1 to 9999 <i>Default Value:</i> value set by ^BY or 10 (if no ^BY value exists).
m = mode	<i>Accepted Values:</i> 0 to 33 (see Table 9, MicroPDF417 Mode on page 105) <i>Default Value:</i> 0 (see Table 9)



Example • This is an example of a MicroPDF417 bar code:

ZPL II CODE	MICRO-PDF417 BAR CODE
<pre> ^XA ^FO100,100^BY6 ^BFN,8,3 ^FDABCDEF GHIJKLMNOPQRSTU V^FS ^XZ </pre>	

To encode data into a MicroPDF417 bar code, 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 bar code (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 bar code. The higher the percentage of check digits that are used, the more resistant the bar code is to damage — however, the size of the bar code increases.
4. Use [Table 9, MicroPDF417 Mode on page 105](#) with the information gathered from the questions above to select the mode of the bar code.

Table 9 • 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
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

Industrial 2 of 5 Bar Codes

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

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.

Format ^BIO, h, f, g



Important • If additional information about the Industrial 2 of 5 bar code, go to www.aimglobal.org.

This table identifies the parameters for this format:

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

➔ **Example •** This is an example of an Industrial 2 of 5 bar code:

ZPL II CODE	INDUSTRIAL 2 OF 5 BAR CODE
<code>^XA</code> <code>^FO100,100^BY3</code> <code>^BIN,150,Y,N</code> <code>^FD123456^FS</code> <code>^XZ</code>	 123456
INDUSTRIAL 2 OF 5 BAR CODE CHARACTERS	
0123456789 Start/Stop (internal)	

^BJ

Standard 2 of 5 Bar Code

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

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.

Format ^BJo, h, f, g



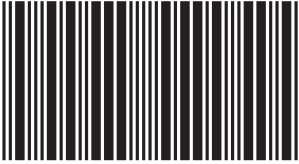
Important • If additional information about the Standard 2 of 5 bar code is required, go to www.aimglobal.org.

This table identifies the parameters for this format:

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



Example • This is an example of a Standard 2 of 5 bar code:

ZPL II CODE	STANDARD 2 OF 5 BAR CODE
<pre>^XA ^FO100,100^BY3 ^BJN,150,Y,N ^FD123456^FS ^XZ</pre>	 123456
STANDARD 2 OF 5 BAR CODE CHARACTERS	
0123456789 Start/Stop (automatic)	

^BK

ANSI Codabar Bar Code

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

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.

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



Important • If additional information about the ANSI Codabar bar code is required, go to www.aimglobal.org.

This table identifies the parameters for this format:

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



Example • This is an example of an ANSI Codabar bar code:

ZPL II CODE

^XA

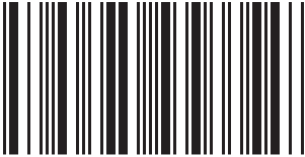
^FO100,100^BY3

^BKN,N,150,Y,N,A,A

^FD123456^FS

^XZ

ANSI CODABAR BAR CODE



A123456A

ANSI CODABAR BAR CODE CHARACTERS

0123456789

Control Characters

- : . \$ / +

Start/Stop Characters

A B C D

^BL

LOGMARS Bar Code

Description 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.

- ^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.

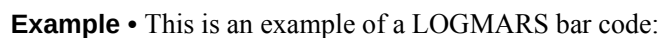
Format ^BL o , h , g



Important • If additional information about the LOGMARS bar code is required, go to www.aimglobal.org.

This table identifies the parameters for this format:

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

[illegible]

Comments The LOGMARS bar code produces a *mandatory* check digit using Mod 43 calculations. For further information on the Mod 43 check digit, see [Mod 43 Check Digit on page 963](#).

^BM

MSI Bar Code

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

Each character in the MSI bar code 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 bar code 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 parameter e is A, plus a quiet zone.

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



Important • If additional information about the MSI bar code is required, go to www.aimglobal.org.

This table identifies the parameters for this format:

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

Parameters	Details
e2 = inserts check digit into the interpretation line	<i>Accepted Values:</i> N = no Y = yes <i>Default Value:</i> N

➔ **Example •** This is an example of a MSI bar code:

ZPL II CODE

^XA
^FO100,100^BY3
^BMN,B,100,Y,N,N
^FD123456^FS
^XZ

MSI BAR CODE



123456

MSI BAR CODE CHARACTERS

1

2

3

4

5

6

7

8

9

^BO

Aztec Bar Code Parameters

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



Note • The Aztec bar code works with firmware version V60.13.0.11A and V50.13.2 or later.

Format ^BOa,b,c,d,e,f,g

This table identifies the parameters for this format:

Parameters	Details
a = orientation	<i>Accepted Values:</i> N = normal R = rotated I = inverted 180 degrees B = read from bottom up, 270 degrees <i>Default Value:</i> current ^FW value
b = magnification factor	<i>Accepted Values:</i> 1 to 10 <i>Default Value:</i> 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	<i>Accepted Values:</i> Y = if data contains ECICs N = if data does not contain ECICs. <i>Default Value:</i> N
d = error control and symbol size/type indicator	<i>Accepted Values:</i> 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” <i>Default Value:</i> 0
e = menu symbol indicator	<i>Accepted Values:</i> Y = if this symbol is to be a menu (bar code reader initialization) symbol N = if it is not a menu symbol <i>Default Value:</i> N

Parameters	Details
f = number of symbols for structured append	<i>Accepted Values:</i> 1 through 26 <i>Default Value:</i> 1
g = optional ID field for structured append	The ID field is a text string with 24-character maximum <i>Default Value:</i> no ID

➔ **Example** • This is an example of the ^B0 command:

ZPL II CODE

^XA
^B0R,7,N,0,N,1,0
^FD 7. This is testing label 7^FS
^XZ

GENERATED LABEL



^BP

Plessey Bar Code

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

Each character in the Plessey bar code 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.

Format ^BPo,e,h,f,g



Important • If additional information about the Plessey bar code is required, go to www.aimglobal.org.

This table identifies the parameters for this format:

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



Example • This is an example of a Plessey bar code:

ZPL II CODE	PLESSEY BAR CODE
<pre>^XA ^FO100,100^BY3 ^BPN,N,100,Y,N ^FD12345^FS ^XZ</pre>	 12345
PLESSEY BAR CODE CHARACTERS	
0123456789 A B C D E F	

^BQ

QR Code Bar Code

Description 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 bar code size, position, and inclination.

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 bar code 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.

Format ^BQa, b, c, d, e



Important • If additional information about the QR Code bar code is required, go to www.aimglobal.org.

This table identifies the parameters for this format:

Parameters	Details
a = field orientation	<i>Fixed Value:</i> normal (^FW has no effect on rotation)
b = model	<i>Accepted Values:</i> 1 (original) and 2 (enhanced – recommended) <i>Default Value:</i> 2
c = magnification factor	<i>Accepted Values:</i> 1 to 10 <i>Default Value:</i> 1 on 150 dpi printers 2 on 200 dpi printers 3 on 300 dpi printers 6 on 600 dpi printers

Parameters	Details
d = H,Q,M,L	<i>Accepted Values:</i> H = ultra-high reliability level Q = high reliability level M = standard level L = high density level <i>Default Value:</i> Q = if empty M = invalid values
e = N,A,B,K	<i>Accepted Values:</i> 1 - 7 <i>Default Value:</i> 7



Example • This is an example of a QR Code bar code:

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

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

Considerations for ^FD When Using the QR Code:

QR Switches (formatted into the ^FD field data)

mixed mode <D>

D = allows 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 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.

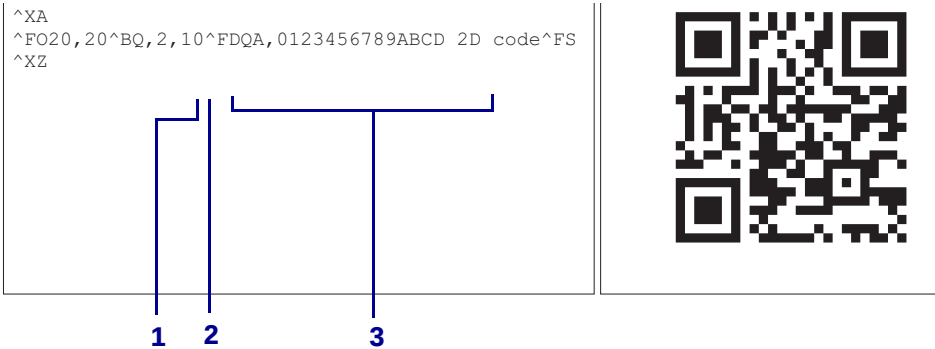
M = Manual Input

Two types of data input mode 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

➔ **Example • 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



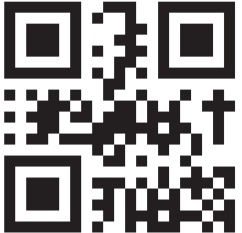
Example • QR Code, normal mode with manual data input:

ZPL II CODE	GENERATED LABEL
<pre> ^XA ^FO20,20^BQ,2,10 ^FDHM,N123456789012345^FS ^XZ </pre>	

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



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

ZPL II CODE	GENERATED LABEL
<pre> ^XA ^FO20,20^BQ,2,10^FDM,AAC-42^FS ^XZ </pre>	

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

➔ **Example • QR Code, mixed mode with manual data input:**

ZPL II CODE	GENERATED LABEL
<pre>^XA ^FO,20,20^BQ,2,10 ^FDD03048F,LM,N0123456789,A12AABB,B0006qrcline^FS ^XZ</pre>	

<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>		qrcline

➡ **Example •** This is an example of QR Code, mixed mode with automatic data input:

```
^XA
^F020,20^BQ,2,10
^FDD03040C,LA,012345678912AABBqrcode^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>	A	(automatic input)
<data character string>		012345678912AABBqrcode



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

```
^XA^SEE:JIS.DAT^CI28
^F020,20^BQ,2,10^FDLM, 日月^FS
^XZ
```

GENERATED LABEL

^BR

GS1 Databar (formerly Reduced Space Symbology)

Description The ^BR command is bar code types for space-constrained identification from EAN International and the Uniform Code Council, Inc.

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

This table identifies the parameters for this format:

Parameters	Details
a = orientation	<i>Accepted Values:</i> - N = Normal - R = Rotated - I = Inverted - B = Bottom-up <i>Default Value:</i> R
b = symbology type in the GS1 DataBar family	<i>Accepted Values:</i> 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/B 12 = UCC/EAN-128 and CC-C <i>Default Value:</i> 1
c = magnification factor	<i>Accepted Values:</i> 1 to 10 <i>Default Value:</i> 24 dot = 6, 12 dot is 3, 8 dot and lower is 2 12 dot = 6, > 8 dot is 3, 8 dot and less is 2
d = separator height	<i>Accepted Values:</i> 1 or 2 <i>Default Value:</i> 1
e = bar code height	The bar code height only affects the linear portion of the bar code. Only UCC/EAN and CC-A/B/C. <i>Accepted Values:</i> 1 to 32000 dots <i>Default Value:</i> 25

Parameters	Details
f = the segment width (GS1 DataBar Expanded only)	<i>Accepted Values:</i> 2 to 22, even numbers only, in segments per line <i>Default Value:</i> 22

→ **Example** • This is an example of Symbology Type 7 - UPC-A:

ZPL II CODE	GENERATED LABEL
<pre>^XA ^FO10,10^BRN,7,5,2,100 ^FD12345678901 this is composite info^FS ^XZ</pre>	

→ **Example** • This is an example of Symbology Type 1 - GS1 DataBar Omnidirectional:

ZPL II CODE	GENERATED LABEL
<pre>^XA ^FO10,10^BRN,1,5,2,100 ^FD12345678901 this is composite info^FS ^XZ</pre>	

^BS

UPC/EAN Extensions

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

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

Format ^BS*o, h, f, g*



Important • If additional information about the UPC/EAN bar code is required, go to www.aimglobal.org.

This table identifies the parameters for this format:

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



Example • This is an example of a UPC/EAN Two-digit bar code:

ZPL II CODE	UPC/EAN 2-DIGIT BAR CODE								
<pre>^XA ^FO100,100^BY3 ^BSN,100,Y,N ^FD12^FS ^XZ</pre>									
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 bar code:

ZPL II CODE	UPC/EAN 5-DIGIT BAR CODE								
<pre>^XA ^FO100,100^BY3 ^BSN,100,Y,N ^FD12345^FS ^XZ</pre>	 12345								
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.

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

➔ **Example** • This is an example of a UPC-A:

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

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 bar code 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 bar code for the value 7000002198 with an extension equal to 04414:

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

^BT

TLC39 Bar Code

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

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

Format ^BT $o, w1, r1, h1, w2, h2$

This table identifies the parameters for this format:

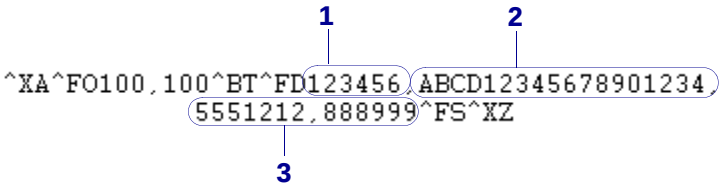
Parameters	Details
o = orientation	<i>Accepted Values:</i> N = normal R = rotated I = inverted B = bottom up
$w1$ = width of the Code 39 bar code	<i>Accepted Value (in dots): 1 to 10</i> <i>Default Value (600 dpi printers): 4</i> <i>Default Value (200- and 300 dpi printer): 2</i>
$r1$ = wide to narrow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



Example • TLC39 Bar Code

This is an example on how to print TLC39 bar code. 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	GENERATED LABEL
<pre>^XA^FO100, 100^BT^FD123456, ABCD12345678901234, 5551212, 888999 ^FS^XZ</pre>	

^BU

UPC-A Bar Code

Description The ^BU command produces a fixed length, numeric symbology. It is primarily used in the retail industry for labeling packages. The UPC-A bar code has 11 data characters. The 6 dot/mm, 12 dot/mm, and 24 dot/mm printheads produce the UPC-A bar code (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.

- ^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 required number of characters.

Format ^BUo, h, f, g, e



Important • If additional information about the UPC-A bar code is required, go to www.aimglobal.org.

This table identifies the parameters for this format:

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

The font style of the interpretation line depends on the modulus (width of narrow bar) selected in ^BY:



Note • 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 bar code with extension:

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

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 962](#).

^BX

Data Matrix Bar Code

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

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

This table identifies the parameters for this format:

Parameters	Details
O = orientation	<i>Accepted Values:</i> N = normal R = rotated 90 degrees (clockwise) I = inverted 180 degrees B = read from bottom up, 270 degrees <i>Default Value:</i> current ^FW value
h = dimensional height of individual symbol elements	<i>Accepted Values:</i> 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	<i>Accepted Values:</i> 0, 50, 80, 100, 140, 200 <i>Default Value:</i> 0 <i>Quality</i> 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.

Parameters	Details
c = columns to encode	<i>Accepted Values: 9 to 49</i> 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 are 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 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	<i>Accepted Values: 9 to 49</i>
f = format ID (0 to 6) — not used with quality set at 200	<i>Accepted Values:</i> 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 <i>Default Value: 6</i>
g = escape sequence control character	<i>Accepted Values: any character</i> <i>Default Value: ~ (tilde)</i> This parameter is used only if quality 200 is specified. It is the escape character for embedding special control sequences within the field data. Important • 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.

Parameters	Details
a = aspect ratio .16↑ The a parameter is only supported in V60.16.5Z and V53.16.5Z or later.	<i>Accepted Values:</i> 1 = square 2 = rectangular <i>Default Value:</i> 1

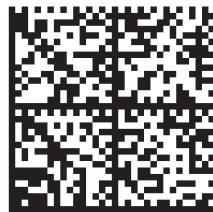
Table 10 • Maximum Field Sizes

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

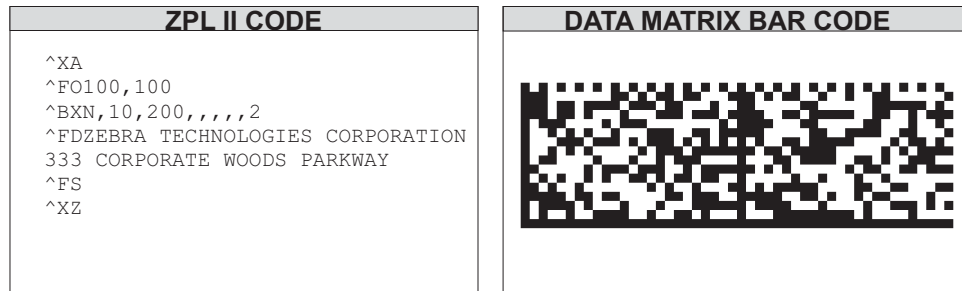


Example • This is an example of a square Data Matrix bar code:

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:



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 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 || can be used to insert carriage returns, line feeds, and the 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 version: 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 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 `~` (tilde)

^BY

Bar Code Field Default

Description 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.

Format ^BYw, r, h

This table identifies the parameters for this format:

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

For parameter r, the actual ratio generated is a function of the number of dots in parameter w, module width. See [Table 11 on page 143](#).



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.

Module width and height (w and h) can be changed at anytime with the ^BY command, regardless of the symbology selected.

Table 11 • Shows 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	2.1:1	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	2.4:1
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

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

^BZ

POSTAL Bar Code

Description The POSTAL bar code 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 bar code specification.

Format ^BZo, h, f, g, t



Important • If additional information about the POSTAL and PLANET bar code is required, go to www.aimglobal.org, or contact the United States Postal Service <http://pe.usps.gov>. If additional information about the INTELLIGENT MAIL bar code is required, see: <http://ribbs.usps.gov/OneCodeSolution>.

This table identifies the parameters for this format:

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

➔ **Example •** This is an example of a POSTNET bar code:

ZPL II CODE	POSTNET BAR CODE
^XA ^FO100,100^BY3 ^BZN,40,Y,N ^FD12345^FS ^XZ	
POSTNET BAR CODE CHARACTERS	
0 1 2 3 4 5 6 7 8 9	

➔ **Example •** This is an example of a USPS Intelligent Mail bar code:

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

^CC ~CC

Change Caret

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

Format ^CCx or ~CCx

This table identifies the parameters for this format:

Parameters	Details
X = caret character change	<i>Accepted Values:</i> any ASCII character <i>Default Value:</i> 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

Change Delimiter

Description 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 (,).

Format ^CDa or ~CDa

This table identifies the parameters for this format:

Parameters	Details
a = delimiter character change	<i>Accepted Values:</i> any ASCII character <i>Default Value:</i> 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

^F010,10

^GB10,10,3

^XZ

^XA

^CD;

^F010;10

^GB10;10;3

^XZ

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

~CD;

^XA^JUS^XZ

^CF

Change Alphanumeric Default Font

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

Format ^CFf,h,w

This table identifies the parameters for this format:

Parameters	Details
f = specified default font	<i>Accepted Values:</i> A through Z and 0 to 9 <i>Initial Value at power-up:</i> A
h = individual character height (in dots)	<i>Accepted Values:</i> 0 to 32000 <i>Initial Value at power-up:</i> 9
w = individual character width (in dots)	<i>Accepted Values:</i> 0 to 32000 <i>Initial Value at power-up:</i> 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	GENERATED LABEL
<pre> ^XA ^CF0,89 ^FO20,50 ^FDA GUIDE TO^FS ^FO20,150 ^FDTHE ZPL II^FS ^FO20,250 ^FDPROGRAMMING^FS ^FO20,350 ^FDLANGUAGE^FS ^XZ </pre>	

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

Change International Font/Encoding

Description 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.

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.

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 parameter **a** = 0-13.

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

This table identifies the parameters for this format:

Parameters	Details
a = desired character set	Accepted values 0 - 12 are Zebra Code Page 850 with specific character replacements. For details, see International Character Sets on page 152 and/or Zebra Code Page 850 — Latin Character Set on page 930 . <i>Accepted Values:</i> 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 <i>(parameter details continued on next page)</i>



Note • These parameters are only valid when parameter **a** = 1 - 13

- 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 fonts internal encoding (Unicode).
- 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.
- 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.
- 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.

Parameters	Details
a = desired character set (continued) .14↑ Values 28 to 30 are only supported in firmware version V60.14.x, V50.14.x, or later. .16↑ Values 31 to 36 are only supported in firmware version x.16.x or later.	11 = Single Byte Encoding - Miscellaneous Character Set 12 = Single Byte Encoding - Japan (ASCII with Yen symbol) Character Set 13 = Zebra Code Page 850 (see page 930) 14 = Double Byte Asian Encodings ^a 15 = Shift-JIS ^b 16 = EUC-JP and EUC-CN ^a 17 = Deprecated - UCS-2 Big Endian ^d 18 to 23 = Reserved 24 = Single Byte Asian Encodings ^a 25 = Reserved 26 = Multibyte Asian Encodings with ASCII Transparency ^a and ^c 27 = Zebra Code Page 1252 (see page 936) 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 page 932) 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 <i>Initial Value at power-up: 0</i>
s1 = source 1 (character output image)	<i>Accepted Values:</i> decimals 0 to 255
d1 = destination 1 (character input)	<i>Accepted Values:</i> decimals 0 to 255
s2 = source 2 (character output image)	<i>Accepted Values:</i> decimals 0 to 255

- a.** 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 fonts internal encoding (Unicode).
- b.** 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.
- c.** 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.
- d.** 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.

Parameters	Details
d2 = destination 2 (character input)	<i>Accepted Values:</i> decimals 0 to 255
... = continuation of pattern	Up to 256 source and destination pairs can be entered in this command.

- a. 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 fonts internal encoding (Unicode).
- b. 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.
- c. 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.
- d. 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↑

80 to FF could mean 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
<pre>^XA ^CI0,21,36 ^FO100,200^A0N50,50^FD\$0123^FS ^XZ</pre>	

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

International Character Sets

Hex	2	3	4	5	5	5	5	6	7	7	7	7
	3	0	0	B	C	D	E	0	B	C	D	E
CI0	#	0	@	[	Φ	]	^	'	{		}	~
CI1	#	0	@	¼	Φ	¾	^	'	¼	½	¾	~
CI2	£	0	@	[	Φ	]	^	'	{		}	~
CI3	f	0	§	[	U	]	^	'	{	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	§	í	Ñ	¿	^	'	{	ñ	ç	~
CI11	£	0	É	Ä	Ö	Ü	^	'	ä	ë	ï	ö
CI12	#	0	@	[	¥	]	^	'	{		}	~
CI13	#	0	@	[	\	]	^	'	{		}	~



Note • ^CI 13 = US keyboard

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

^CM

Change Memory Letter Designation

Description 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.

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

Format ^CMa, b, c, d

This table identifies the parameters for this format:

Parameters	Details
a = memory alias for B:	<i>Accepted Values:</i> B:, E:, R:, A:, and NONE <i>Default Value:</i> B:
b = memory alias for E:	<i>Accepted Values:</i> B:, E:, R:, A:, and NONE <i>Default Value:</i> E:
c = memory alias for R:	<i>Accepted Values:</i> B:, E:, R:, A:, and NONE <i>Default Value:</i> R:
d = memory alias for A:	<i>Accepted Values:</i> B:, E:, R:, A:, and NONE <i>Default Value:</i> A:

Comments If 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 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
```

^CN

Cut Now

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



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 299.

Supported Devices

- KR403

Format ^CNa

This table identifies the parameters for this format:

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

^CO

Cache On



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.

Description 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.

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

This table identifies the parameters for this format:

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



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

^COY, 40

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 ^CO 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 ^CO 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.

^CP

Remove Label

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

Supported Devices

- KR403

Format ^CPa

This table identifies the parameters for this format:

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

^CT ~CT

Change Tilde

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

Format ^CTa or ~CTa

This table identifies the parameters for this format:

Parameters	Details
a = change control command character	<i>Accepted Values:</i> any ASCII character <i>Default Value:</i> 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

Code Validation

Description 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:

- 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

This table identifies the parameters for this format:

Parameters	Details
a = code validation	<i>Accepted Values:</i> N = no Y = yes <i>Default Value:</i> 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.

ZPL II CODE	GENERATED LABEL
^XA ^CVY ^FO50,50 ^BEN,100,Y,N ^FD97823456 890^FS ^XZ	
^XA ^CVY ^FO50,50 ^BEN,100,Y,N ^FD9782345678907^FS ^XZ	
^XA ^CVY ^FO50,50 ^BEN,100,Y,N ^FD97823456789081^FS ^XZ	
^XA ^CVY ^FO50,50 ^BEN,100,Y,N ^FD97823456789^FS ^XZ	
^XA ^CVY ^FO50,50 ^BQN2,3 ^FDHM,BQRCODE-22^FS ^XZ	

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

Font Identifier

Description 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.

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

This table identifies the parameters for this format:

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

DIRECTORY OF R:*. * *	
R:NEWFONT.FNT	65268
R:MYFONT.FNT	65268
582164 BYTES FREE R:	

Label Listing Before Assignment

DIRECTORY OF R:*. * *	
F R:NEWFONT.FNT	65268
AQ R:MYFONT.FNT	65268
582164 BYTES FREE R:	

Label Listing After Assignment

~DB

Download Bitmap Font

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

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

This table identifies the parameters for this format:

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

Parameters	Details
<code>data</code> = 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: <pre>#xxx.h.w.x.y.i.data</pre> #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) <code>data</code> = 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

Download Encoding

Description 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 www.zebra.com.

Format ~DEd:o.x,s,data

This table identifies the parameters for this format:

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

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".

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

^DF

Download Format

Description 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.

While 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

This table identifies the parameters for this format:

Parameters	Details
d = device to store image	Accepted Values: R:, E:, B:, and A: Default Value: R:
o = image name	Accepted Values: 1 to 8 alphanumeric characters Default Value: if a name is not specified, UNKNOWN is used
x = extension	Fixed Value: .ZPL

For a complete example of the ^DF and ^XF command, see [^DF and ^XF — Download format and recall format on page 49](#).

➔ **Example** • This example is generated using the ^XF command to recall this format:

ZPL II CODE	GENERATED LABEL
<pre>^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</pre>	ZEBRA PRINTER BUILT BY ZEBRA

~DG

Download Graphics

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

For more saving and loading options when downloading files, see [~DY on page 175](#).

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

This table identifies the parameters for this format:

Parameters	Details
d = device to store image	<i>Accepted Values:</i> R:, E:, B:, and A: <i>Default Value:</i> R:
o = image name	<i>Accepted Values:</i> 1 to 8 alphanumeric characters <i>Default Value:</i> if a name is not specified, UNKNOWN is used
x = extension	<i>Fixed Value:</i> .GRF
t = total number of bytes in 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



Examples • These are some example related to the ~DG command:

To determine the t parameter use this formula:

$$\frac{xz}{8} \times yz = \text{totalbytes}$$

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:

$$\begin{aligned} 8 \times 128 &= 1024 \\ t &= 1024 \end{aligned}$$

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

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

$$w = 8$$

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

$$w = 8$$

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.

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
<pre> ~DGR:SAMPLE.GRF,00080,010, FFFFFFFFFFFFFFFFFFFF 8000FFFF0000FFFF0001 8000FFFF0000FFFF0001 8000FFFF0000FFFF0001 FFFF0000FFFF0000FFFF FFFF0000FFFF0000FFFF FFFF0000FFFF0000FFFF FFFFFFFFFFFFFFFFFFFF ^XA ^F020,20^XGR:SAMPLE.GRF,1,1^FS ^XZ </pre>	

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

Abort Download Graphic

Description 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.

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

Download Intellifont (Scalable Font)

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

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

This table identifies the parameters for this format:

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



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 173](#), [~DU on page 174](#), and [~DY on page 175](#).

~DT

Download Bounded TrueType Font

Description 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 174](#). ZTools creates a downloadable file that includes a ~DT command. For information on converting and downloading Intellifont information, see [~DS on page 172](#).

Format ~DTd:o.x,s,data

This table identifies the parameters for this format:

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

Download Unbounded TrueType Font

Description 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. 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

This table identifies the parameters for this format:

Parameters	Details
d = font location	<i>Accepted Values:</i> R:, E:, B:, and A: <i>Default:</i> R:
O = font name	<i>Accepted Values:</i> 1 to 8 alphanumeric characters <i>Default Value:</i> if a name is not specified, UNKNOWN is used
x = extension	<i>Format:</i> .FNT
s = font size	<i>Accepted Values:</i> the number of memory bytes required to hold the Zebra-downloadable format of the font <i>Default Value:</i> if no data is entered, the command is ignored
data = data string	<i>Accepted Values:</i> a string of ASCII hexadecimal values (two hexadecimal digits/byte). The total number of two-digit values must match parameter S. <i>Default Value:</i> 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 172](#), [~DT on page 173](#), and [~DY on page 175](#).

~DY

Download Objects

Description 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.



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 ~DYd:f,b,x,t,w,data

This table identifies the parameters for this format:

Parameters	Details
d = file location  .15† .NRD and .PAC files reside on E: in firmware versions V60.15.x, V50.15.x, or later.	<i>Accepted Values:</i> R:, E:, B:, and A: <i>Default Value:</i> R:
f = file name	<i>Accepted Values:</i> 1 to 8 alphanumeric characters <i>Default Value:</i> 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.	<i>Accepted Values:</i> 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 <i>Default Value:</i> 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.	<i>Accepted Values:</i> 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) <i>Default Value:</i> a value other than the accepted values defaults to .GRF
t = total number of bytes in file .14† .TTE is only supported in firmware versions V60.14.x, V50.14.x, or later.	<i>Accepted Values:</i> .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
<code>w</code> = 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.	<i>Accepted Values:</i> .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
<code>data</code> = data	ASCII hexadecimal encoding, ZB64, or binary data, depending on <code>b</code>. 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 datastreams. 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,,
```

**Examples** • These examples show:

- that when the `^IM` command is used with the `^FO` command, the `^IM` command (see [^IM on page 244](#)) 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 242](#)) 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
^XZ
```

Comments For more information on ZB64 encoding and compression, see [ZB64 Encoding and Compression on page 965](#).



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 (`.OTF`) are supported if they are downloaded as a TrueType font. In the printer `.OTF` fonts have the `.TTF` extension.

~EG

Erase Download Graphics

See [^ID](#) on page 240.

^FB

Field Block

Description 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.

Format ^FBa, b, c, d, e

This table identifies the parameters for this format:

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



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

ZPL II CODE	GENERATED LABEL
<pre> ^XA ^CF0,30,30^FO25,50 ^FB250,4,, ^FDFD command that IS\& preceded by an FB \&command. ^FS ^XZ </pre>	FD command that IS preceded by an FB command.
<pre> ^XA ^CF0,30,30^FO25,50 ^FDFD command that IS NOT preceded by an FB command.^FS ^XZ </pre>	FD command that IS NOT preceded by an FB cor

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 ^FO (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.



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 ^FO and ^FT that define the justification of the origin.

The ^FB command does not support soft hyphens as a potential line break point. 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

Field Clock

Description 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.

Format ^FCa,b,c

This table identifies the parameters for this format:

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



Example • 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.

ZPL II CODE	GENERATED LABEL
<pre>^XA ^FO10,100^A0N,50,50 ^FC%,{,# ^FDPrimary: %m/%d/%y^FS ^FO10,200^A0N,50,50 ^FC%,{,# ^FDSecondary: {m/{d/{y^FS ^FO10,300^A0N,50,50 ^FC%,{,# ^FDThird: #m/#d/#y^FS ^XZ</pre>	Primary: 00/00/00 Secondary: 01/01/00 Third: 01/01/00

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 977](#).

^FD

Field Data

Description The ^FD command defines the data string for the field. The field data can be any printable character except those used as command prefixes (^ and ~).

Format ^FDa

This table identifies the parameters for this format:

Parameters	Details
a = data to be printed	<i>Accepted Values:</i> any data string up to 3072 bytes <i>Default Value:</i> 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 147](#) and [^CT ~CT on page 158](#). 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 [Comments on the ^FB command on page 181](#).

^FH

Field Hexadecimal Indicator

Description 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.

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

This table identifies the parameters for this format:

Parameters	Details
a = hexadecimal indicator	<i>Accepted Values:</i> any character except current format and control prefix (^ and ~ by default) <i>Default Value:</i> _ (underscore)

➔ **Example** • This is an example of how to enter a hexadecimal value directly into a ^FD statement: This is an example for ascii data using ^CI0.

ZPL II CODE	GENERATED LABEL
<pre>^XA ^FO100,100 ^AD^FH ^FDTilde _7e used for HEX^FS ^XZ</pre>	Tilde ~ used for HEX
<pre>^XA ^FO100,100 ^AD^FH\ ^FDTilde \7E used for HEX^FS ^XZ</pre>	Tilde ~ used for HEX

➔ **Examples** • These are examples of how ^FH works with UTF-8 and UTF-16BE:

UTF-8

ZPL II CODE	GENERATED LABEL
<pre>^XA ^CI28 ^LL500 ^FO100,100 ^AA,20,20 ^FH^FDU+00A1 in UTF8 = _C2_A1^FS ^XZ</pre>	U+00A1 in UTF8 = ;

UTF-16BE

ZPL II CODE	GENERATED LABEL
<pre>^XA ^CI29 ^LL500 ^FO100,100 ^AA,20,20 ^FH^FDU+00A1 in UTF16BE = _00_A1^FS ^XZ</pre>	U+00A1 in UTF16BE = ;

^FL

Font Linking



This command is available only for printers with firmware versions V60.14.x, V50.14.x, or later.

Description The ^FL command provides the ability to link any TrueType font, including private character fonts, to associated fonts.

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, please use the font wizard in ZebraNet Bridge.

Format ^FL<ext>,<base>,<link>

This table identifies the parameters for this format:

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) to which <ext> is associated. This can be partially or fully wild carded; all files that match the wildcard specification will have this extension associated with it. 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 associated 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: <i>Accepted Values:</i> 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> <i>Default Value:</i> must be an accepted value or it is ignored

Comments A font can have up to five fonts linked to it. The printer 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.



Examples • 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 the Asian glyphs, which is why Asian glyphs do not print.

ZPL II CODE	GENERATED LABEL
<pre>^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</pre>	NO FONT LINKING TEST WITH SWISS721 DRAGONFLY TEST WITH ANMDJ DRAGONFLY 蜻蜓

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 reprints, the Asian characters are printed using the ANMDJ . TTF font, rather than the SWISS721 . TTF font.

ZPL II CODE	GENERATED LABEL
^XA^LL1200^CW1,E:SWISS721.TTF^CW2,E:ANMDJ.TTF^CI28^FS ^F0100,100^A0,50,50^FDFONT LINKING^FS ^F0100,300^A1,50,50^FDTEST WITH SWISS721^FS ^F0100,400^A1,50,50^FDDRAGONFLY 蜻蜓^FS ^F0100,600^A2,50,50^FDTEST WITH ANMDJ^FS ^F0100,700^A2,50,50^FDDRAGONFLY 蜻蜓^FS ^XZ	FONT LINKING TEST WITH SWISS721 DRAGONFLY 蜻蜓 TEST WITH ANMDJ DRAGONFLY 蜻蜓

^FM

Multiple Field Origin Locations

Description The ^FM command allows you to control the placement of bar code symbols.

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, . . .

This table identifies the parameters for this format:

Parameters	Details
x1 = x-axis location of first symbol (in dots)	<i>Accepted Values:</i> 0 to 32000 e = exclude this bar code from printing <i>Default Value:</i> a value must be specified
y1 = y-axis location of first symbol (in dots)	<i>Accepted Values:</i> 0 to 32000 e = exclude this bar code from printing <i>Default Value:</i> a value must be specified
x2 = x-axis location of second symbol (in dots)	<i>Accepted Values:</i> 0 to 32000 e = exclude this bar code from printing <i>Default Value:</i> a value must be specified
y2 = y-axis location of second symbol (in dots)	<i>Accepted Values:</i> 0 to 32000 e = exclude this bar code from printing <i>Default Value:</i> a value must be specified
... = continuation of X,Y pairs	<i>Maximum number of pairs:</i> 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:

ZPL II CODE	GENERATED LABEL
<pre>^XA ^FM100,100,100,600,100,1200 ^BY2,3 ^B7N,5,5,9,83,N ^FDZebra Technologies Corporation strives to be the expert supplier of innovative solutions to specialty 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</pre>	

1

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:

ZPL II CODE	GENERATED LABEL
<pre>^XA ^FM100,100,e,e,100,1200 ^BY2,3 ^B7N,5,5,9,83,N ^FDZebra Technologies Corporation strives to be the expert supplier of innovative solutions to specialty 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</pre>	

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.

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

Field Number

Description The ^FN command numbers the data fields. This command is used in both ^DF (Store Format) and ^XF (Recall Format) commands.

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"

This table identifies the parameters for this format:

Parameters	Details
# = number to be assigned to the field	<i>Accepted Values:</i> 0 to 9999 <i>Default Value:</i> 0
"a" = optional parameter*	<i>Accepted Values:</i> 255 alphanumeric characters maximum (a-z,A-Z,1-9 and space) <i>Default Value:</i> 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 [^DF and ^XF — Download format and recall format on page 49](#).

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).



Example • The ^FN1 "Name" would result in "Name" being used as the prompt on the KDU unit.

^FO

Field Origin

Description The ^FO command sets a field origin, relative to the label home (^LH) position. ^FO sets the upper-left corner of the field area by defining points along the x-axis and y-axis independent of the rotation.

Format ^FOx,y,z

This table identifies the parameters for this format:

Parameters	Details
x = x-axis location (in dots)	Accepted Values: 0 to 32000 Default Value: 0
y = y-axis location (in dots)	Accepted Values: 0 to 32000 Default Value: 0
z = justification	Accepted Values: 0 = left justification 1 = right justification 2 = auto justification (script dependent) Default Value: last accepted ^FW value or ^FW default

.14↑

The z parameter is only supported in firmware versions V60.14.x, V50.14.x, or later.

Comments If the value entered for the x or y parameter is too high, it could position the field origin completely off the label.



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 971](#).

The auto justification option might cause unexpected results if variable fields or bidirectional text are used with ^FO. For the best results with bidirectional text and/or variable fields, use either the left or right justification option.

^FP

Field Parameter

Description The ^FP command allows vertical and reverse formatting of the font field, commonly used for printing Asian fonts.

Format ^FPd,g

This table identifies the parameters for this format:

Parameters	Details
d = direction	<i>Accepted Values:</i> H = horizontal printing (left to right) V = vertical printing (top to bottom) R = reverse printing (right to left) <i>Default Value:</i> H
g = additional inter-character gap (in dots)	<i>Accepted Values:</i> 0 to 9999 <i>Default Value:</i> 0 if no value is entered

➔ **Example** • This is an example of how to implement reverse and vertical print:

ZPL II CODE	GENERATED LABEL
<pre>^XA ^FO100,50 ^FPV,10 ^AV ^FDvertical^FS ^XZ</pre>	
<pre>^XA ^FO350,50 ^FPR,10 ^AV ^FDreverse^FS ^XZ</pre>	



For vertical and reverse printing directions, combining semantic clusters are used to place characters.

This command interacts with the justification parameters of ^FO and ^FT and with the rotation parameter of ^A. For output and examples, see [Field Interactions on page 971](#).

^FR

Field Reverse Print

Description 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.

Format ^FR



Example • In this example, the ^GB command creates areas of black allowing the printing to appear white:

ZPL II CODE	GENERATED LABEL
<pre> ^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 </pre>	

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

Field Separator

Description 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).

Format ^FS

^FT

Field Typeset

Description 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.



Note • The ^FT command is capable of concatenation of fields.

Format ^FTx,y,z

This table identifies the parameters for this format:

Parameters	Details
x = x-axis location (in dots)	<i>Accepted Values:</i> 0 to 32000 <i>Default Value:</i> position after last formatted text field
y = y-axis location (in dots)	<i>Accepted Values:</i> 0 to 32000 <i>Default Value:</i> position after last formatted text field
z = justification	<i>Accepted Values:</i> 0 = left justification 1 = right justification 2 = auto justification (script dependent) <i>Default Value:</i> 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.



The z parameter is only supported in firmware version V60.14.x, V50.14.x, or later.

Table 12 • Typeset Justification

Left Justified	Text	For examples, see Field Interactions on page 971 .
	Bar Codes	Origin is base of bar code, at left edge
	Graphic Boxes	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 971 .
	Bar Codes	Origin is base of bar code, at 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
<pre>^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</pre>	

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



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 971](#)

^FV

Field Variable

Description ^FV replaces the ^FD (field data) command in a label format when the field is variable.

Format ^FVa

This table identifies the parameters for this format:

Parameters	Details
a = variable field data to be printed	<i>Accepted Values:</i> 0 to 3072 byte string <i>Default Value:</i> 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	GENERATED LABEL
<pre>^XA ^FO40,40 ^GB300,203,8^FS ^FO55,60^CF0,25 ^FVARIABLE DATA #1^FS ^FO80,150 ^FDFIXED DATA^FS ^MCN ^XZ</pre>	VARIABLE DATA #1 FIXED DATA
<pre>^XA ^FO55,60^CF0,25 ^FVARIABLE DATA #2^FS ^MCY ^XZ</pre>	VARIABLE DATA #2 FIXED DATA

Comments ^FV fields are always cleared after the label is printed. ^FD fields are not cleared.

^FW

Field Orientation

Description 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.

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

This table identifies the parameters for this format:

Parameters	Details
r = rotate field	<i>Accepted Values:</i> N = normal R = rotated 90 degrees I = inverted 180 degrees B = bottom-up 270 degrees, read from bottom up <i>Initial Value at Power-up:</i> N
z = justification  The z parameter is available only with printers with firmware version V60.14.x, V50.14.x, or later.	<i>Accepted Values:</i> 0 = left justification 1 = right justification 2 = auto justification (script dependent) <i>Default Value:</i> auto for ^TB and left for all other commands



Example • This example shows how ^FW rotation works in conjunction with ^FO. 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
<pre>^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^FDXXXXXXXXXX^FS ^XZ</pre>	

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.



^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

Comment

Description 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.

Format ^FXc

This table identifies the parameters for this format:

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	GENERATED LABEL
<pre>^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</pre>	

Comments Correct usage of the ^FX command includes following it with the ^FS command.

^GB

Graphic Box

Description 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 a label. The same format command is used for drawing either boxes or lines.

Format ^GBw, h, t, c, r

This table identifies the parameters for this format:

Parameters	Details
w = box width (in dots)	<i>Accepted Values:</i> value of t to 32000 <i>Default Value:</i> value used for thickness (t) or 1
h = box height (in dots)	<i>Accepted Values:</i> value of t to 32000 <i>Default Value:</i> value used for thickness (t) or 1
t = border thickness (in dots)	<i>Accepted Values:</i> 1 to 32000 <i>Default Value:</i> 1
c = line color	<i>Accepted Values:</i> B = black W = white <i>Default Value:</i> B
r = degree of corner-rounding	<i>Accepted Values:</i> 0 (no rounding) to 8 (heaviest rounding) <i>Default Value:</i> 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 value t.

The roundness-index is used to determine a rounding-radius for each box. Formula:

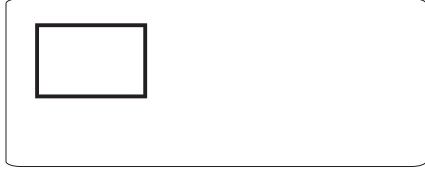
$$\text{rounding-radius} = (\text{rounding-index} / 8) * (\text{shorter side} / 2)$$

where the shorter side is the lesser of the width and height (after adjusting for minimum and default values).



Examples • Here are a few examples of graphic boxes:

Width: 1.5 inch; Height: 1 inch; Thickness: 10; Color: default; Rounding: default

ZPL II CODE	GENERATED LABEL
<pre>^XA ^FO50,50 ^GB300,200,10^FS ^XZ</pre>	

Width: 0 inch; Height: 1 inch; Thickness: 20; Color: default; Rounding: default:

ZPL II CODE	GENERATED LABEL
<pre>^XA ^FO50,50 ^GB0,203,20^FS ^XZ</pre>	

Width: 1 inch; Height: 0 inch; Thickness: 30; Color: default; Rounding: default

ZPL II CODE	GENERATED LABEL
<pre>^XA ^FO50,50 ^GB203,0,20^FS ^XZ</pre>	

Width: 1.5 inch; Height: 1 inch; Thickness: 10; Color: default; Rounding: 5

ZPL II CODE	GENERATED LABEL
<pre>^XA ^FO50,50 ^GB300,200,10,,5^FS ^XZ</pre>	

^GC

Graphic Circle

Description 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.

Format ^GCd, t, c

This table identifies the parameters for this format:

Parameters	Details
d = circle diameter (in dots)	<i>Accepted Values:</i> 3 to 4095 (larger values are replaced with 4095) <i>Default Value:</i> 3
t = border thickness (in dots)	<i>Accepted Values:</i> 2 to 4095 <i>Default Value:</i> 1
c = line color	<i>Accepted Values:</i> B = black W = white <i>Default Value:</i> B



Example • This is an example of how to create a circle on the printed label:

ZPL II CODE	GENERATED LABEL
<pre>^XA ^FO50,50 ^GC250,10,B^FS ^XZ</pre>	

^GD

Graphic Diagonal Line

Description 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.

Format ^GDw,h,t,c,o

This table identifies the parameters for this format:

Parameters	Details
w = box width (in dots)	<i>Accepted Values:</i> 3 to 32000 <i>Default Value:</i> value of t (thickness) or 3
h = box height (in dots)	<i>Accepted Values:</i> 3 to 32000 <i>Default Value:</i> value of t (thickness) or 3
t = border thickness (in dots)	<i>Accepted Values:</i> 1 to 32000 <i>Default Value:</i> 1
c = line color	<i>Accepted Values:</i> B = black W = white <i>Default Value:</i> B
o = orientation (direction of the diagonal)	<i>Accepted Values:</i> R (or /) = right-leaning diagonal L (or \) = left-leaning diagonal <i>Default Value:</i> 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	GENERATED LABEL
<pre>^XA ^FO150,100 ^GB350,203,10^FS ^FO155,110 ^GD330,183,10,,R^FS ^XZ</pre>	

^GE

Graphic Ellipse

Description The ^GE command produces an ellipse in the label format.

Format ^GEw, h, t, c

This table identifies the parameters for this format:

Parameters	Details
w = ellipse width (in dots)	<i>Accepted Values:</i> 3 to 4095 (larger values are replaced with 4095) <i>Default Value:</i> value used for thickness (t) or 1
h = ellipse height (in dots)	<i>Accepted Values:</i> 3 to 4095 <i>Default Value:</i> value used for thickness (t) or 1
t = border thickness (in dots)	<i>Accepted Values:</i> 2 to 4095 <i>Default Value:</i> 1
c = line color	<i>Accepted Values:</i> B = black W = white <i>Default Value:</i> B



Example • This is an example of how to create an ellipse on a printed label:

ZPL II CODE	GENERATED LABEL
<pre>^XA ^FO100,100 ^GE300,100,10,B^FS ^XZ</pre>	

^GF

Graphic Field

Description 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.

Format ^GFa,b,c,d,data

This table identifies the parameters for this format:

Parameters	Details
a = compression type	<i>Accepted Values:</i> 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.) <i>Default Value:</i> A
b = binary byte count	<i>Accepted Values:</i> 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. <i>Default Value:</i> command is ignored if a value is not specified
c = graphic field count	<i>Accepted Values:</i> 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). <i>Default Value:</i> command is ignored if a value is not specified
d = bytes per row	<i>Accepted Values:</i> 1 to 99999 This is the number of bytes in the downloaded data that comprise one row of the image. <i>Default Value:</i> command is ignored if a value is not specified

Parameters	Details
data = data	<i>Accepted Values:</i> 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.

```
^F0100,100^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.

```
^F0100,100^GFB,8000,8000,80,Binary data
```

^GS

Graphic Symbol

Description The ^GS command enables you to generate the registered trademark, copyright symbol, and other symbols.

Format ^GS o, h, w

This table identifies the parameters for this format:

Parameters	Details
o = field orientation	<i>Accepted Values:</i> N = normal R = rotate 90 degrees clockwise I = inverted 180 degrees B = bottom-up, 270 degrees <i>Default Value:</i> N or last ^FW value
h = character height proportional to width (in dots)	<i>Accepted Values:</i> 0 to 32000 <i>Default Value:</i> last ^CF value
w = character width proportional to height (in dots)	<i>Accepted Values:</i> 0 to 32000 <i>Default Value:</i> last ^CF value



Example • 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	GENERATED LABEL
^XA^CFD ^FO50,50 ^FDZEBRA PROGRAMMING^FS ^FO50,75 ^FDLANGUAGE II (ZPL II)^FS ^FO280,75 ^GS^FDC^FS ^XZ	ZEBRA PROGRAMMING LANGUAGE II (ZPL IITM)

A = ® (Registered Trade Mark)

B = © (Copyright)

C = ™ (Trade Mark)

D = UL (Underwriters Laboratories approval)

E = SPC (Canadian Standards Association approval)

~HB

Battery Status

Description 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.



Important • This command only responds to mobile printers.

Format ~HB

Parameters: when the printer receives the command, it returns:

<STX>bb.bb, hh.hh, bt<ETX><CR><LF>

<STX>	=	ASCII start-of-text character
bb.bb	=	current battery voltage reading to the nearest 1/4 volt
hh.hh	=	current head voltage reading to the nearest 1/4 volt
bt	=	battery temperature in Celsius
<ETX>	=	ASCII end-of-text character
<CR>	=	ASCII carriage return
<LF>	=	ASCII line feed character

Comments This command is used for the power-supply battery of the printer and should not be confused with the battery backed-up RAM.

~HD

Head Diagnostic

Description The ~HD command echoes printer status information that includes the power supply and head temperature using the terminal emulator.

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
P33 INTERFACE = None
P34 INTERFACE = None
P35 INTERFACE = None
Dynamic_top_position = 0008
No ribbon A/D = 0000
```

Revision 5

^HF

Host Format

Description The ^HF command sends stored formats to the host.

Format ^HFd,o,x

This table identifies the parameters for this format:

Parameters	Details
d = device to recall image	<i>Accepted Values:</i> R:, E:, B:, and A: <i>Default Value:</i> R:
o = image name	<i>Accepted Values:</i> 1 to 8 alphanumeric characters <i>Default Value:</i> if a name is not specified, UNKNOWN is used
x = extension	<i>Fixed Value:</i> .ZPL



Example • This example shows the sequence and results.

Using a terminal emulator, you download this code to the printer:

```
^XA
^DFB:FILE1.ZPL
^F0100,100^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,
^F0100,100^A0,100^FDTEST^FS
^XZ
```

^HG

Host Graphic

Description 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.

Format ^HGd:o.x

This table identifies the parameters for this format:

Parameters	Details
d = device location of object	<i>Accepted Values:</i> R:, E:, B:, and A: <i>Default Value:</i> search priority
o = object name	<i>Accepted Values:</i> 1 to 8 alphanumeric characters <i>Default Value:</i> if a name is not specified, UNKNOWN is used
x = extension	<i>Fixed Value:</i> .GRF

Comments For more information on uploading graphics, see [^HY on page 237](#).

^HH

Configuration Label Return

Description The ^HH command echoes printer configuration back to the host, using a terminal emulator.

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
U32.10.2 <-	FIRMWARE
U2.2.6.98.A	HARDWARE ID
CUSTOMIZED	CONFIGURATION
1024.....R:	RAM
8192.....B:	MEMORY CARD
0768.....E:	ONBOARD FLASH
NONE	FORMAT CONUERT
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

Host Identification

Description 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 objects.

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 objects

only options specific to printer are shown (cutter, options, et cetera.)

~HM

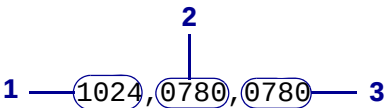
Host RAM Status

Description 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.

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

Host Query

Description The ~HQ command group causes the printer to send information back to the host.

Supported Devices

- Xi4
- ZM400/ZM600 with v53.17.1Z or later
- S4M with v53.17.1Z or later
- G-Series with v56.16.5 or later
or
G-Series with v61.16.5 or later

Format ~HQquery-type

This table identifies the parameters for this format:

Parameter	Details
query-type	For detailed examples of these parameters, see ~HQ Examples on page 223. Accepted Values: ES = requests the printer’s status - see Table 13 on page 221 and Table 14 on page 222 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 Value: 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 13 • 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 rror E	0	00000000	0	0	0	0	0	0	0	0
Error resent P	1	00000000	X	X	X	X	X	X	X	X
Printhead hermistor 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 etection Error	1	00000000	X	X	X	X	X	X	8	X
Bad rinthead Element	1	00000000	X	X	X	X	X	X	4	X
Motor ver Temperature	1	00000000	X	X	X	X	X	X	2	X
Printhead ver Temperature	1	00000000	X	X	X	X	X	X	1	X
Cutter ault F	1	00000000	X	X	X	X	X	X	X	8
Head pen O	1	00000000	X	X	X	X	X	X	X	4
Ribbon ut O	1	00000000	X	X	X	X	X	X	X	2
Media ut O	1	00000000	X	X	X	X	X	X	X	1
Clear Paper Path Failed ^a	1 ^a	00000000	X	X	X	X	8 ^a	X	X	X
Paper Feed Error ^a	1 ^a	00000000	X	X	X	X	4 ^a	X	X	X
Presenter Not Running ^a	1 ^a	00000000	X	X	X	X	2 ^a	X	X	X
Paper Jam during Retract ^a	1 ^a	00000000	X	X	X	X	1 ^a	X	X	X
Black Mark not Found ^a	1 ^a	00000000	X	X	X	8 ^a	X	X	X	X
Black Mark Calabrate Error ^a	1 ^a	00000000	X	X	X	4 ^a	X	X	X	X
Retract Function timed out ^a	1 ^a	00000000	X	X	X	2 ^a	X	X	X	X
Paused ^a	1 ^a	00000000	X	X	X	1 ^a	X	X	X	X

^a. This error flag is supported only on KR403 printers.

Table 14 • Warning 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 arning W	0	00000000	0	0	0	0	0	0	0	0
Warning resent P	1	00000000	X	X	X	X	X	X	X	X
Paper-near-end Sensor ^b	1 ^b	00000000	X	X	X	X	X	X	X	8 ^b
Replace rintheadP	1	00000000	X	X	X	X	X	X	X	4
Clean rintheadP	1	00000000	X	X	X	X	X	X	X	2
Need o alibrateC edia M	1	00000000	X	X	X	X	X	X	X	1
Sensor 1 (Paper before head) ^b	1 ^b	00000000	X	X	X	X	X	X	1 ^b	X
Sensor 2 (Black mark) ^b	1 ^b	00000000	X	X	X	X	X	X	2 ^b	X
Sensor 3 (Paper after head) ^b	1 ^b	00000000	X	X	X	X	X	X	4 ^b	X
Sensor 4 (loop ready) ^b	1 ^b	00000000	X	X	X	X	X	X	8 ^b	X
Sensor 5 (presenter) ^b	1 ^b	00000000	X	X	X	X	X	1 ^b	X	X
Sensor 6 (retract ready) ^b	1 ^b	00000000	X	X	X	X	X	2 ^b	X	X
Sensor 7 (in retract) ^b	1 ^b	00000000	X	X	X	X	X	4 ^b	X	X
Sensor 8 (at bin) ^b	1 ^b	00000000	X	X	X	X	X	8 ^b	X	X

b. This error flag is supported only on KR403 printers.

~HQ Examples

This section provides detail examples of all the available parameters.

➔ **Example 1** • This example shows how to request the printer’s status.

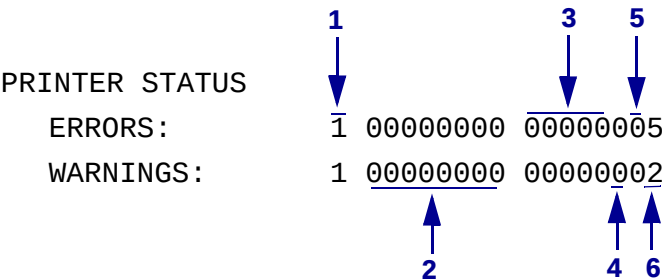
1. To request the printer’s status, type ~HQES
The printer responds with data similar to this:

```
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 2 • This example shows how the printer responds when the printer receives the ~HQES command:

1. 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 3 • This is an example of how to retrieve the hardware address of the internal wired print server.

1. 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 4 • 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 on page 273](#).

1. 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.

1. 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:

0, A, 0000, 0000, 0000

↑ ↑ ↑ ↑ ↑

1 2 3 4 5

1	Element failure
2	Manual (M) or automatic (A) range
3	First test element
4	Last test element
5	Failure count



Example 5 • 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 6 • 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 **1**
The printer responds with data similar to this:

```

MAINTENANCE ALERT MESSAGES 2
  CLEAN: PLEASE CLEAN PRINT HEAD
  REPLACE: PLEASE REPLACE PRINT HEAD

```



Example 7 • 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.

1. 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.

2. To change the units of measure to centimeters, type:

```
^XA^MA, , , , C
^XZ
```

The units of measure are set to centimeters.

3. 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 8 • 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.

1. 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  "
```

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 9 • This is an example of how to request the printer's Plug and Play string.

1. 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 10 • This is an example of how to retrieve the printer's serial number.

1. To get the printer's serial number, type ~HQSN
The printer responds with data similar to this:

```
SERIAL NUMBER
41A06440023
```



Example 11 • This is an example of how to retrieve the printer's USB product ID and BCD release version.

1. 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

Host Status Return

Description 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.



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
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	=	<i>buffer full</i> flag (1 = receive buffer full)
g	=	<i>communications diagnostic mode</i> flag (1 = diagnostic mode active)
h	=	<i>partial format</i> flag (1 = partial format in progress)
iii	=	unused (always 000)
j	=	<i>corrupt RAM</i> flag (1 = configuration data lost)
k	=	temperature range (1 = under temperature)
l	=	temperature range (1 = over temperature)

-
- a.** This string specifies the printer's baud rate, number of data bits, number of stopbits, 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.

The nine-digit binary number is read according to this table:

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 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 <i>(available only on certain printer models)</i> 1 001 = 38400 <i>(available only on certain printer models)</i> 1 010 = 57600 <i>(available only on certain printer models)</i> 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 ^b
n	=	unused
o	=	<i>head up</i> flag (1 = head in up position)
p	=	<i>ribbon out</i> flag (1 = ribbon out)
q	=	<i>thermal transfer mode</i> flag (1 = Thermal Transfer Mode selected)
r	=	Print Mode
		0 = Rewind
		1 = Peel-Off
		2 = Tear-Off
		3 = Cutter
		4 = Applicator
		5 = Delayed cut
		6 = Reserved ^c
		7 = Reserved ^c
		8 = Reserved ^c
		9 = RFID
		A = Kiosk Cut
s	=	print width mode
t	=	<i>label waiting</i> flag (1 = label waiting in Peel-off Mode)
uuuuuuuu	=	labels remaining in batch
v	=	<i>format while printing</i> flag (always 1)
www	=	number of graphic images stored in memory



Values 4 to 5 are only supported in firmware version V60.14.x, V50.14.x, V53.15.x, or later.

- b.** 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.
- c.** These values are only supported on the Xi4 and ZM400/ZM600 printers.

The eight-digit binary number is read according to this table:

mmm = m7 m6 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 <STX>xxxx, y<ETX><CR><LF>

xxxx	=	password
y	=	0 (static RAM not installed)
		1 (static RAM installed)

^HT

Host Linked Fonts List



This command is available only for printers with firmware version V60.14.x, V50.14.x, or later.

Description The ^HT command receives the complete list of font links over a communication port.



Example • 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

Return ZebraNet Alert Configuration

Description This command returns the table of configured ZebraNet Alert settings to the host.

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 355](#) for complete information on the individual parameter settings.

```
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

Host Verification

Description This command is used to return data from specified fields, along with an optional ASCII header, to the host computer. The command can be used with any field that has been assigned a number with the ^RT command or the ^FN and ^RF commands.

Format ^HV#, n, h

This table identifies the parameters for this format:

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. <i>Accepted Values:</i> 0 to 9999 <i>Default Value:</i> 0
n = number of bytes to be returned	<i>Accepted Values:</i> 1 to 256 <i>Default Value:</i> 64
h = header	Header to be returned with the data. <i>Accepted Values:</i> 0 to 3072 bytes <i>Default Value:</i> no header

^HW

Host Directory List

Description ^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.

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

This table identifies the parameters for this format:

Parameters	Details
d = location to retrieve object listing	<i>Accepted Values:</i> R:, E:, B:, A: and Z: <i>Default Value:</i> R:
o = object name	<i>Accepted Values:</i> 1 to 8 alphanumeric characters <i>Default Value:</i> asterisk (*). A question mark (?) can also be used.
x = extension	<i>Accepted Values:</i> any extension conforming to Zebra conventions <i>Default Value:</i> asterisk (*). A question mark (?) can also be used.

Parameters	Details
f = format  The f parameter is only supported in firmware version V60.16.0Z and V53.16.0Z or later.	<i>Accepted Values:</i> c = column format d = default format <i>Default Value:</i> d

➔ **Example** • Listed is an example of the ^HW command to retrieve from information R:

```

^XA
^HWR:*. *
^XZ

```

➔ **Example** • 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

Upload Graphics

Description The ^HY command is an extension of the ^HG command. ^HY is used to upload graphic objects from the printer in any supported format.

Format ^HYd:o.x

This table identifies the parameters for this format:

Parameters	Details
d = location of object	<i>Accepted Values:</i> R:, E:, B:, and A: <i>Default Value:</i> search priority
o = object name	<i>Accepted Values:</i> 1 to 8 alphanumeric characters <i>Default Value:</i> an object name must be specified
x = extension	<i>Accepted Values:</i> G = .GRF (raw bitmap format) P = .PNG (compressed bitmap format) <i>Default Value:</i> format of stored image

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

Display Description Information

Description 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.

Format ^HZb

This table identifies the parameters for this format:

Parameters	Details
b = display description to return	<i>Accepted Values:</i> - 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 <i>Default Value:</i> if the value is missing or invalid, the command is ignored

Format ^HZ0, d:o.x, l

This table identifies the parameters for this format:

Parameters	Details
d = location of stored object	<i>Accepted Values:</i> R:, E:, B:, and A: <i>Default Value:</i> R:
o = object name	<i>Accepted Values:</i> 1 to 8, or 1 to 16 alphanumeric characters based on parameter l. <i>Default Value:</i> 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

Parameters	Details
l = long filename support	<i>Accepted Values:</i> 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. <i>Default Value:</i> N



Example • This example shows the object data information for the object SAMPLE . GRF located on R : .

```
^XA
^HZ0,R: SAMPLE . GRF
^XZ
```

^ID

Object Delete

Description 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.

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

This table identifies the parameters for this format:

Parameters	Details
d = location of stored object	<i>Accepted Values:</i> R:, E:, B:, and A: <i>Default Value:</i> R:
o = object name	<i>Accepted Values:</i> any 1 to 8 character name <i>Default Value:</i> if a name is not specified, UNKNOWN is used
x = extension	<i>Accepted Values:</i> any extension conforming to Zebra conventions <i>Default Value:</i> .GRF



Example • To delete stored formats from DRAM:

```
^XA
^IDR:* .ZPL^FS
^XZ
```



Example • To delete formats and images named SAMPLE from DRAM, regardless of the extension:

```
^XA
^IDR:SAMPLE.*^FS
^XZ
```


→ **Example** • To delete the image SAMPLE1.GRF prior to storing SAMPLE2.GRF:

```
^XA
^F025,25^AD,18,10
^FDDelete^FS
^F025,45^AD,18,10
^FDthen Save^FS
^IDR:SAMPLE1.GRF^FS
^ISR:SAMPLE2.GRF^FS^XZ
```

→ **Example** • 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 *.ZOB, 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

Image Load

Description 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 ^F00,0.



Important • See [^IS](#) on page 245.

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

This table identifies the parameters for this format:

Parameters	Details
d = location of stored object	<i>Accepted Values:</i> R:, E:, B:, and A: <i>Default Value:</i> R:
o = object name	<i>Accepted Values:</i> 1 to 8 alphanumeric characters <i>Default Value:</i> if a name is not specified, UNKNOWN is used
x = extension	<i>Fixed Value:</i> .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 on page 245](#) command.

ZPL II CODE	GENERATED LABEL
^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	ZEBRA TECHNOLOGIES CORP ARTICLE# 900123 LOCATION LINE 12 DESCRIPTION ZEBRA THERMAL TRANSFER PRINTER REG. NO. 54321 WORK NUMBER Z58643 COMMENTS: Testing Stored Graphic Label Formats!!

^IM

Image Move

Description 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.

Format ^IMd:o.x

This table identifies the parameters for this format:

Parameters	Details
d = location of stored object	<i>Accepted Values:</i> R:, E:, B:, and A: <i>Default Value:</i> search priority
o = object name	<i>Accepted Values:</i> 1 to 8 alphanumeric characters <i>Default Value:</i> if a name is not specified, UNKNOWN is used
x = extension	<i>Fixed Value:</i> .GRF, .PNG

➔ **Example** • This example moves the image SAMPLE.GRF from DRAM and prints it in several locations in its original size.

```
^XA
^F0100,100^IMR:SAMPLE.GRF^FS
^F0100,200^IMR:SAMPLE.GRF^FS
^F0100,300^IMR:SAMPLE.GRF^FS
^F0100,400^IMR:SAMPLE.GRF^FS
^F0100,500^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

Image Save

Description 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.

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](#) on page 242.

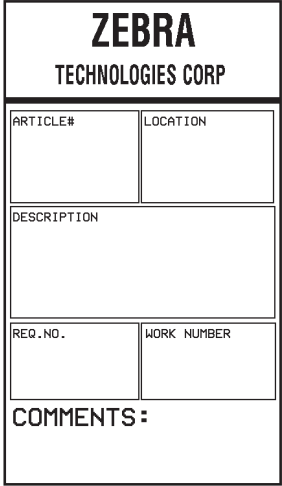
Format ^ISd:o.x,p

This table identifies the parameters for this format:

Parameters	Details
d = location of stored object	<i>Accepted Values:</i> R:, E:, B:, and A: <i>Default Value:</i> R:
o = object name	<i>Accepted Values:</i> 1 to 8 alphanumeric characters <i>Default Value:</i> if a name is not specified, UNKNOWN is used
x = extension	<i>Accepted Values:</i> .GRF or .PNG <i>Default Value:</i> .GRF
p = print image after storing	<i>Accepted Values:</i> N = no Y = yes <i>Default Value:</i> 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	GENERATED LABEL
<pre>^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</pre>	

~JA

Cancel All

Description The ~JA command cancels all format commands in the buffer. It also cancels any batches that are printing.

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

Initialize Flash Memory

Description The ^JB command is used to initialize various types of Flash memory available in the Zebra printers.

Format ^JBa

This table identifies the parameters for this format:

Parameters	Details
a = device to initialize	<i>Acceptable Values:</i> A = Option Flash memory B = Flash card (PCMCIA) E = internal Flash memory <i>Default Value:</i> 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

Description 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

Set Media Sensor Calibration

Description The ~JC command is used to force a label length measurement and adjust the media and ribbon sensor values.

Format ~JC

Comments In Continuous Mode, only the media and ribbon sensors are calibrated.

This command is ignored on the HC100™ printer.

~JD

Enable Communications Diagnostics

Description 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.

Format ~JD

~JE

Disable Diagnostics

Description The ~JE command cancels Diagnostic Mode and returns the printer to normal label printing.

Format ~JE

~JF

Set Battery Condition

Description 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

This table identifies the parameters for this format:

Parameters	Details
p = pause on low voltage	<i>Accepted Values:</i> Y (pause on low voltage) or N (do not pause) N is suggested when the printer is powered by the Car Battery Adapter. <i>Default Value:</i> Y

~JG

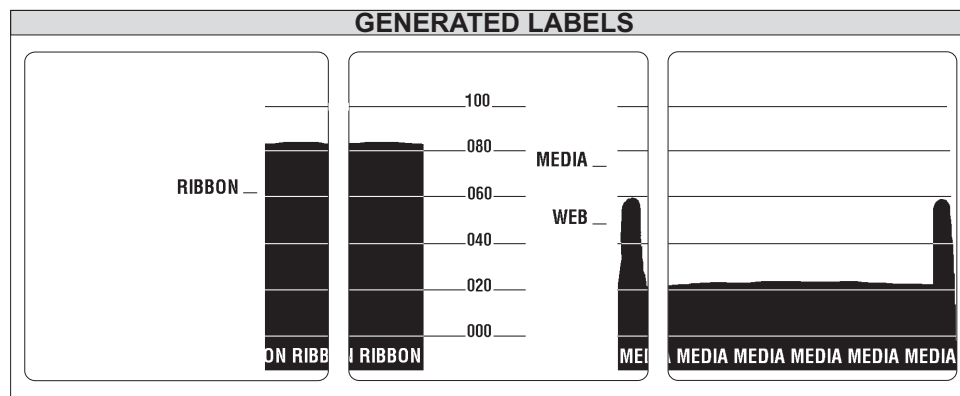
Graphing Sensor Calibration

Description The ~JG command prints a graph (media sensor profile) of the sensor values.

Format ~JG



Example • 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.



Early Warning Settings

Description The ^JH command configures the early warning messages that appear on the LCD.

Supported Devices

- *XiIII*, *XiIIIPlus*, *Xi4*
- *PAX3*, *PAX4*
- *ZM400*, *ZM600*
- *S4M*
- G-Series (“f” parameter only)

Format ^JHa,b,c,d,e,f,g,h,i,j

This table identifies the parameters for this format:

Parameter	Details
a = early warning media	This parameter is for <i>XiIIIPlus</i> , <i>Xi4</i> , <i>PAX3</i> , and <i>PAX4</i> printers only.
a = supplies warning (<i>Xi4</i> printers only)	<i>Accepted Values:</i> E = enable D = disable <i>Default Value:</i> D
b = labels per roll	This parameter is for <i>XiIIIPlus</i> , <i>PAX3</i> , and <i>PAX4</i> printers only. <i>Accepted Values:</i> 100 to 9999 <i>Default Value:</i> 900
c = media replaced	This parameter is for <i>XiIIIPlus</i> , <i>PAX3</i> , and <i>PAX4</i> printers only. <i>Accepted Values:</i> Y = yes N = no <i>Default Value:</i> N

Parameter	Details																				
d = ribbon length	This parameter is for <i>XiIIIPlus</i>, <i>PAX3</i>, and <i>PAX4</i> printers only. The accepted values for the <i>XiIII</i> series printers are 100M through 450M. <i>PAX</i> series printers allow for values up to 900M. <i>Accepted Values:</i> <table> <tr> <td>0 = 100M</td><td>10 = 600M</td></tr> <tr> <td>1 = 150M</td><td>11 = 650M</td></tr> <tr> <td>2 = 200M</td><td>12 = 700M</td></tr> <tr> <td>3 = 250M</td><td>13 = 750M</td></tr> <tr> <td>4 = 300M</td><td>14 = 800M</td></tr> <tr> <td>5 = 350M</td><td>15 = 850M</td></tr> <tr> <td>6 = 400M</td><td>16 = 900M</td></tr> <tr> <td>7 = 450M</td><td></td></tr> <tr> <td>8 = 500M</td><td></td></tr> <tr> <td>9 = 550M</td><td></td></tr> </table> <i>Default Value:</i> 7	0 = 100M	10 = 600M	1 = 150M	11 = 650M	2 = 200M	12 = 700M	3 = 250M	13 = 750M	4 = 300M	14 = 800M	5 = 350M	15 = 850M	6 = 400M	16 = 900M	7 = 450M		8 = 500M		9 = 550M	
0 = 100M	10 = 600M																				
1 = 150M	11 = 650M																				
2 = 200M	12 = 700M																				
3 = 250M	13 = 750M																				
4 = 300M	14 = 800M																				
5 = 350M	15 = 850M																				
6 = 400M	16 = 900M																				
7 = 450M																					
8 = 500M																					
9 = 550M																					
e = ribbon replaced	This parameter is for <i>XiIIIPlus</i>, <i>PAX3</i>, and <i>PAX4</i> printers only. <i>Accepted Values:</i> Y = yes N = no <i>Default Value:</i> N																				
f = early warning maintenance	This parameter is for <i>Xi4</i>, <i>PAX4</i>, <i>ZM400</i>, <i>ZM600</i>, and <i>S4M</i> printers only. <i>Accepted Values:</i> E = enabled D = disabled <i>Default Value:</i> D  Important • On G-Series printers, this parameter must be enabled for the ^MA driven system to work.																				

Parameter	Details																						
g = head cleaning interval	Accepted value exceptions: accepted values for XiIII printer are 100M through 450M; accepted values for 600 dpi XiIII printers are 100M through 150M; accepted values for PAX4 series printers can accept values up to 900M by increments of 50M; accepted values for ZM400/ZM600 and S4M printers are 0M through 450M. <i>Accepted Values:</i> <table> <tr> <td>0 = 100M</td><td>11 = 650M</td></tr> <tr> <td>1 = 150M (default for 96XiIII)</td><td>12 = 700M</td></tr> <tr> <td>2 = 200M</td><td>13 = 750M</td></tr> <tr> <td>3 = 250M</td><td>14 = 800M</td></tr> <tr> <td>4 = 300M</td><td>15 = 850M</td></tr> <tr> <td>5 = 350M</td><td>16 = 900M</td></tr> <tr> <td>6 = 400M</td><td></td></tr> <tr> <td>7 = 450M (default for all except 96XiIII)</td><td></td></tr> <tr> <td>8 = 500M</td><td></td></tr> <tr> <td>9 = 550M</td><td></td></tr> <tr> <td>10 = 600M</td><td></td></tr> </table> <i>Default Value:</i> see above	0 = 100M	11 = 650M	1 = 150M (default for 96XiIII)	12 = 700M	2 = 200M	13 = 750M	3 = 250M	14 = 800M	4 = 300M	15 = 850M	5 = 350M	16 = 900M	6 = 400M		7 = 450M (default for all except 96XiIII)		8 = 500M		9 = 550M		10 = 600M	
0 = 100M	11 = 650M																						
1 = 150M (default for 96XiIII)	12 = 700M																						
2 = 200M	13 = 750M																						
3 = 250M	14 = 800M																						
4 = 300M	15 = 850M																						
5 = 350M	16 = 900M																						
6 = 400M																							
7 = 450M (default for all except 96XiIII)																							
8 = 500M																							
9 = 550M																							
10 = 600M																							
h = head clean	<i>Accepted Values:</i> N = No Y = Yes <i>Default Value:</i> N																						
i = head life threshold	<i>Accepted Values:</i> 0 – 0 in or off 100-3500000 in <i>Default Value:</i> 1000000																						
j = head replaced	<i>Accepted Values:</i> N = no Y = yes <i>Default Value:</i> N																						

Comments To permanently save the changes to the ^JH command, send ^XA^JUS^XZ.

^JI

Start ZBI (Zebra BASIC Interpreter)



Identifies features that are available in printers with firmware version V60.16.2Z, V53.16.2Z, or later.

Description ^JI works much like the ~JI command. Both commands are sent to the printer to initialize the Zebra BASIC Interpreter.

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 ^JIId:o.x,b,c,d

This table identifies the parameters for this format:

Parameters	Details
d = location of program to run after initialization	<i>Acceptable Values:</i> R:, E:, B:, and A: <i>Default Value:</i> location must be specified
o = name of program to run after initialization	<i>Accepted Values:</i> any valid program name <i>Default Value:</i> name must be specified
x = extension of program to run after initialization	<i>Fixed Value:</i> .BAS, .BAE  .BAE is only supported in firmware version V60.16.0Z or later
b = console control	<i>Accepted Values:</i> Y = console on N = console off <i>Default Value:</i> Y
c = echoing control	<i>Accepted Values:</i> Y = echo on N = echo off <i>Default Value:</i> Y
d = memory allocation for ZBI *	<i>Accepted Values:</i> 20K to 1024K <i>Default Value:</i> 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

Start ZBI (Zebra BASIC Interpreter)



Identifies features that are available in printers with firmware version V60.16.2Z, V53.16.2Z, or later.

Description ~JI works much like the ^JI command. Both commands are sent to the printer to initialize the Zebra BASIC Interpreter.

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

Set Auxiliary Port

Description The ^JJ command allows you to control an online verifier or applicator device.

Format ^JJa,b,c,d,e,f

This table identifies the parameters for this format:

Parameters	Details
a = operational mode for auxiliary port	<i>Accepted Values:</i> 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. <i>Default Value:</i> 0
b = application mode	<i>Accepted Values:</i> 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. <i>Default Value:</i> 0

Parameters	Details
c = application mode start signal print	<i>Accepted Values:</i> - 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. <i>Default Value:</i> 0
d = application label error mode	<i>Accepted Values:</i> - e = error mode—the printer asserts the <i>Service Required</i> 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. <i>Default Value:</i> f
e = reprint mode	<i>Accepted Values:</i> - 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. <i>Default Value:</i> d
f = ribbon low mode	<i>Accepted Values:</i> - e = <i>enabled</i> – printer warning issued when ribbon low. - d = <i>disabled</i> – printer warning not issued when ribbon low. <i>Default Value:</i> e

~JL

Set Label Length

Description 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.

Format ~JL

^JM

Set Dots per Millimeter

Description 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 (^FO) placement on the label (see example below).

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

This table identifies the parameters for this format:

Parameters	Details
n = set dots per millimeter	<i>Accepted Values:</i> 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 <i>Default Value:</i> A



Example • This example of the affects of alternating the dots per millimeter:

ZPL II CODE	GENERATED LABEL
<pre>^XA ^JMA^FS ^FO100,100 ^B2N,50,Y,N,N ^FD1234567890^FS ^XZ</pre>	
<pre>^XA ^JMB^FS ^FO100,100 ^B2N,50,Y,N,N ^FD1234567890^FS ^XZ</pre>	

Comments If ^JMB is used, the UPS MaxiCode bar code becomes out of specification.

~JN

Head Test Fatal

Description The ~JN command turns on the head test option. When activated, ~JN causes the printer to halt when a head test failure is encountered.

Once an error is encountered the printer remains in error mode until the head test is turned off (~JO) 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 ~JO command is not received by the printer.

~JO

Head Test Non-Fatal

Description 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.

~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

Pause and Cancel Format

Description The ~JP command clears the format currently being processed and places the printer into Pause Mode.

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

Terminate Zebra BASIC Interpreter



Identifies features that are available in printers with firmware version V60.16.2Z, V53.16.2Z, or later.

Description The ~JQ command is used when Zebra BASIC Interpreter is active. Sending ~JQ to the printer terminates the ZBI session.

Format ~JQ

Comments Entering ZPL at the command prompt also terminates a ZBI session.

~JR

Power On Reset

Description 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.

Format ~JR

^JS

Sensor Select

Format ^JSa



Note • This command is ignored on Zebra ZM400/ZM600™ printers. This command is only for use with the Z series and S4M printers (with the exception of the ZM400/ZM600).

This table identifies the parameters for this format:

Parameters	Details
a = sensor selection	<i>Accepted Values:</i> A = auto select R = reflective sensor T = transmissive sensor <i>Default Value:</i> Z series = A and the S4M = R

~JS

Change Backfeed Sequence

Description The ~JS command is used to control the backfeed sequence. This command can be used on printers with or without built-in cutters.

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

This table identifies the parameters for this format:

Parameters	Details
b = backfeed order in relation to printing	<i>Accepted Values:</i> 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 O = 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 50 percent backfeed. <i>Default Value:</i> 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.

This command is ignored on the HC100™ printer.

^JT

Head Test Interval

Description 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.

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

This table identifies the parameters for this format:

Parameters	Details
#### = four-digit number of labels printed between head tests	<i>Accepted Values:</i> 0000 to 9999 If a value greater than 9999 is entered, it is ignored. <i>Default Value:</i> 0000 (off)
a = manually select range of elements to test	<i>Accepted Values:</i> N = no Y = yes <i>Initial Value at Power-up:</i> N
b = first element to check when parameter a is Y	<i>Accepted Values:</i> 0 to 9999 <i>Initial Value at Power-up:</i> 0
c = last element to check when parameter a is Y	<i>Accepted Values:</i> 0 to 9999 <i>Initial Value at Power-up:</i> 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

Configuration Update

Description The ^JU command sets the active configuration for the printer.

Format ^JUa

This table identifies the parameters for this format:

Parameters	Details
a = active configuration	<i>Accepted Values:</i> F = reload factory settings N = reload factory network settings These values are lost at power-off if not saved with ^JUS. R = recall last saved settings S = save current settings These values are used at power-on. <i>Default Value:</i> a value must be specified

^JW

Set Ribbon Tension

Description ^JW sets the ribbon tension for the printer it is sent to.

Format ^JWt

This table identifies the parameters for this format:

Parameters	Details
t = tension	<i>Accepted Values:</i> L = low M = medium H = high <i>Default Value:</i> a value must be specified

Comments ^JW is used only for *PAX* series printers.

~JX

Cancel Current Partially Input Format

Description 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.

Format ~JX

^JZ

Reprint After Error

Description 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.

This command remains active until another ^JZ command is sent to the printer or the printer is turned off.

Format ^JZa

This table identifies the parameters for this format:

Parameters	Details
a = reprint after error	<i>Accepted Values:</i> N = no Y = yes <i>Initial Value at Power-up: Y</i>

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

Kill Battery (Battery Discharge Mode)

Description To maintain performance of the rechargeable battery in the portable printers, the battery must be fully discharged and recharged regularly. The ~KB command places the printer in battery discharge mode. This allows the battery to be drained without actually printing.

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

Select Date and Time Format (for Real Time Clock)

Description 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.

Format ^KDa

This table identifies the parameters for this format:

Parameters	Details
a = value of date and time format	<i>Accepted Values:</i> 0 = normal, displays <i>Version Number</i> 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) <i>Default Value:</i> 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 977](#).

^KL

Define Language

Description The ^KL command selects the language displayed on the control panel.

Format ^KL*a*

This table identifies the parameters for this format:

Parameters	Details
<i>a</i> = language	Accepted 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 ^a 15 = Simplified Chinese ^a 16 = Traditional Chinese ^a 17 = Russian ^a 18 = Polish ^a Default Value: 1

a. These values are only supported on the Xi4 and ZM400/ZM600 printers.

^KN

Define Printer Name

Description 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.

Format ^KNa, b

This table identifies the parameters for this format:

Parameters	Details
a = printer name	<i>Accepted Values:</i> up to 16 alphanumeric characters <i>Default Value:</i> 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	<i>Accepted Values:</i> up to 35 alphanumeric characters <i>Default Value:</i> 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. Note • 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 an description:
This shows how a configuration looks before using this command and after using this command:

```
^XA
^KNZebra1,desk_printer
^XZ
```

Before using this command:

After using this command:

PRINTER CONFIGURATION	
Zebra Technologies ZTC 105SL-200dpi	
+18.....	DARKNESS
-016.....	TEAR OFF
TEAR OFF.....	PRINT MODE
NON-CONTINUOUS.....	MEDIA TYPE
WEB.....	SENSOR TYPE
THERMAL-TRANS.....	PRINT METHOD
101 4/8 MM.....	PRINT WIDTH
1233.....	LABEL LENGTH
7.0IN 177MM.....	MAXIMUM LENGTH
PARALLEL.....	PARALLEL COMM.
RS232.....	SERIAL COMM.
9600.....	BAUD
8 BITS.....	DATA BITS
NONE.....	PARITY
XON/XOFF.....	HOST HANDSHAKE
NONE.....	PROTOCOL
000.....	NETWORK
NORMAL MODE.....	COMM
<~> ?E~	
<^>	

PRINTER CONFIGURATION	
Zebra Technologies ZTC 105SL-200dpi Zebra1 desk_printer	
+18.....	DARKNESS
-016.....	TEAR OFF
TEAR OFF.....	PRINT MODE
NON-CONTINUOUS.....	MEDIA TYPE
WEB.....	SENSOR TYPE
THERMAL-TRANS.....	PRINT METHOD
101 4/8 MM.....	PRINT WIDTH
1233.....	LABEL LENGTH
7.0IN 177MM.....	MAXIMUM LENGTH
PARALLEL.....	PARALLEL COMM.
RS232.....	SERIAL COMM.
9600.....	BAUD
8 BITS.....	DATA BITS
NONE.....	PARITY
XON/XOFF.....	HOST HANDSHAKE
NONE.....	PROTOCOL
000.....	NETWORK
NORMAL MODE.....	COMM
<~> ?E~	
<^>	

^KP

Define Password

Description The ^KP command is used to define the password that must be entered to access the control panel switches and LCD Setup Mode.

Format ^KP*a*, *b*

This table identifies the parameters for this format:

Parameters	Details
<i>a</i> = mandatory four-digit password	<i>Accepted Values:</i> any four-digit numeric sequence <i>Default Value:</i> 1234
<i>b</i> = password level	<i>Accepted Values:</i> 1, 2, 3, 4 <i>Default Value:</i> 3



Example • This is an example of how to set a new control panel password:

^XA

^KP5678

^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

Kiosk Values

Description The ^KV command sets several parameters that affect the printers operation when ^MM is set to K - Kiosk mode

Supported Devices

- KR403

Format ^KV*a, b, c, d, e*

This table identifies the parameters for this format:

Parameters	Details
a = kiosk cut amount	<i>Accepted Values:</i> 0 = normal cut 10-60 = partial cut, value = mm of media left uncut <i>Default Value:</i> 0 This parameter is ignored if it is missing or invalid. The current value of the parameter remains unchanged.
b = kiosk cut margin	<i>Accepted Values:</i> 2 - 9 = mm of distance <i>Default Value:</i> 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	<i>Accepted Values:</i> 0 = Eject page when new page is printed 1 = Retract page when new page is printed 2 = Do nothing when new page is printed <i>Default Value:</i> 0 This parameter is ignored if it is missing or invalid. The current value of the parameter remains unchanged.
d = kiosk present timeout	<i>Accepted Values:</i> 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. <i>Default Value:</i> 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	<i>Accepted Values:</i> 0 = paper is fed straight through the presenter 3-1023 = loop length in mm. <i>Default Value: 400</i> 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.

^LF

List Font Links



This command is available only for printers with firmware version V60.14.x, V50.14.x, or later.

Description The ^LF command prints out a list of the linked fonts.



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	GENERATED LABEL
^XA ^LF ^XZ	LIST OF FONT LINKS E:SWISS721.TTF E:ANMDJ.TTF E:MSGOTHIC.TTF

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

Label Home

Description The ^LH command sets the label home position.

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

This table identifies the parameters for this format:

Parameters	Details
x = x-axis position (in dots)	<i>Accepted Values:</i> 0 to 32000 <i>Initial Value at Power-up:</i> 0 or last permanently saved value
y = y-axis position (in dots)	<i>Accepted Values:</i> 0 to 32000 <i>Initial Value at Power-up:</i> 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

Label Length

Description 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).

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

This table identifies the parameters for this format:

Parameters	Details
y = y-axis position (in dots)	<i>Accepted Values:</i> 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. <i>Default Value:</i> typically set through the LCD (if applicable), or to the maximum label length capability of the printer.

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

Label Reverse Print

Description 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.

Using the ^LR is identical to placing an ^FR command in all current and subsequent fields.

Format ^LRa

This table identifies the parameters for this format:

Parameters	Details
a = reverse print all fields	<i>Accepted Values:</i> N = no Y = yes <i>Initial Value at Power-up:</i> 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	GENERATED LABEL
<pre>^XA^LRY ^FO100,50 ^GB195,203,195^FS ^FO180,110^CFG ^FDLABEL^FS ^FO130,170 ^FDREVERSE^FS ^XZ</pre>	

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

Label Shift

Description 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.

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.

This table identifies the parameters for this format:

Parameters	Details
a = shift left value (in dots)	<i>Accepted Values: -9999 to 9999</i> <i>Initial Value at Power-up: 0</i>

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

Label Top

Description 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.

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

This table identifies the parameters for this format:

Parameters	Details
x = label top (in dot rows)	<i>Accepted Values:</i> HC100: 0 to 120 XiIIIPlus 600dpi: -240 to 240 All other Zebra printers: -120 to 120 <i>Default Value:</i> 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

Set Maintenance Alerts



This command is available only for printers with firmware version V60.15.x, V50.15.x, or later.

Description 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.

Supported Devices

- Xi4
- ZM400/ZM600
- S4M with v53.15.5Z or later
- G-Series



Important • ^MA settings do not impact or effect the functionality of the Xi4 Supplies Warning system.

Format ^MAtype,print,printlabel_threshold,frequency,units

This table identifies the parameters for this format:

Parameters	Details
type = type of alert	<i>Accepted Values:</i> R = head replacement C = head cleaning <i>Default Value:</i> 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	<i>Accepted Values:</i> Y = print a label N = do not print label <i>Default Value:</i> N
printlabel_threshold = distance where the first alert occurs	<i>Accepted Values:</i> R = head replacement (unit of measurement for head is km with a range of 0 to 150 km) C = clean head (unit of measurement is 1 meter = 39.37 inches with a range of 0 to 2000 meters. The range for G-Series printers is 100 to 2000 meters. 0 = off (when set to 0, the selected alert is disabled; otherwise it is enabled. <i>Default Value:</i> R = 50 km (1,968,500 inches) and C = 0 (off).

Parameters	Details
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. <i>Default Value:</i> 0 (print on power-up).
units = odometer and printhead maintenance commands	The units parameter reports units of the odometer and printhead maintenance commands, as follows: ~HQOD, ~HQPH, ~WQOD, ~WQPH. <i>Accepted Values:</i> C = centimeters (displays as: cm) I = inches (displays as: ") M = meters (displays as: M) <i>Default Value:</i> 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.

1. 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 clean.

2. For this example, the message on the label looks like this:

PLEASE CLEAN PRINT HEAD

For details resetting the units of measure, see the ~HQ examples on [page 223](#).

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

Map Clear

Description 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.

Format ^MCa



Important • To produce a label template, ^MC must be used with ^FV.

This table identifies the parameters for this format:

Parameters	Details
a = map clear	<i>Accepted Values:</i> Y (clear bitmap) or N (do not clear bitmap) <i>Initial Value at Power-up:</i> 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

Media Darkness

Description The ^MD command adjusts the darkness relative to the current darkness setting.

Format ^MDa

This table identifies the parameters for this format:

Parameters	Details
a = media darkness level	<i>Accepted Values: -30 to 30, depending on current value</i> <i>Initial Value at Power-up: 0</i> 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 label formats.



Important • The darkness setting range for the XiIIIPlus and Xi4 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 and Xi4 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

Media Feed

Description The ^MF command dictates what happens to the media at power-up and at head-close after the error clears.

Format ^MFp, h

This table identifies the parameters for this format:

Parameters	Details
p = feed action at power-up	<i>Accepted Values:</i> F = feed to the first web after sensor C = (see ~JC on page 250 definition) L = (see ~JL on page 263 definition) N = no media feed S = short calibration ^a <i>Default Value: C</i>
h = feed action after closing printhead	<i>Accepted Values:</i> F = feed to the first web after sensor C = (see ~JC on page 250 definition) L = (see ~JL on page 263 definition) N = no media feed S = short calibration ^a <i>Default Value: C</i>

^a. These values are supported only on Xi4, XiIIIPlus, PAX, ZM400/ZM600, 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

Set Maintenance Information Message



This command is available only for printers with firmware version V60.15.x, V50.15.x, or later.

Description 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.

Supported Devices

- Xi4
- ZM400/ZM600
- S4M with v53.15.5Z or later
- G-Series

Format ^MItype,message

This table identifies the parameters for this format:

Parameters	Details
type = identifies the type of alert	<i>Accepted Values:</i> R = head replacement C = head cleaning <i>Default Value:</i> 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. <i>Default Value:</i> 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:

PRINT HEAD NEEDS REPLACEMENT - CALL
EXT 1000

^ML

Maximum Label Length

Description The ^ML command lets you adjust the maximum label length.

Format ^MLa

This table identifies the parameters for this format:

Parameters	Details
a = maximum label length (in dot rows)	<i>Accepted Values:</i> 0 to maximum length of label <i>Default Value:</i> 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

Print Mode

Description The ^MM command determines the action the printer takes after a label or group of labels has printed.

Format ^MMa, b

This table identifies the parameters for this format:

Parameters	Details
a = desired mode	<i>Accepted Values:</i> - 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 <i>Default Value:</i> The values available for parameter a depend on the printer being used and whether it supports the option. For RFID printers: - A = R110PAX4 print engines - F = other RFID printers
b = prepeel select	<i>Accepted Values:</i> - N = no - Y = yes <i>Default Value:</i> N The command is ignored if parameters are missing or invalid. The current value of the command remains unchanged.

a. This value is not supported on the KR403 printer.

b. This value is supported only on the ZM400 and ZM600 printers.

c. This value is supported only 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 backing 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.
- 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 ^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

Media Tracking

Description The ^MN command relays to the printer what type of media is being used (continuous or non-continuous) for purposes of 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 ^MMV 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

This table identifies the parameters for this format:

Parameters	Details
a = media being used	<i>Accepted Values:</i> N = continuous media Y = non-continuous media web sensing ^{a, b} W = non-continuous media web sensing ^{a, b} M = non-continuous media mark sensing A = auto-detects the type of media during calibration ^{a, c} V = continuous media, variable length ^d <i>Default Value:</i> a value must be entered or the command is ignored

- a.** Provides the same result.
b. This value is not supported on the KR403 printer.
c. This parameter is supported only on G-series printers.
d. This parameter is supported only on the KR403 printer.

Comments This command is ignored on the HC100™ printer.

^MP

Mode Protection

Description 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.

Because this command has only one parameter, each mode must be disabled with an individual ^MP command.

Format ^MPa

This table identifies the parameters for this format:

Parameters	Details
a = mode to protect	<i>Accepted Values:</i> - 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 <i>Default Value:</i> 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 110X1111Plus-300dpi ZBR1462977	
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
1820.....	LABEL LENGTH
39.0IN 988MM.....	MAXIMUM LENGTH
MEDIA DISABLED.....	EARLY WARNING
PRINT 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
<=> ZEH.....	CONTROL PREFIX
<=> SEH.....	FORMAT PREFIX
<=> ZCH.....	DELIMITER CHGR
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 RED S.
079.....	MEDIA LED
039.....	RIBBON LED
039.....	MARK LED
+10.....	LCD ADJUST
DPCSWFXM.....	MODES ENABLED
.....	MODES DISABLED
1248 12/7MM FULL.....	RESOLUTION
V60.15.12P07 C.....	FIRMWARE
V30 79089 57.....	HARDWARE ID
CUSTOMIZED.....	CONFIGURATION
NONE.....	COMPACT FLASH
11776K.....	RAM
2048K.....	ONBOARD FLASH
NONE.....	FORMAT CONVERT
005 DISPLAY.....	P32 INTERFACE
001 POWER SUPPLY.....	P35 INTERFACE
.....	TWINAX/CDAX ID
11/18/06.....	IDLE DISPLAY
16:10.....	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 CH.....	NONRESET CNTR
39850 CH.....	RESET CNTR1
39850 CH.....	RESET CNTR2
4207 LABELS.....	NONRESET CNTR
4207 LABELS.....	RESET CNTR1
4207 LABELS.....	RESET CNTR2
06 36850.0 5 NYS07015.41008.07-VH1..	
2006-06-14 16:30:23 TIME STAMP	

FIRMWARE IN THIS PRINTER IS COPYRIGHTED

^MT

Media Type

Description The ^MT command selects the type of media being used in the printer.

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

This table identifies the parameters for this format:

Parameters	Details
a = media type used	<i>Accepted Values:</i> T = thermal transfer media D = direct thermal media <i>Default Value:</i> a value must be entered or the command is ignored

Comments This command is ignored on the HC100™ printer.

^MU

Set Units of Measurement

Description 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.

^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

This table identifies the parameters for this format:

Parameters	Details
a = units	<i>Accepted Values:</i> D = dots I = inches M = millimeters <i>Default Value:</i> D
b = format base in dots per inch	<i>Accepted Values:</i> 150, 200, 300 <i>Default Value:</i> a value must be entered or the command is ignored
c = desired dots-per-inch conversion	<i>Accepted Values:</i> 300, 600 <i>Default Value:</i> 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^F0100,100^GB1024,128,128^FS

Field based on millimeters:

^MUm^F012.5,12.5^GB128,16,16^FS

Field based on inches:

^MUi^F0.493,.493^GB5.044,.631,.631^FS



Example • This is an example of Converting dpi Values.

Convert a 150 dpi format to a 300 dpi format with a base in dots:

`^MUd, 150, 300`

Convert a 150 dpi format to a 600 dpi format with a base in dots:

`^MUd, 150, 600`

Convert a 200 dpi format to a 600 dpi format with a base in dots:

`^MUd, 200, 600`

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

Modify Head Cold Warning

Description The ^MW command allows you to set the head cold warning indicator based on the operating environment.

Format ^MWa

This table identifies the parameters for this format:

Parameters	Details
a = enable head cold warning	<i>Accepted Values:</i> Y = enable head cold warning N = disable head cold warning



Important • When a parameter is **not** given, the instruction is *ignored*.

^NC

Select the Primary Network Device

Description The ^NC command selects the wired or wireless print server as the primary network device.

Supported Devices

- Xi4
- ZM400/ZM600
- RZ400/RZ600

The Xi4, ZM400, ZM600, RZ400, and 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. [Table 15](#) outlines priorities and identifies which device becomes the active print server when multiple print servers are installed.

Table 15 • 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 ^a	
Wired	X	X	X	Internal
		X	X	External
			X	Wireless
Wireless	X	X	X	Wireless
	X	X		Internal
		X		External

a. A wireless option board must have an active radio that can properly associate to an access point.

Format ^NCa

This table identifies the parameters for this format:

Parameters	Details
a = primary network device	<i>Accepted Values:</i> 1 = wired primary 2 = wireless primary <i>Default Value: 1</i> must be an accepted value or it is ignored

~NC

Network Connect

Description The ~NC command is used to connect a particular printer to a network by calling up the printer's network ID number.

Format ~NC###

This table identifies the parameters for this format:

Parameters	Details
### = network ID number assigned (must be a three-digit entry)	<i>Accepted Values: 001 to 999</i> <i>Default Value: 000 (none)</i>

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

Change Network Settings

Description The ^ND command changes the network settings on supported printers.

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.

Supported Devices

- Xi4 with firmware V53.17.1Z or later
- RZ400/RZ600 with firmware R53.15.xZ or later
- ZM400/ZM600 with firmware V53.15.xZ or later
- G-Series

Format ^NDa,b,c,d,e,f,g,h,i,j

This table identifies the parameters for this format:

Parameters	Details
a = the device that is being modified	<i>Accepted Values:</i> 1 = external wired 2 = internal wired 3 = wireless
b = IP resolution	<i>Accepted Values:</i> 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 <i>Default Value: A</i>
c = IP address	<i>Accepted Values:</i> Any properly formatted IP address in the xxx.xxx.xxx.xxx format.
d = subnet mask	<i>Accepted Values:</i> Any properly formatted subnet mask in the xxx.xxx.xxx.xxx format.
e = default gateway	<i>Accepted Values:</i> Any properly formatted gateway in the xxx.xxx.xxx.xxx format.
f = WINS server address	<i>Accepted Values:</i> Any properly formatted WINS server in the xxx.xxx.xxx.xxx format.

Parameters	Details
g = connection timeout checking	<i>Accepted Values:</i> Y = yes N = no <i>Default Value:</i> Y
h = timeout value	Time, in seconds, before the connection times out. <i>Accepted Values:</i> 0 through 9999 <i>Default Value:</i> 300
i = ARP broadcast interval	Time, in minutes, that the broadcast is sent to update the device's ARP cache. <i>Accepted Values:</i> 0 through 30 <i>Default Value:</i> 0 (no ARP sent)
j = base raw port number	The port number that the printer should use for its RAW data. <i>Accepted Values:</i> 1 through 65535 <i>Default Value:</i> 9100

^NI

Network ID Number

Description 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.

Format ^NI###

This table identifies the parameters for this format:

Parameters	Details
### = network ID number assigned (must be a three-digit entry)	<i>Accepted Values: 001 to 999</i> <i>Default Value: 000 (none)</i>

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.

~NR

Set All Network Printers Transparent

Description The ~NR command sets all printers in the network to be transparent, regardless of ID or current mode.

Format ~NR

Comments The commands ~NC, ^NI, ~NR, and ~NT are used only with RS-485 printer communications.

^NS

Change Wired Networking Settings

Description Use this command to change the wired print server network settings.

Format ^NSa, b, c, d, e, f, g, h, i

The following table identifies the parameters for this format.

Parameters	Details
a = IP resolution	<i>Accepted Values:</i> A = ALL B = BOOTP C = DHCP AND BOOTP D = DHCP G = GLEANING ONLY R = RARP P = PERMANENT <i>Default Value:</i> A_ Use of GLEANING ONLY is not recommended when the Wireless Print Server or Wireless Plus Print Server is installed.
b = IP address	<i>Accepted Values:</i> Any properly formatted IP address in the xxx.xxx.xxx.xxx format.
c = subnet mask	<i>Accepted Values:</i> Any properly formatted subnet mask in the xxx.xxx.xxx.xxx format.
d = default gateway	<i>Accepted Values:</i> Any properly formatted gateway in the xxx.xxx.xxx.xxx format.
e = WINS server address	<i>Accepted Values:</i> Any properly formatted WINS server in the xxx.xxx.xxx.xxx format.
f = connection timeout checking	<i>Accepted Values:</i> Y = Yes N = No <i>Default Value:</i> Y
g = timeout value	Time, in seconds, before the connection times out. <i>Accepted Values:</i> 0 through 9999 <i>Default Value:</i> 300
h = ARP broadcast interval	Time, in minutes, that the broadcast is sent to update the device's ARP cache. <i>Accepted Values:</i> 0 through 30 <i>Default Value:</i> 0 (no ARP sent)

Parameters	Details
<code>i</code> = base raw port number	The port number that the printer should use for its RAW data. <i>Accepted Values:</i> 1 through 65535 <i>Default Value:</i> 9100

Comments For the Xi4, ZM400, and ZM600 printers, Zebra recommends that you use the ^ND command instead of the ^NS command.



Example •

```

^XA
^NSa,192.168.0.1,255.255.255.0,192.168.0.2
^XZ

```

~NT

Set Currently Connected Printer Transparent

Description The ~NT command sets the currently connected network printer to be 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.

^PA

Advanced Text Properties



This command is available only for printers with firmware version V60.14.x, V50.14.x, or later.

Description The ^PA command is used to configure advanced text layout features.

Format ^PAa, b, c, d

This table identifies the parameters for this format:

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. <i>Accepted Values:</i> 0 = off (space as default glyph) 1 = on (default glyph of font is used, often a hollow box, but depends on the font.) <i>Default Value:</i> 0
b = bidirectional text layout	This determines whether the bidirectional text layout is turned on or off. <i>Accepted Values:</i> 0 = off 1 = on <i>Default Value:</i> 0
c = character shaping	This determines whether character shaping is turned on or off. <i>Accepted Values:</i> 0 = off 1 = on <i>Default Value:</i> 0
d = OpenType table support	This determines whether the OpenType support is turned on or off. <i>Accepted Values:</i> 0 = off 1 = on <i>Default Value:</i> 0

^PF

Slew Given Number of Dot Rows

Description 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.

Format ^PF#

This table identifies the parameters for this format:

Parameters	Details
# = number of dots rows to slew	<i>Accepted Values:</i> 0 to 32000 <i>Default Value:</i> a value must be entered or the command is ignored

^PH ~PH

Slew to Home Position

Description The ^PH or ~PH command causes the printer to feed one blank label.

The ~PH command feeds one label after the format currently being printed is done or when the printer is placed in pause.

The ^PH command feeds one blank label after the current format prints.

Format ^PH or ~PH

~PL

Present Length Addition

Description The ~PL command 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 when a ^PN is executed, then, is 50mm + ~PL value + ^PN value.

Supported Devices

- KR403

Format ^PNa

This table identifies the parameters for this format:

Parameters	Details
a = additional eject length	<i>Accepted Values:</i> 0-255 = additional mm of media to eject <i>Default Value:</i> 40 The command is ignored if parameters are missing or invalid.

^PM

Printing Mirror Image of Label

Description The ^PM command prints the entire printable area of the label as a mirror image. This command flips the image from left to right.

Format ^PMa

This table identifies the parameters for this format:

Parameters	Details
a = print mirror image of entire label	<i>Accepted Values:</i> N = no Y = yes <i>Default Value:</i> N



Example • This is an example of printing a mirror image on a label:

ZPL II CODE	GENERATED LABEL
<pre>^XA^PMY ^FO100,100 ^CFG ^FDMIRROR^FS ^FO100,160 ^FDIMAGE^FS ^XZ</pre>	

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.

^PN

Present Now

Description 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 50mm + ~PL value + ^PN value. (See [~PL on page 320](#)).

Supported Devices

- KR403

Format ^PNa

This table identifies the parameters for this format:

Parameters	Details
a = media eject length	<i>Accepted Values:</i> 0-255 = additional mm of media to eject <i>Default Value:</i> none The command is ignored if parameters are missing or invalid.

^PO

Print Orientation

Description The ^PO 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.

Format ^POa

This table identifies the parameters for this format:

Parameters	Details
a = invert label 180 degrees	Accepted Values: N = normal I = invert Default Value: N



Example • This is an example of printing a label at 180 degrees:

ZPL II CODE	GENERATED LABEL
<pre>^XA^CFD ^POI ^LH330,10 ^FO50,50 ^FDZEBRA TECHNOLOGIES^FS ^FO50,75 ^FDVernon Hills, IL^FS ^XZ</pre>	

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 ^PO commands are issued in the same label format, only the last command sent to the printer is used.

Once the ^PO command is sent, the setting is retained until another ^PO 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

Programmable Pause

Description The ~PP command stops printing after the current label is complete (if one is printing) and places the printer in Pause Mode.

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

Print Quantity

Description 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.

Format ^PQq,p,r,o

This table identifies the parameters for this format:

Parameters	Details
q = total quantity of labels to print	<i>Accepted Value:</i> 1 to 99,999,999 <i>Default Value:</i> 1
p = pause and cut value (labels between pauses)	<i>Accepted Value:</i> 1 to 99,999,999 <i>Default Value:</i> 0 (no pause)
r = replicates of each serial number	<i>Accepted Value:</i> 0 to 99,999,999 replicates <i>Default Value:</i> 0 (no replicates)
o = override pause count	<i>Accepted Values:</i> N = no Y = yes <i>Default Value:</i> N

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

Print Rate

Description The ^PR command determines the media and slew speed (feeding a blank label) during printing.

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.



Important • Some models go to default print speed when power is turned off.

Format ^PRp, s, b

This table identifies the parameters for this format:

Parameters	Details
p = print speed	<i>Accepted Values:</i> 1 = 25.4 mm/sec. (1 inch/sec.)^a 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.) 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^b 14 = 14 in/sec^b <i>Default Value: A</i>

a. This value is supported only on the 110Xi4-600dpi and 110XiIIIPlus-600dpi printers.

b. This value is supported only on the Xi4 printer.

Parameters	Details
s = slew speed	<i>Accepted Values:</i> 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.) 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^b 14 = 14 in/sec^b <i>Default Value: D</i>
b = backfeed speed	<i>Accepted Values:</i> 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.) 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^b 14 = 14 in/sec^b <i>Default Value: A</i>

a. This value is supported only on the 110Xi4-600dpi and 110XiIIIPlus-600dpi printers.

b. This value is supported only on the Xi4 printer.

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.

~PR

Applicator Reprint

Description If the ~PR command is enabled (see [^JJ on page 261](#)), the last label printed reprints, similar to the applicator asserting the Reprint signal on the applicator port.

Supported Devices

- *XiIIIPlus*
- Xi4 with firmware V53.17.1Z or later
- *PAX2, PAX4*
- S4M
- ZM400/ZM600

Format ~PR

Comments Pressing PREVIOUS on the control panel also causes the last label to reprint.

~PS

Print Start

Description 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.

Format ~PS

^PW

Print Width

Description The ^PW command allows you to set the print width.

Format ^PWa

This table identifies the parameters for this format:

Parameters	Details
a = label width (in dots)	<i>Accepted Values:</i> 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. <i>Default Value:</i> last permanently saved value

Comments This command is ignored on the HC100™ printer.

~R0

Reset Advanced Counter

Description The ~R0 command resets the advanced counters used by the printer to monitor label generation in inches, centimeters, and number of labels. These resettable counters are available and can be reset.

Format ~R0c

This table identifies the parameters for this format:

Parameters	Details
C = counter number	<i>Accepted Values:</i> 1 = reset counter 1 2 = reset counter 2 R = reset head replaced counter ^a R resets the head cleaned counter and the head replaced counter. C = reset head cleaned counter ^a <i>Default Value:</i> a value must be specified or the command is ignored

a. These values are only supported on Xi4, ZM400/ZM600, S4M, and G-Series printers.



Example • This example shows how the counter portion of the printer configuration label looks when counter 1 is reset by sending ~R01:

Before

```
296862 IN..... NONRESET CNTR
296862 IN..... RESET CNTR1
296862 IN..... RESET CNTR2
753289 CM..... NONRESET CNTR
753289 CM..... RESET CNTR1
753289 CM..... RESET CNTR2
92928 LABLS..... NONRESET CNTR
92928 LABLS..... RESET CNTR1
92928 LABLS..... RESET CNTR2
```

After

```
296876 IN..... NONRESET CNTR
0 IN..... RESET CNTR1
296876 IN..... RESET CNTR2
753323 CM..... NONRESET CNTR
0 CM..... RESET CNTR1
753323 CM..... RESET CNTR2
92930 LABLS..... NONRESET CNTR
0 LABLS..... RESET CNTR1
92930 LABLS..... RESET CNTR2
```

^SC

Set Serial Communications

Description The ^SC command allows you to change the serial communications parameters you are using.

Format ^SCa,b,c,d,e,f

This table identifies the parameters for this format:

Parameters	Details
a = baud rate	<i>Accepted Values:</i> 110^a; 300; 600; 1200; 2400; 4800; 9600; 14400; 19200; 28800; 38400; or 57600; 115200 <i>Default Value:</i> must be specified or the parameter is ignored
b = word length (in data bits)	<i>Accepted Values:</i> 7 or 8 <i>Default Value:</i> must be specified
c = parity	<i>Accepted Values:</i> N (none), E (even), or O (odd) <i>Default Value:</i> must be specified
d = stop bits	<i>Accepted Values:</i> 1 or 2 <i>Default Value:</i> must be specified
e = protocol mode	<i>Accepted Values:</i> X = XON/XOFF D = DTR/DSR R = RTS M = DTR/DSR XON/XOFF^b <i>Default Value:</i> must be specified
f = Zebra protocol	<i>Accepted Values:</i> A = ACK/NAK N = none Z = Zebra <i>Default Value:</i> must be specified

a. This value is not supported on Xi4, ZM400/ZM600, and S4M printers.

b. 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

Set Darkness

Description 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.

Format ~SD##

This table identifies the parameters for this format:

Parameters	Details
## = desired darkness setting (two-digit number)	<i>Accepted Values:</i> 00 to 30 <i>Default Value:</i> last permanently saved value



Important • The darkness setting range for the *XiIIIPlus* and *Xi4* 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* and *Xi4* Darkness Setting:

^MD8.3

~SD8.3

Comments The ^MD command value, if applicable, is added to the ~SD command.

^SE

Select Encoding Table

Description The ^SE command is used to select the desired ZPL or ZPL II encoding table.

Format ^SEd:o.x

This table identifies the parameters for this format:

Parameters	Details
d = location of encoding table	<i>Accepted Values:</i> R:, E:, B:, and A: <i>Default Value:</i> R:
o = name of encoding table	<i>Accepted Value:</i> 1 to 8 alphanumeric characters <i>Default Value:</i> a value must be specified
x = extension	<i>Fixed Value:</i> .DAT

The encoding tables are provided with the font card or downloaded in flash with the font. The table appears as XXXXXXXX.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

Serialization Field (with a Standard ^FD String)

Description The ^SF command allows you to serialize a standard ^FD string. The maximum size of the mask and increment string is 3K combined.



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

This table identifies the parameters for this format:

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 fimrware 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 fimrware 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. <i>Mask String placeholders:</i> - 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

Parameters	Details
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 fimrware version x.14 and later, combining semantic clusters) defined in the serial string. Invalid characters (or in fimrware version x.14 and later, combining semantic clusters) are assumed to be equal to a value of zero in that characters (or in fimrware version x.14 and later, combining semantic clusters) position. The increment value for alphabetic strings start with ‘A’ or ‘a’ as the zero placeholder. This means to increment an alphabetic character (or in fimrware 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	GENERATED LABELS
<pre>^XA ^FO100,100 ^CF0,100 ^FD12A^SFnnA,F^FS ^PQ3 ^XZ</pre>	12K 12F 12A

This mask has the first characters (or in fimrware version x.14 and later, the first combining semantic clusters) as alphanumeric (nn = 12) and the last digit as 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 label, 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...
```



Examples • These examples show the importance of capitalization and location within the mask.

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
<pre> ^XA ^FO100,50^A0N,50,50^FDzzZ^SFnnN,I^FS ^PQ10 ^XZ </pre>	

In this example, lower case 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
<pre> ^XA ^FO100,50^A0N,50,50^FDzzZ^SFnnN,i^FS ^PQ10 ^XZ </pre>	



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.

^SI

Set Sensor Intensity



This command is available only for printers with firmware versions V53.15.x or later.

Description 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.

Supported Devices

- Xi4 with firmware V53.17.1Z or later
- ZM400/ZM600 with firmware V53.15.xZ or later

Format ^SIa, b

This table identifies the parameters for this format:

Parameters	Details
a = indicates the setting to modify	<i>Accepted Values:</i> 1 = transmissive sensor brightness setting 2 = transmissive sensor baseline setting <i>Default Value:</i> 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. <i>Accepted Values:</i> 0 to 196 <i>Default Value:</i> must be an accepted value or the entire command is ignored

^SL

Set Mode and Language (for Real-Time Clock)

Description The ^SL command is used to specify the Real-Time Clock’s mode of operation and language for printing information.



Important •

- Time is read when the image is created. If the image stays in the queue longer than the specified time the image will be recreated with a new time.
- There are incidents when the same time or a larger space of time may be printed on labels. This is due to the format complexity and print speed.

Format ^SLa, b

This table identifies the parameters for this format:

Parameters	Details
a = mode	<i>Accepted Values:</i> 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. <i>Time Now</i> 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 Example: SL30,1 = Accuracy tolerance of 30 seconds and use English. <i>Default Value:</i> S

a. These values are only supported on the Xi4 and ZM400/ZM600 printers.

Parameters	Details
b = language .14↑ Value 13 is only supported in firmware versions V60.14.x, V50.14.x, or later.	<i>Accepted Values:</i> 1 = English 2 = Spanish 3 = French 4 = German 5 = Italian 6 = Norwegian 7 = Portuguese 8 = Swedish 9 = Danish 10 = Spanish 2 11 = Dutch 12 = Finnish 13 = Japanese 14 = Korean ^a 15 = Simplified Chinese ^a 16 = Traditional Chinese ^a 17 = Russian ^a 18 = Polish ^a <i>Default Value:</i> the language selected with ^KL or the control panel

a. These values are only supported on the Xi4 and ZM400/ZM600 printers.

Comments These are some comments to be aware of:

- The ^SL command must be placed before the first ^F0 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 977](#).

^SN

Serialization Data

Description 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 bar code fields. A maximum of 12 of the right-most integers are subject to indexing.



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 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

This table identifies the parameters for this format:

Parameters	Details
v = starting value	<i>Accepted Values:</i> 12-digits maximum for the portion to be indexed <i>Default Value:</i> 1
n = increment or decrement value	<i>Accepted Values:</i> 12-digit maximum <i>Default Value:</i> 1 To indicate a decrement value, precede the value with a minus (–) sign.
z = add leading zeros (if needed)	<i>Accepted Values:</i> N = no Y = yes <i>Default Value:</i> N



Example • This example shows incrementing by a specified value:

ZPL II CODE	GENERATED LABELS
<pre> ^XA ^FO260,110 ^CFG ^SN001,1,Y^FS ^PQ3 ^XZ </pre> <i>Note: The ZPL II code above will generate three separate labels, seen to the right.</i>	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 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



In x.13 and earlier, the starting value consists of the right-most consecutive sequence of digits.



In x.14 and later, the starting value consists of the first number working backwards 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



In x.13 and earlier, the starting value consists of the right-most consecutive sequence of digits, including any leading spaces.



In x.14 or later, the starting value consists of the first number working backwards in the backing store consecutive sequence of digits, including any leading spaces.

The starting value consists of the first number working backwards in the backing store consecutive sequence of digits, including any leading spaces. The width (number of digits in the sequence) is determined by scanning from right to left until the first alpha character (except a space) is encountered. To create a specific width, manually place leading spaces or zeros as necessary. Suppressed zeros are replaced by spaces. During the serialization process, when the entire number contains all zeros, the last zero is not suppressed.

The ^SN command replaces the Field Data (^FD) command within a label formatting program.

^SO

Set Offset (for Real-Time Clock)

Description The ^SO command is used to set the secondary and the tertiary offset from the primary 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 ^SOa,b,c,d,e,f,g

This table identifies the parameters for this format:

Parameters	Details
a = clock set	<i>Accepted Values:</i> 2 = secondary 3 = third <i>Default Value:</i> value must be specified
b = months offset	<i>Accepted Values:</i> -32000 to 32000 <i>Default Value:</i> 0
c = days offset	<i>Accepted Values:</i> -32000 to 32000 <i>Default Value:</i> 0
d = years offset	<i>Accepted Values:</i> -32000 to 32000 <i>Default Value:</i> 0
e = hours offset	<i>Accepted Values:</i> -32000 to 32000 <i>Default Value:</i> 0
f = minutes offset	<i>Accepted Values:</i> -32000 to 32000 <i>Default Value:</i> 0
g = seconds offset	<i>Accepted Values:</i> -32000 to 32000 <i>Default Value:</i> 0

For more detail on set offset, see [Real Time Clock on page 977](#).

^SP

Start Print

Description 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.

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, media motion does not stop. If the next segment is not ready, the printer stops mid-label and wait 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

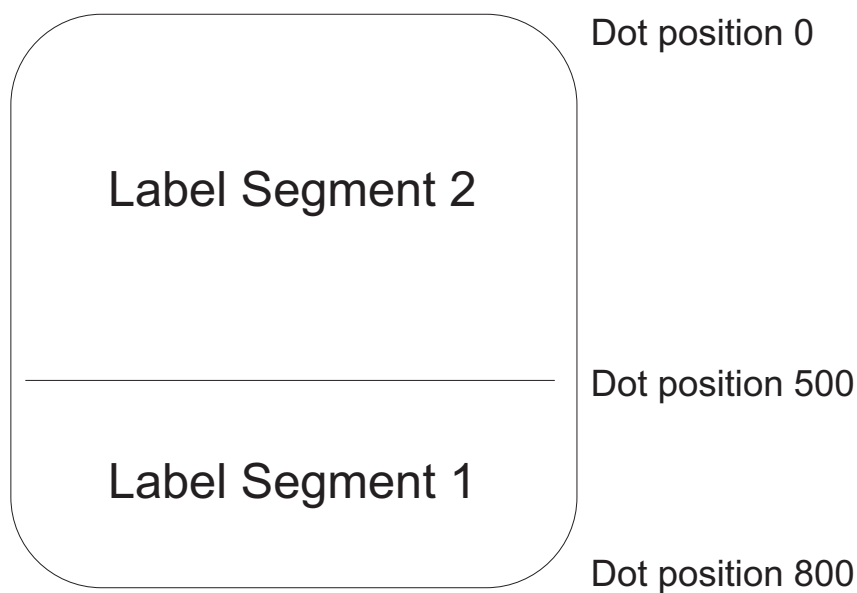
Comments ^SPa

This table identifies the parameters for this format:

Parameters	Details
a = dot row to start printing	<i>Accepted Values:</i> 0 to 32000 <i>Default Value:</i> 0



Example • In this example, a label 800 dot rows in length uses ^SP500. Segment 1 prints while commands in Segment 2 are being received and formatted.



^SQ

Halt ZebraNet Alert

Description The ^SQ command is used to stop the ZebraNet Alert option.

Format ^SQa,b,c

This table identifies the parameters for this format:

Parameters	Details
a = condition type	<i>Accepted Values:</i> 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	<i>Accepted Values:</i> 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	<i>Accepted Values:</i> Y = halt messages N = start messages <i>Default Value:</i> Y

^SR

Set Printhead Resistance

Description The ^SR command allows you to set the printhead resistance.

Format ^SR####

This table identifies the parameters for this format:

Parameters	Details
#### = resistance value (four-digit numeric value)	<i>Accepted Value:</i> 0488 to 1175 <i>Default Value:</i> 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

Set Media Sensors

Description 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.

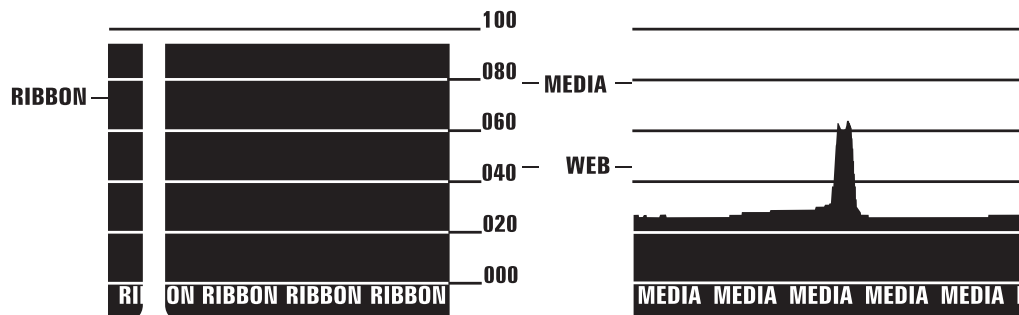
Format ^SSw,m,r,l,m2,r2,a,b,c

This table identifies the parameters for this format:

Parameters	Details
w = web (three-digit value)	<i>Accepted Values:</i> 000 to 100 <i>Default Value:</i> value shown on the media sensor profile or configuration label
m = media (three-digit value)	<i>Accepted Values:</i> 000 to 100 <i>Default Value:</i> value shown on the media sensor profile or configuration label
r = ribbon (three-digit value)	<i>Accepted Values:</i> 000 to 100 <i>Default Value:</i> value shown on the media sensor profile or configuration label
l = label length (in dots, four-digit value)	<i>Accepted Values:</i> 0001 to 32000 <i>Default Value:</i> value calculated in the calibration process
m2 = intensity of media LED (three-digit value)	<i>Accepted Values:</i> 000 to 255 <i>Default Value:</i> value calculated in the calibration process
r2 = intensity of ribbon LED (three-digit value)	<i>Accepted Values:</i> 000 to 255 <i>Default Value:</i> value calculated in the calibration process
a = mark sensing (three-digit value)	<i>Accepted Values:</i> 000 to 100 <i>Default Value:</i> value calculated in the calibration process
b = mark media sensing (three-digit value)	<i>Accepted Values:</i> 000 to 100 <i>Default Value:</i> value calculated in the calibration process
c = mark LED sensing (three-digit value)	<i>Accepted Values:</i> 000 to 255 <i>Default Value:</i> 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

Set Date and Time (for Real-Time Clock)

Description The ^ST command sets the date and time of the Real-Time Clock.

Format ^STa, b, c, d, e, f, g

This table identifies the parameters for this format:

Parameters	Details
a = month	<i>Accepted Values:</i> 01 to 12 <i>Default Value:</i> current month
b = day	<i>Accepted Values:</i> 01 to 31 <i>Default Value:</i> current day
c = year	<i>Accepted Values:</i> 1998 to 2097 <i>Default Value:</i> current year
d = hour	<i>Accepted Values:</i> 00 to 23 <i>Default Value:</i> current hour
e = minute	<i>Accepted Values:</i> 00 to 59 <i>Default Value:</i> current minute
f = second	<i>Accepted Values:</i> 00 to 59 <i>Default Value:</i> current second
g = format	<i>Accepted Values:</i> A = a.m. P = p.m. M = 24-hour military <i>Default Value:</i> M

For more details on set date and time, see [Real Time Clock on page 977](#).

^SX

Set ZebraNet Alert

Description The ^SX command is used to configure the ZebraNet Alert System.

Format ^SXa,b,c,d,e,f

This table identifies the parameters for this format:



Note • The values in this table apply to firmware version V48.12.4 or later.

Parameters	Details
a = condition type	<i>Accepted Values:</i> - 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) - W = all errors (in RFID printers only) * = all errors (in non-RFID printers) <i>Default Value:</i> if the parameter is missing or invalid, the command is ignored

Parameters	Details
b = destination for route alert	<i>Accepted Values:</i> A = serial port B* = parallel port C = e-mail address D = TCP/IP E = UDP/IP F = SNMP trap <i>Default Value:</i> if this parameter is missing or invalid, the command is ignored * Requires bidirectional communication.
c = enable <i>condition set</i> alert to this destination	<i>Accepted Values:</i> N = no Y = yes <i>Default Value:</i> Y or previously configured value
d = enable condition clear alert to this destination	<i>Accepted Values:</i> N = no Y = yes <i>Default Value:</i> 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	<i>Accepted Values:</i> 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	<i>Accepted Values:</i> TCP port # (0 to 65535) UDP 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

Set ZPL

Description 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.

This command remains active until another ^SZ command is sent to the printer or the printer is turned off.

Format ^SZa

This table identifies the parameters for this format:

Parameters	Details
a = ZPL version	<i>Accepted Values:</i> 1 = ZPL 2 = ZPL II <i>Default Value:</i> 2

Comments If the parameter is missing or invalid, the command is ignored.

~TA

Tear-off Adjust Position

Description 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.

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.

This table identifies the parameters for this format:

Parameters	Details
### = change in media rest position (3-digit value in dot rows must be used.)	<i>Accepted Values:</i> -120 to 120 0 to 120 (on the HC100) <i>Default Value:</i> last permanent value saved

Comments If the parameter is missing or invalid, the command is ignored.

^TB

Text Blocks



This command is available only for printers with firmware version V60.14.x, V50.14.x, or later.

Description 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.



Note • ^TB is the preferred command for printing fields or blocks of text, instead of ^FB.

Format ^TBa, b, c

This table identifies the parameters for this format:

Parameters	Details
a = block rotation	<i>Accepted Values:</i> N = normal R = rotate 90 degrees clockwise I = invert 180 degrees B = read from bottom up-270 degrees <i>Default Value:</i> whatever was specified by the last ^A (which has the default of ^FW)
b = block width in dots	<i>Accepted Values:</i> 1 to the width of the label in dots <i>Default Value:</i> 1 dot
c = block height in dots	<i>Accepted Values:</i> 1 to the length of the label in dots <i>Default Value:</i> 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. This is a list of defined escape sequences: <<> prints <.
- The ^TB command has an automatic word-wrap function. Soft hyphens do not print and are not used as a line break position.

^TO

Transfer Object

Description The ^TO 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.

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 ^TO command.

At least one source parameter (d, O, or X) and one destination parameter (S, O, or X) must be specified. If only ^TO is entered, the command is ignored.

Format ^TOs:o.x,d:o.x

This table identifies the parameters for this format:

Parameters	Details
S = source device of stored object	<i>Accepted Values:</i> R:, E:, B:, and A: <i>Default Value:</i> if a drive is not specified, all objects are transferred to the drive set in parameter S
O = stored object name	<i>Accepted Values:</i> any existing object conforming to Zebra conventions <i>Default Value:</i> if a name is not specified, * is used — all objects are selected
X = extension	<i>Accepted Values:</i> any extension conforming to Zebra conventions <i>Default Value:</i> if an extension is not specified, * is used — all extensions are selected
d = destination device of the stored object	<i>Accepted Values:</i> R:, E:, B:, and A: <i>Default Value:</i> a destination must be specified
O = name of the object at destination	<i>Accepted Values:</i> up to 8 alphanumeric characters <i>Default Value:</i> if a name is not specified, the name of the existing object is used
X = extension	<i>Accepted Values:</i> any extension conforming to Zebra conventions <i>Default Value:</i> 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 ^TO command.



Example • To copy the object **ZLOGO . GRF** from DRAM to an optional Memory Card and rename it **ZLOG01 . GRF**, write the following format:

```
^XA
^TOR:ZLOGO . GRF,B:ZLOG01 . 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

Print Configuration Label

Description 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.

Format ~WC

Comments This command works only when the printer is idle.

PRINTER CONFIGURATION	
Zebra Technologies ZTC Z4MPlus-200 dpi 140XiIIplus 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
<^> SEH.....	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.
072.....	RIBBON S.
100.....	TAKE LABEL
015.....	MEDIA LED
103.....	RIBBON LED
+10.....	LCD ADJUST
DPSWFXM.....	MODES ENABLED
.....	MODES DISABLED
832 8/MM FULL.....	RESOLUTION
V60.14.52 <-.....	FIRMWARE
V23 --- 57.....	HARDWARE ID
CUSTOMIZED.....	CONFIGURATION
3584k.....R:	RAM
2048k.....E:	ONBOARD FLASH
NONE.....	FORMAT CONVERT
CUTTER.....	OPTION
FW VERSION.....	IDLE DISPLAY
04/03/07.....	RTC DATE
15:51.....	RTC TIME
NO.....	RFID READY
SELECTED ITEMS.....	PASSWORD LEVEL
GL 47277.04MS050224.79000.04.VH2....	

FIRMWARE IN THIS PRINTER IS COPYRIGHTED

^WD

Print Directory Label

Description The ^WD command is used to print a label listing bar codes, objects stored in DRAM, or fonts.

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

This table identifies the parameters for this format:

Parameters	Details
d = source device — optional	<i>Accepted Values:</i> R:, E:, B:, A: and Z: <i>Default Value:</i> R:
o = object name — optional	<i>Accepted Values:</i> 1 to 8 alphanumeric characters <i>Default Value:</i> * The use of a ? (question mark) is also allowed.
x = extension — optional	<i>Accepted Values:</i> any extension conforming to Zebra conventions .14↑ .TTF and .TTE are only supported in firmware version V60.14.x, V50.14.x, or later. .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 <i>Default Value:</i> * 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

Write Query

Description The ~WQ command triggers the printer to print a label with odometer, maintenance or alert, and printhead history information.

Supported Devices

- Xi4 with firmware V53.17.1Z or later
- ZM400/ZM600 with firmware V53.15.xZ or later
- S4M with firmware V53.15.4Z or later
- G-Series

Format ~WQquery-type

This table identifies the parameters for this format:

Parameter	Details
query-type	For detailed examples of these parameters, see ~WQ Examples on page 368. <i>Accepted Values:</i> - ES = requests the printer's status. For details see, Table 16 on page 366 and Table 17 on page 367. - 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 <i>Default Value:</i> must be an accepted value or the command is ignored

Table 16 • Error Flags (~WQES)

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 Reset P	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 Temperature	1	00000000	X	X	X	X	X	X	2	X
Printhead Temperature	1	00000000	X	X	X	X	X	X	1	X
Cutter Fault F	1	00000000	X	X	X	X	X	X	X	8
Head Pen O	1	00000000	X	X	X	X	X	X	X	4
Ribbon Out O	1	00000000	X	X	X	X	X	X	X	2
Media Out O	1	00000000	X	X	X	X	X	X	X	1
Clear Paper Path Failed ^a	1 ^a	00000000	X	X	X	X	8 ^a	X	X	X
Paper Feed Error ^a	1 ^a	00000000	X	X	X	X	4 ^a	X	X	X
Presenter Not Running ^a	1 ^a	00000000	X	X	X	X	2 ^a	X	X	X
Paper Jam during Retract ^a	1 ^a	00000000	X	X	X	X	1 ^a	X	X	X
Black Mark not Found ^a	1 ^a	00000000	X	X	X	8 ^a	X	X	X	X
Black Mark Calibrate Error ^a	1 ^a	00000000	X	X	X	4 ^a	X	X	X	X
Retract Function timed out ^a	1 ^a	00000000	X	X	X	2 ^a	X	X	X	X
Paused ^a	1 ^a	00000000	X	X	X	1 ^a	X	X	X	X

a. This error flag is supported only on KR403 printers.

Table 17 • Warning Flags (~WQES)

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 warning W	0	00000000	0	0	0	0	0	0	0	0
Warning resent P	1	00000000	X	X	X	X	X	X	X	X
Paper-near-end Sensor ^b	1 ^b	00000000	X	X	X	X	X	X	X	8 ^b
Replace rintheadP	1	00000000	X	X	X	X	X	X	X	4
Clean rintheadP	1	00000000	X	X	X	X	X	X	X	2
Need to calibrate Media M	1	00000000	X	X	X	X	X	X	X	1
Sensor 1 (Paper before head) ^b	1 ^b	00000000	X	X	X	X	X	X	1 ^b	X
Sensor 2 (Black mark) ^b	1 ^b	00000000	X	X	X	X	X	X	2 ^b	X
Sensor 3 (Paper after head) ^b	1 ^b	00000000	X	X	X	X	X	X	4 ^b	X
Sensor 4 (loop ready) ^b	1 ^b	00000000	X	X	X	X	X	X	8 ^b	X
Sensor 5 (presenter) ^b	1 ^b	00000000	X	X	X	X	X	1 ^b	X	X
Sensor 6 (retract ready) ^b	1 ^b	00000000	X	X	X	X	X	2 ^b	X	X
Sensor 7 (in retract) ^b	1 ^b	00000000	X	X	X	X	X	4 ^b	X	X
Sensor 8 (at bin) ^b	1 ^b	00000000	X	X	X	X	X	8 ^b	X	X

b. 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.

1. To request the printer’s status, type ~WQES

A label similar to this prints out:

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:

PRINTER STATUS			
ERRORS:	1	00000000	00000005
WARNINGS:	1	00000000	00000002

2

4

6

1

3

5

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.

1. To request the printer's status, type `~WQES`

A label similar to this prints out:

PRINTER STATUS		
ERRORS:	1	00000000 0000000B
WARNINGS:	0	00000000 00000000

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:

MAC ADDRESS
00:07:4d:2c:e0:7a



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 on page 273](#).

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,0000,0000,0000

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:

PRINT HEAD TEST RESULTS	
<u>0,A,0000,0000,0000</u>	
1	2
3	4
5	

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:

PRINT METERS		
TOTAL NONRESETTABLE:	8560	"
USER RESETTABLE CNTR1:	9	"
USER RESETTABLE CNTR2:	8560	"

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.

2. To check the settings, type ~WQOD.

A label similar to this prints out:

PRINT METERS		
TOTAL NONRESETTABLE:	21744	cm
USER RESETTABLE CNTR1:	24	cm
USER RESETTABLE CNTR2:	21744	cm



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:

MAINTENANCE ALERT MESSAGES	
CLEAN:	PLEASE CLEAN PRINT HEAD
REPLACE:	PLEASE REPLACE PRINT HEAD

➔ **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:

	LAST CLEANED:	257 "
	HEAD LIFE HISTORY	
	# DISTANCE	
1	1:	257 "
2	2:	1489 "
	3:	7070 "

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:

PLUG AND PLAY MESSAGES
MFG: Zebra Technologies
CMD: ZPL
MDL: Gx420d

➔ **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:

SERIAL NUMBER
30A07070005

➔ **Example** • This is an example of howto 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:

USB INFORMATION	
PID:	0084
RELEASE VERSION:	15.01

^XA

Start Format

Description 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).

Format ^XA

Comments Valid ZPL II format requires that label formats should start with the ^XA command and end with the ^XZ command.

^XB

Suppress Backfeed

Description 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.

Format ^XB

^XB in the 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

Recall Format

Description 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.

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

This table identifies the parameters for this format:

Parameters	Details
d = source device of stored image	<i>Accepted Values:</i> R:, E:, B:, and A: <i>Default Value:</i> search priority (R:, E:, B:, and A:)
o = name of stored image	<i>Accepted Values:</i> 1 to 8 alphanumeric characters <i>Default Value:</i> if a name is not specified, UNKNOWN is used
x = extension l	<i>Fixed Value:</i> .ZPL

For a complete example of the ^DF and ^XF command, see [^DF and ^XF — Download format and recall format on page 49](#).

^XG

Recall Graphic

Description 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.

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

This table identifies the parameters for this format:

Parameters	Details
d = source device of stored image	Accepted Values: R:, E:, B:, and A: Default Value: search priority (R:, E:, B:, and A:)
o = name of stored image	Accepted Values: 1 to 8 alphanumeric characters Default Value: if a name is not specified, UNKNOWN is used
x = extension l	Fixed Value: .GRF
mx = magnification factor on the x-axis	Accepted Values: 1 to 10 Default Value: 1
my = magnification factor on the y-axis	Accepted Values: 1 to 10 Default Value: 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
^F0100,100^XGR: SAMPLE.GRF,1,1^FS
^F0100,200^XGR: SAMPLE.GRF,2,2^FS
^F0100,300^XGR: SAMPLE.GRF,3,3^FS
^F0100,400^XGR: SAMPLE.GRF,4,4^FS
^F0100,500^XGR: SAMPLE.GRF,5,5^FS
^XZ
```


^XS

Set Dynamic Media Calibration

Description 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).

Supported Devices

- G-Series

Format ^XSlength,threshold

This table identifies the parameters for this format:

Parameters	Details
length = dynamic length calibration	<i>Accepted Values:</i> Y = enable N = disable <i>Default Value:</i> Y
threshold = dynamic threshold calibration	<i>Accepted Values:</i> Y = enable N = disable <i>Default Value:</i> Y
gain = dynamic gain calibration (to be in a future implementation)	<i>Accepted Values:</i> Y = enable N = disable <i>Default Value:</i> Y

^XZ

End Format

Description 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).

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

Printer Sleep

Description The ^ZZ command places the printer in an idle or shutdown mode.

Format ^ZZt,b

This table identifies the parameters for this format:

Parameters	Details
t = number of second (idle time) prior to shutdown	<i>Accepted Values:</i> 0 to 999999 – setting 0 disables automatic shutdown <i>Default Value:</i> last permanently saved value or 0
b = label status at shutdown	<i>Accepted Values:</i> Y = indicates to shutdown when labels are still queued N = indicates all labels must be printed before shutting down <i>Default Value:</i> N

Comments The ^ZZ command is only valid on the PA400 and PT400 battery-powered printers.



Notes • _____



ZPL RFID Commands

This section contains the ZPL II commands for RFID-specific applications.

For more information about the RFID commands, refer to the *RFID Programming Guide*. A copy is available on the User CD provided with your printer and online at <http://www.zebra.com/manuals>.

RFID Command Overview

In addition to reading or encoding RFID tags, the RFID ZPL commands also provide for RFID exception handling, such as setting the number of read/write retries before declaring a transponder defective (set with ^RR, ^RT, and ^WT) or setting the number of labels that will be attempted if an error occurs (set with ^RS).

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 RFID labels specified by the ^RS command. If the problem persists, the printer follows the error handling instructions specified by the ^RS command: 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 • Consider the following before using any command in this section:

- Before using a particular command, verify that it is compatible with your printer and firmware version. See [Table 18 on page 383](#).
- If a parameter in the following tables is designated as *not applicable* for a particular printer, any value entered for the parameter will be ignored, but the place holder for the field is required.

Printer and Firmware Compatibility

Table 18 shows which RFID ZPL commands you can use with different printers and firmware versions.

Table 18 • Supported Commands Based on Printer and Firmware

Command	Function	UHF Printers										HF Printers	
		R110Xi and R170Xi	RZ400 and RZ600	R110PAX4		R4Mplus					RP4T	R110Xi HF	R2844-Z
Firmware													
all	R53.16.X	R62.X.X	R63.X.X	SP994X	SP999X	SP1027X	SP1056X	SP1082X	all	all	all		
^HL or ~HL on page 385	Return RFID Data Log to Host	*	*	*	*	*	*	*	*	*	*	*	—
^HR on page 386	Calibrate RFID Transponder Position	*	*	*	*	*	*	*	*	*	*	*	—
^RA on page 390	Read AFI or DSFID Byte	—	—	—	—	—	—	—	—	—	—	*	—
^RB on page 392	Define EPC Data Structure	*	*	*	*	*	*	*	*	*	*	*	—
^RE on page 394	Enable/Disable E.A.S. Bit	—	—	—	—	—	—	—	—	—	—	*	*
^RF on page 395	Read or Write RFID Format	*	*	*	*	*	*	*	*	*	*	*	—
^RI on page 398	Get RFID Tag ID	* (R60.13.0.13ZD or later)	*	* (R62.13.0.13ZC or later)	* (R63.13.0.12Z or later)	*	*	*	*	*	*	*	*
^RM on page 399	Enable RFID Motion	*	*	*	*	*	*	*	*	*	*	*	—
^RN on page 400	Detect Multiple RFID Tags in Encoding Field	* (R60.13.0.3 or later)	*	*	—	—	—	—	—	—	*	*	—
~RO on page 331	Reset Advanced Counters	*	*	*	*	*	*	*	*	*	*	*	—
^RQ on page 401	Quick Write EPC Data and Passwords	* (R60.15.7Z or later)	*	* (R62.15.7Z or later)	—	—	—	—	—	—	*	—	—

* = Supported — = Not supported

- a. Not ^FH capable. Also, parameters t and a do not apply. b. For parameter e, the only accepted value is N for No Action.
c. Use the ^RF, ^RM, and ^RR commands rather than the ^RT command. d. Use the ^RF, ^RM, ^RR, and ^WV commands rather than the ^WT command.

Table 18 • Supported Commands Based on Printer and Firmware (Continued)

Command	Function	UHF Printers										HF Printers	
		R110Xi and R170Xi	RZ400 and RZ600	R110PAX4	R4Mplus						RP4T	R110Xi HF	R2844-Z
Firmware													
all	R53.16.X	R62.X.X	R63.X.X	SP994X	SP999X	SP1027X	SP1056X	SP1082X	all	all	all		
^RR on page 403	Specify RFID Retries for a Block	*	*	*	*	*	*	*	*	*	*	*	—
^RS on page 404	Set Up RFID Parameters	*	*	*	*	*	*	*	*	*	* b	*	*
^RT on page 411	Read RFID Tag	* c	* c	* c	* c	* c	* c	* c	* c	* c	* c	* c	*
~RV on page 414	Report RFID Encoding Results	*	*	*	*	* (SP994J or later)	* (SP999C or later)	*	*	*	*	—	—
^RW on page 415	Set RF Power Levels for Read and Write	*	*	*	*	*	*	*	*	*	*	*	—
^RZ on page 417	Set RFID Tag Password and Lock Tag	*	*	*	*	*	*	*	*	*	*	—	—
^WF on page 420	Encode AFI or DSFID Byte	—	—	—	—	—	—	—	—	—	—	*	—
^WT on page 422	Write (Encode) Tag	* d	* d	* d	* d	* d	* d	* d	* d	* d	* d	* d	*
^WV on page 424	Verify RFID Encoding Operation	*	*	*	*	*	*	*	*	*	*	—	—

* = Supported — = Not supported

- a. Not ^FH capable. Also, parameters t and a do not apply. b. For parameter e, the only accepted value is N for No Action.
 c. Use the ^RF, ^RM, and ^RR commands rather than the ^RT command. d. Use the ^RF, ^RM, ^RR, and ^WV commands rather than the ^WT command.

^HL or ~HL

Return RFID Data Log to Host

Description The printer continually logs RFID data and stores it in the printer's RAM. Use this command to request that the RFID data log be returned to the host computer, to clear the current data log, and to restart data recording. The data returned show the status of the RFID read, write, and lock commands and show any resulting error codes.

Format ^HL or ~HL

Comments

- Data is shown in the format sent by the ^RFW command (ASCII, Hex, or EPC).
- In the log, the data displays in this manner:
C, EEEE, DDDDDDDDDDDDDDDDDDDDDDDDDDDDD
where
C = the RFID operation (R = read, W = write, L = lock)
EEEE = the RFID error code
DDDDDDDDDDDDDDDDDDDDDDDDDDDD = data read or written
- If the log exceeds 64K (approximately 2000 operations), the data log is cleared automatically, and data recording restarts. When this happens, the following appears in the log:
Logfile automatically reset
- If the printer loses power, the log is lost. If the log results are important to you, retrieve the log frequently.

^HR

Calibrate RFID Transponder Position

Description Use this command to initiate an RFID transponder calibration for a specific RFID label. Results are returned to the host computer. This calibration is used to determine the optimal programming position for RFID media that may not meet the transponder placement specifications for the printer.

During transponder calibration, the printer feeds the RFID label one-dot row at a time while taking readings (via the READ TAG command and the WRITE TAG commands) to profile the RFID transponder. Based on the results, the printer determines the optimal programming position for the label and returns a results table to the host. The calibrated value is used as the programming position for the ^RS command, can be overwritten by the ^RS command, and is saved to nonvolatile memory (the value is saved even if the power is turned off).

This calibration takes into account the print mode, backfeed mode, and tear off position. The **RUN** option in the **RFID TAG CALIB** control panel parameter performs the same calibration but does not create a results table.



Important • If a label format specifies a value for parameter **p** (read/write position of the transponder) in the ^RS command, that value will be used for the programming position for all RFID labels until a new position is specified or until the printer is turned Off (**O**) and then back On (**I**).

Format ^HRa,b,c,d

This table identifies the parameters for this format.

Parameters	Details
a = start string	User text to appear before the results table. <i>Accepted values:</i> any string less than 65 characters <i>Default value:</i> start
b = end string	User text to appear after the results table. <i>Accepted values:</i> any string less than 65 characters <i>Default value:</i> end
c = start position	Start position of calibration range. <i>Default Value:</i> Forward: F0 to Fxxx (where xxx is the label length in millimeters, or 999, whichever is less) Backward: B0 to B30 (in millimeters, 30 mm maximum) <i>Default value:</i> F0  Note • The printer will move the media to the start position relative to the leading edge of the label and then perform the RFID tag calibration.

Parameters	Details
d = end position	End position of calibration range. <i>Accepted Values:</i> Forward: F0 to Fxxx (where xxx is the label length in millimeters, or 999, whichever is less) Backward: B0 to B30 (in millimeters, 30 mm maximum) <i>Default Value:</i> Label length as shown on the ZPL configuration label

Comments

- Based on the recommended transponder placement position for most RFID labels, the printer's default RFID programming position is zero for the R110PAX4. For other RFID printers, the default programming position is the label length minus 1 mm (0.04 in.). To return to the default programming position at any time, use the RESTORE option in the **RFID TAG CALIB** control panel parameter.
- At the end of calibration, a results table is returned to the host. 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)



Example 1 • If 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
```

In this example, 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 2 • The example shown below is the result of ^HRstart,end,B12,F82.

```
start
Position=F20
Leading edge
B12, ,
B9, ,
B8, ,
...
B1, ,
F0, ,
F1, ,
F2, ,
F3, ,
...
F12, ,
F13,R,
F14,R,W
...
F19,R,W
F20,R,W  <--- ***** F20 mm
F21,R,W
...
F26,R,W
F27,R,
F28, ,
..
F81, ,
F82, ,
Trailing edge
end
```

^RA

Read AFI or DSFID Byte



Important • This command is not supported by all printers or firmware. See [Printer and Firmware Compatibility on page 383](#) for the list of printers and firmware with which you can use this command.

Description Use this command to read the AFI or DSFID byte. The data can be returned to the host via the ^HV command.

Format ^RA#, f, r, m, b

This table identifies the parameters for this format.

Parameters	Details
# = field number specified with another command	The value assigned to this parameter should be the same as the one used in the ^RT command. <i>Accepted values:</i> 0 to 9999 <i>Default value:</i> 0
f = format	<i>Accepted values:</i> 0 = ASCII 1 = Hexadecimal <i>Default value:</i> 0
r = number of retries	<i>Accepted values:</i> 0 to 10 <i>Default value:</i> 0
m = motion	<i>Accepted values:</i> 0 = Feed label after writing. 1 = No Feed after writing. Other ZPL may cause a feed. <i>Default value:</i> 0
b = type of byte to read	<i>Accepted values:</i> A = AFI byte D = DSFID byte <i>Default value:</i> A



Example 1 • This example reads the AFI byte in ASCII format and returns AFI Byte:x to the host. The printer will retry the command five times if necessary. A voided label is generated if the read is unsuccessful after these retries. The data read will go into the ^FN1 location of the recalled format.

```
^XA
^F020,120^A0N,60^FN1^FS
^RA1,0,5,0^FS
^HV1,,AFI Byte:^FS
^XZ
```



Example 2 • This example reads the DSFID byte in ASCII format and returns DSFID Byte:x to the host. The printer will retry the command three times if necessary. A voided label is generated if the read is unsuccessful after these retries. The data read will go into the ^FN1 location of the recalled format.

```
^XA
^F020,120^A0N,60^FN1^FS
^RA1,0,3,0,D^FS
^HV1,,DSFID Byte:^FS
^XZ
```

^RB

Define EPC Data Structure



Important • This command is not supported by all printers or firmware. See [Printer and Firmware Compatibility on page 383](#) for the list of printers and firmware with which you can use this command.

Description Use this command to define the structure of EPC data, which can be read from or written to an RFID transponder. 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.

RFID transponders 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

This table identifies the parameters for this format.

Parameters	Details
n = total bit size of the partitions	Specify the number of bits to include in the partitions. <i>Accepted values:</i> 1 to n, where n is the bit size of the tag. <i>Default value:</i> 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. <i>Accepted values:</i> 1 to 64 <i>Default value:</i> 1



Example 1 •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^FD1000.67108000.1122921504606846976^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 2 •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 3 •This example uses the SGTIN-64 standard, which defines 64-bit structure in the following way:

	Header	Filter Value	Company Prefix Index	Item Reference	Serial Number
SGTIN-64	2 bits	3 bits	14 bits	20 bits	25 bits
	10 (binary value)	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
^RB64,2,3,14,20,25
^RFW,E^FD0,3,12345,544332,22335221^FS
^XZ
```

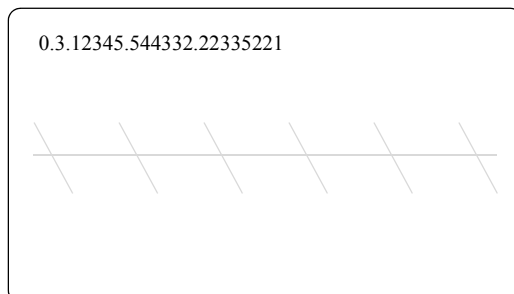
These commands would put

- 0 in the header
- 3 as the filter value
- 12345 as the company prefix
- 544332 as the item reference
- 22335221 as the serial number

To read this EPC data and print the results on the label, you would use the following code:

```
^XA
^RB64,2,3,14,20,25
^F050,50^A0N,40^FN0^FS
^FN0^RFR,E^FS
^XZ
```

The resulting label would look like this:



^RE

Enable/Disable E.A.S. Bit



Important • This command is not supported by all printers or firmware. See [Printer and Firmware Compatibility on page 383](#) for the list of printers and firmware with which you can use this command.

Description Use this command to enable or disable the Electronic Article Surveillance (E.A.S.) bit that is available in some ISO15693 tags (such as Philips). This command works only on those ISO15693 transponders and will be ignored if the tag does not support E.A.S.

Format ^REt,r

The following table identifies the parameters for this format.

Parameters	Details
t = Enable/disable the E.A.S. bit in the ISO15693 transponder	<i>Accepted values:</i> N = Disable E.A.S. Y = Enable E.A.S. <i>Default value:</i> N
r = number of retries	<i>Accepted values:</i> 0 to 10 <i>Default value:</i> 0



Example • This example enables the E.A.S. bit in the transponder. It will retry the command five times if necessary.

```
^XA
^REy,5
^XZ
```

^RF

Read or Write RFID Format

Description Use this command to read or write to (encode) an RFID tag. 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.

Format ^RFo,f,b,n,m

This table identifies the parameters for this format.

Parameters	Details
o = operation	Specifies the action to be performed. <i>Accepted Values:</i> - 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) <i>Default Value:</i> W
f = format	<i>Accepted Values:</i> - A = ASCII - H = Hexadecimal - E = EPC (ensure proper setup with the ^RB command) <i>Default Value:</i> H
b = starting block number	For tag types other Gen 2: Specifies the starting block number. <i>Accepted Values:</i> 0 to <i>n</i>, where <i>n</i> is the maximum number of blocks for the tag. <i>Default Value:</i> 0 For Gen 2 tag type only: What you specify for this parameter depends on what you entered for the operation parameter. - When W, L, or R are specified for the operation parameter, this parameter specifies a 16-bit word block number. <i>Accepted Values:</i> 0 to <i>n</i>, where <i>n</i> is the maximum number of blocks for the bank specified in the memory bank parameter <i>Default Value:</i> 0 - When P is specified for the operation parameter, this parameter specifies which password to read. <i>Accepted Values:</i> - K = kill password - A = access password <i>Default Value:</i> <u>K</u>

Parameters	Details
n = number of bytes to read or write	Specifies the number of bytes to read or write. For high-frequency (HF) printers: <i>Accepted Values:</i> 1 to <i>n</i>, where <i>n</i> is the maximum number of bytes for the tag. <i>Default Value:</i> 1 For Gen 2 tag type only: When E is specified for the memory bank parameter, this value is not required. <i>Accepted Values:</i> 1 to <i>n</i>, where <i>n</i> is the maximum number of bytes for the tag. <i>Default Value:</i> 1 For all other printers and tag types: This parameter applies only when the starting block number is 1. <i>Accepted Values:</i> 1 to <i>n</i>, where <i>n</i> is the maximum number of bytes for the tag. For UCODE EPC 1.19, <i>n</i> is 32. <i>Default Value:</i> 1
m = memory bank	 Note • This parameter applies to Gen 2 tags only. Specifies the Gen 2 memory bank. See the <i>RFID Programming Guide</i> for more information about Gen 2 memory. <i>Accepted Values:</i> - E = EPC 96-bit (command automatically performs operation on Gen 2 bit address 20_h and accesses 12 bytes of the EPC memory bank) 0 = Reserved 1 = EPC 2 = TID (Tag ID) 3 = User <i>Default Value:</i> E



Example 1 • This example encodes 96-bit data in ASCII format.

```
^XA
^RS4
^RFw,a^FD00 my data^FS
^XZ
```



Example 2 • This example encodes 64-bit data in hexadecimal format.

```
^XA
^RS3
^RFw,H^FD1122334455667788^FS
^XZ
```

➡ **Example 3** • 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 4** • This example encodes 4 bytes of hexadecimal formatted data, starting in block 3 of Gen 2 EPC bank 1.

```
^XA
^RS8
^RFW,H,3,4,1^FD11112222^FS
^XZ
```

➡ **Example 5** • 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.”

```
^XA
^RS8
^RFR,H,0,8,2^FN1^FS^HV1,,8-byte Tag ID Data:^FS
^XZ
```

^RI

Get RFID Tag ID

Description Use this command to get the unique serial number of the tag and return it in hexadecimal format. The data can be sent back to the host via the ^HV command.

For Gen 2 tag types, this command will return the 32-bit tag ID (TID) for the tag. If your Gen 2 tag supports TID data beyond 32 bits, see [^RF on page 395](#) to access the TID memory bank.

Format ^RI#,s,r,m

This table identifies the parameters for this format.

Parameters	Details
# = number to be assigned to the field	Accepted values: 0 to 9999 Default value: 0
s = specify data order	 Note • This parameter applies only to the R110Xi HF and R2844-Z printers. Accepted values: Accepted Values: 0 = Most significant byte first for Tag*It and PicoTag. Least significant byte first for I*code and ISO15693. 1 = Reverse the data order Default value: 0
r = number of retries	Accepted values: 0 to 10 Default value: 0
m = motion	Accepted values: 0 = Feed label after writing 1 = No Feed after writing (other ZPL commands may cause a feed) Default value: 0



Example • This example reads a tag ID, prints it on a label, and sends string Tag ID:XXXXXXXX to the host. The data read will go into the ^FN0 location of the format. The printer will retry the command five times, if necessary.

```

^XA
^F020,120^A0N,60^FN0^FS
^RI0,,5^FS
^HV0,,Tag ID:^FS
^XZ

```



Enable RFID Motion



Important • This command is not supported by all printers or firmware. See [Printer and Firmware Compatibility on page 383](#) for the list of printers and firmware with which you can use this command.

Description Use this command to enable or disable RFID motion. By default, labels automatically print at the end of the format. This command allows you to inhibit the label from actually moving when it reaches the program position, which is useful for debugging, setup, and custom applications. This parameter is not persistent (carried over from label to label).

Format ^RMe

This table identifies the parameters for this format.

Parameters	Details
e = enable	<i>Accepted values:</i> Y = Yes, move the label N = No, do not move the label <i>Default value:</i> Y

^RN

Detect Multiple RFID Tags in Encoding Field



Important • This command is not supported by all printers or firmware. See [Printer and Firmware Compatibility on page 383](#) for the list of printers and firmware with which you can use this command.

Description Use this command to enable or disable detection of multiple RFID tags in the encoding field. By default, the printer checks for more than one tag in the field before attempting to read or write. If more than one tag is found, the label over the antenna support is voided, and the **RFID ERR STATUS** parameter on the control panel displays **MULTIPLE TAGS**. To speed up printing and encoding by up to 200 ms, the check may be disabled. This parameter is persistent (carried over from label to label).

Format ^RNe

The following table identifies the parameters for this format.

Parameters	Details
e = enable	<i>Accepted Values:</i> Y = Yes, check for multiple tags N = No, do not check for multiple tags <i>Default Value:</i> Y

^RQ

Quick Write EPC Data and Passwords



Important • This command is not supported by all RFID tags, printers, or firmware. See [Printer and Firmware Compatibility on page 383](#) for the list of printers and firmware with which you can use this command.

Description Use this command with an Alien Higgs RFID tag and appropriate firmware to write the EPC data, access password, and kill password with one command. Doing so reduces the encoding time.



Note • The access password on the tag to be written to must be 00000000 prior to sending this command.

Format ^RQf,c,o[data]

The following table identifies the parameters for this format.

Parameters	Details
f = format	<i>Accepted Values:</i> A = ASCII H = Hexadecimal E = EPC <i>Default Value:</i> H
c = chip type	<i>Accepted Values:</i> 0 (Higgs IC tag) <i>Default Value:</i> 0
o = option	<i>Accepted Values:</i> 0 (write 96-bit EPC) <i>Default Value:</i> 0
data = the EPC data, access password, and kill password	Use the ^FD command to specify the passwords in the following format: ^FD[EPC],[access],[kill] where: EPC = the EPC data in the format specified by the f parameter. The data should match what would be programmed with the ^RF command. access = an optional access password in hexadecimal format. If this field is left blank, 0x00000000 is written as the access password. kill = an optional kill password in hexadecimal format. If this field is left blank, 0x00000000 is written as the kill password.

→ **Example 1** • This example writes the EPC data to the tag in hexadecimal format. The unspecified access and kill passwords are written as zeroes (0x00000000).

```
^XA^RQ^FD112233445566778899001122^XZ
```

→ **Example 2** • This example writes the EPC data to the tag in hexadecimal format. The access password is written as 0xAAAAAAAA, and the kill password is written as 0xBBBBBBBB.

```
^XA^RQ^FD112233445566778899001122,AAAAAAAA,BBBBBBBB^XZ
```

→ **Example 3** • This example writes the EPC data to the tag in EPC format. The unspecified access and kill passwords are written as zeroes (0x00000000).

```
^XA^RB96,30,30,30,6^RQE^FD1234.5678.9012.12^XZ
```

→ **Example 4** • This example writes the EPC data to the tag in hexadecimal format. The access password is written as 0xAAAAAAAA, and the unspecified kill password is written as zeroes (0x00000000).

```
^XA^RB96,30,30,30,6^RQE^FD1234.5678.9012.12,AAAAAAAA^XZ
```

^RR

Specify RFID Retries for a Block



Important • This command is not supported by all printers or firmware. See [Printer and Firmware Compatibility on page 383](#) for the list of printers and firmware with which you can use this command.

Description Use this command to change the number of times that the printer attempts to read or write to a particular block of a single RFID tag. By default, the printer will attempt six retries. This command is persistent and will be used in subsequent formats if not provided.



Note • This command's function is different than the “number of labels” parameter in the ^RS command.

Format ^RRn

This table identifies the parameters for this format.

Parameters	Details
n = number of retries	Accepted Values: 0 to 10 Default Value: 0



Examples •

Set read block retries to 5

```
^XA
^FN1^RR5^RFR,H^FS
^HV1^FS
^XZ
```

Set write block retries to 2

```
^XA
^RR2^RFW,H^FD1234^FS
^XZ
```

^RS

Set Up RFID Parameters

Description Use this command to set up RFID parameters including tag type, read/write position of the transponder, and error handling.



Important • 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 transponder 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 ^RS*t, p, v, n, e, a, c, s*

This table identifies the parameters for this format.

Parameters	Details
t = tag type	Tells the printer/print engine which tag type you are using. If you specify a tag type that is not supported by your printer or firmware, the printer uses its default value. For the supported tag types and defaults, see Table 19 on page 407. UHF Printers 0 = None 1 = EPC Class 0 2 = EPC Class 0 Plus 3 = EPC Class 1 64-bit 4 = EPC Class 1 96-bit 5 = UCODE EPC 1.19 6 = Impinj Class 0 Plus 7 = ISO 18000-06A 8 = EPC Class 1, Generation 2 (Gen 2) 9 = ISO 18000-06B HF Printers  Note • Only the R110Xi HF printer (firmware version R65.X.X) supports the use of letters for this parameter. All other printers use the numbers. - A or 0 = None - B or 1 = Auto detect (query tag to determine) - C or 2 = Tag*It (Texas Instruments Tagit tags) - D or 3 = I*code (Phillips Icode tags) - E or 4 = Pico Tag (Inside Technology's) - F or 5 = ISO 15693 - G or 6 = EPC tag (13.56 MHz) - H or 7 = UID Tag - I or 8 = Mifare UltraLight

Parameters	Details
p = read/write position of the transponder	Setting read/write program position using Absolute Mode (dots from top of label) Sets the read/write position of the transponder in vertical (Y axis) dot rows from the top of the label. Set to 0 (no movement) if the transponder is already in the effective area without moving the media.  Important • If a label format specifies a value for this parameter, this value will be used for the programming position for all labels until a new position is specified or until the printer is turned off (O) and then back on (I). <i>Accepted Values:</i> 0 to label length <i>Default Value:</i> For the R2844-Z: 0 For all other supported printers: label length minus 1 mm (1/16 in.) Setting read/write program position using Relative Mode (leading edge method) Sets the read/write position relative to the leading edge of the label. Specified in millimeters and allowing for distances before and after the label's leading edge. Values xx are represented in millimeters relative to the leading edge of the label. • Fxxx - Move media forward, printing bitmap, for xxx MM before reading or encoding. • Bxx - Move media backward for xx MM before reading or encoding. <i>Accepted Values:</i> Forward: F0 to Fxxx (where xxx is the label length in millimeters, or 999, whichever is less) Backward: B0 to B30 (in millimeters, 30 mm maximum)  Note • When using a backward program position, you must ensure that there is enough media to backup beyond the print line. <i>Default Value:</i> For the PAX printers: 0 For all other supported printers: F0  Note • F0 moves the leading edge of the label to the print line. Where the program position is displayed (control panel, configuration label, or web page) the program position will show the value in the method it was set: • xxxx DOTS (Absolute Mode) • Fxxx MM (Relative Mode) • Bxx MM (Relative Mode)
v = length of void printout	Sets the length of the void printout in vertical (Y axis) dot rows. <i>Accepted values:</i> 0 to label length <i>Default value:</i> label length

Parameters	Details
n = number of labels	The number of labels that will be attempted in case of read/encode failure. <i>Accepted values:</i> 1 to 10 <i>Default value:</i> 3
e = error handling	If an error persists after the specified number of labels are tried, perform this error handling action. <i>Accepted values:</i> - 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) <i>Default value:</i> 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 does not apply to the R2844-Z. For the R4Mplus, this parameter applies only to printers with firmware version SP994X (R4Mplus European version). When the value for parameter p (read/write position of the transponder) is nonzero, this parameter changes the number of start and stop print signals required for printing. In Single mode, one start print command is required. In Double mode, two are required, so the printer will resume printing only after the second start print command is received. <i>Accepted values:</i> - 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.) <i>Default value:</i> S
c = certify tag with a pre-read	Not applicable.
s = void print speed	 Note • This parameter does not apply to the R2844-Z. For the R4Mplus, this parameter applies only to printers with firmware version SP994X (R4Mplus European version). If a label is voided, the speed at which “VOID” will be printed across the label. <i>Accepted values:</i> any valid print speed <i>Default value:</i> the printer’s maximum print speed

Supported Tag Types Table 19 shows the tag types supported by different RFID printers/print engines and firmware versions. Depending on your country or on the firmware version that you are using, your printer may not support all of the tag types listed. If you specify an unsupported tag type, the printer uses the default value. If a tag type is shown as supported but does not work with your printer, you may need to upgrade the printer's firmware (see <http://www.zebra.com/firmware>).

Table 19 • Supported Commands Based on Printer and Firmware

Printer	UHF Printers											HF Printers		
	R110Xi R170Xi		R110PAX4				R4Mplus				RZ400/ RZ600	R110Xi/ HF		R2844-Z
Firmware Version	R60.13.X	R60.15.X and later	R62.13.X	R62.15.X and later	R63.13.X	R63.15.X and later	SP994O, SP999E, SP1027E, SP1082E, and earlier	SP994P, SP999F, SP1027F, SP1082F, and later	SP1056D and earlier	SP1056E and later	R53.16.X and later	R65.13.X	R65.15.X and later	all
Tag Type														
UHF Tag Types and Options														
None (no tag type specified)	*	*	*	*	—	—	—	—	—	—	—	—	—	—
EPC Class 0	*	*	*	*	—	—	—	—	—	—	—	—	—	—
EPC Class 0 Plus	*	*	*	*	—	—	—	—	—	—	—	—	—	—
EPC Class 1 64-bit	*	*	*	*	—	—	—	—	—	—	—	—	—	—
EPC Class 1 96-bit	#	*	#	*	—	*	*	*	—	—	—	—	—	—
UCODE EPC 1.19	*	a *	—	*	#	*	#	*	#	*	—	—	—	—
Impinj Class 0 Plus	*	*	*	*	—	—	—	—	—	—	—	—	—	—
ISO 18000-06A	—	—	*	—	—	*	*	*	—	—	—	—	—	—
EPC Class 1, Generation 2 (Gen 2)	*	#	*	#	—	#	*	#	*	#	#	—	—	—
ISO 18000-06B	*	*	*	*	—	*	*	*	*	*	—	—	—	—
HF Tag Types and Options														
Auto-detect the tag type by querying the tag	—	—	—	—	—	—	—	—	—	—	—	—	—	#
Tag*It (Texas Instruments Tagit tags)	—	—	—	—	—	—	—	—	—	—	—	—	—	*
I*code (Phillips Icode tags)	—	—	—	—	—	—	—	—	—	—	—	—	—	*
Pico Tag (Inside Technology’s)	—	—	—	—	—	—	—	—	—	—	—	—	—	*
ISO 15693	—	—	—	—	—	—	—	—	—	—	—	#	#	*
EPC tag	—	—	—	—	—	—	—	—	—	—	—	—	—	*

= Default value * = Accepted value — = Not supported

a. Requires R60.13.0.13ZD or later.

Table 19 • Supported Commands Based on Printer and Firmware (Continued)

Printer	UHF Printers											HF Printers	
	R110Xi R170Xi		R110PAX4				R4Mplus				RZ400/ RZ600	R110Xi HF	R2844-Z
Firmware Version	R60.13.X	R60.15.X and later	R62.13.X	R62.15.X and later	R63.13.X	R63.15.X and later	SP994O, SP999E, SP1027E, SP1082E, and earlier	SP994P, SP999F, SP1027F, SP1082F, and later	SP1056D and earlier	SP1056E and later	R53.16.X and later	R65.13.X	R65.15.X and later
Tag Type													
UID Tag	—	—	—	—	—	—	—	—	—	—	—	—	—
Mifare UltraLight	—	—	—	—	—	—	—	—	—	—	—	*	*

= Default value * = Accepted value — = Not supported

a. Requires R60.13.0.13ZD or later.

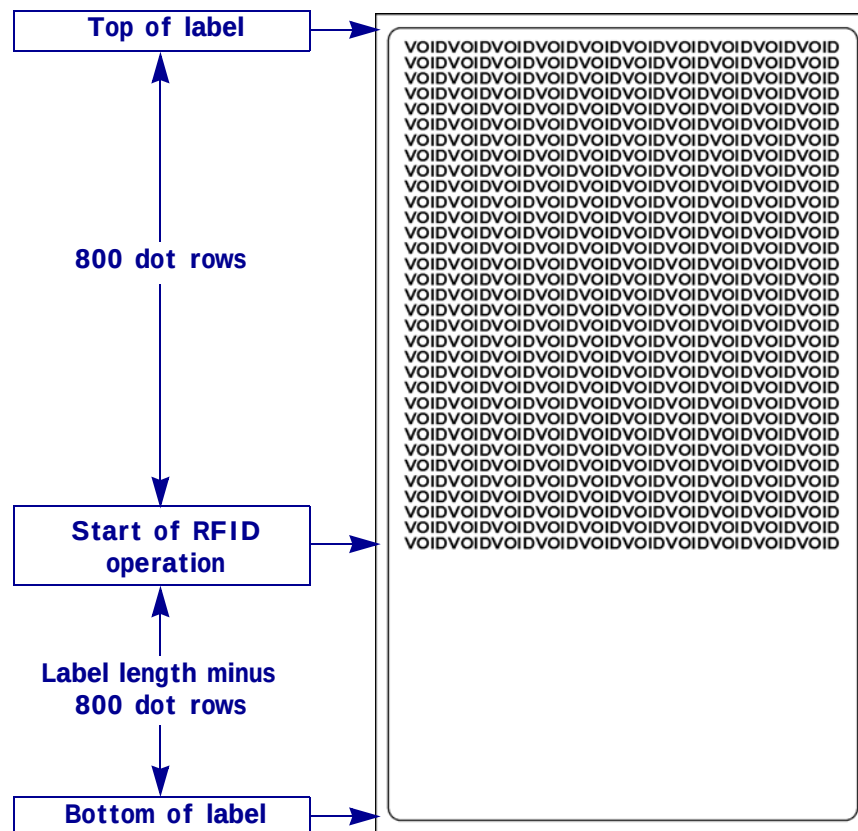


Example 1 • This example sets the printer to move the media to 800 dots from the top of the media [or label length minus 800 from the bottom (leading edge) of the media] and voids the rest of the media in case of an error. The printer will try to print two labels and then will pause if printing and encoding fail.

```
^XA  
^RS,800,,2,P^FS  
^XZ
```

Figure 3 shows the resulting voided label. Note where the void starts. The media has been moved 800 dot rows from the top of the label (label length minus 800 dot rows from the bottom (leading edge) of a label) to bring the transponder into the effective area to read/write a tag. If the printer fails the operation, the rest of the media is voided.

Figure 3 • Sample Void Label



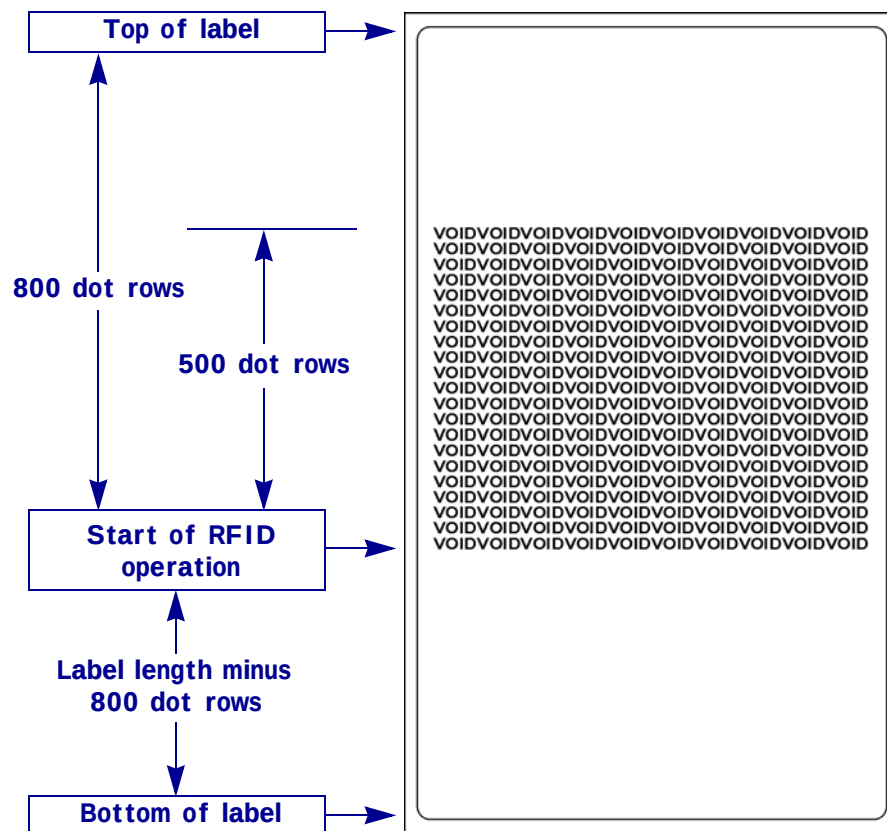


Example 2 • This example sets the printer to move the media to 800 dots from the top of the media [or label length - 500 from the bottom (leading edge) of the media] and prints “VOID” 500 dots in vertical length (Y axis) in case of an error.

```
^XA
^RS,800,500,2,P^FS
^XZ
```

Figure 4 shows the resulting voided label. Note where the void starts. The media has been moved 800 dot rows from the top of the label [label length minus 800 dot rows from the bottom (leading edge) of a label] to bring the transponder into the effective area to read/write a tag. If the printer fails the operation, an area that is 500 dot rows of the media is voided instead of the entire rest of the media.

Figure 4 • Sample Void Label, 500 Dot Row Area Voided



Example 3 • The following are examples of Absolute Mode and Relative Mode for the p parameter.

Absolute Mode

1. ^RS,520 sets the encode position at 520 dots from the top edge of the label.
2. ^RS,0 programs the tag without moving the media.

Relative Mode

1. ^RS,F1 sets the encode position 1 mm forward from the leading edge of the label.
2. ^RS,B10 sets the encode position 10 mm backwards from the leading edge of the label.
3. ^RS,F0 sets the encode position at the leading edge of the label.
4. ^RS,B0 sets the encode position at the leading edge of the label.

^RT

Read RFID Tag



Note • The ^RT command is provided only for backward-compatibility with label formats that were developed for older Zebra RFID printers. See [Printer and Firmware Compatibility on page 383](#) for the list of printers and firmware with which you should not use this command.

Description Use this command to tell the printer to read the current RFID tag data. The data can be returned to the host via the ^HV command.

Format ^RT#, b, n, f, r, m, s

This table identifies the parameters for this format.

Parameters	Details
# = number to be assigned to the field	<i>Accepted values:</i> 0 to 9999 <i>Default value:</i> 0
b = starting block number	<i>Accepted values:</i> 0 to <i>n</i> , where <i>n</i> is the maximum number of blocks for the tag. <i>Default value:</i> 0
n = number of blocks/bytes to read	 Note • This parameter does NOT apply to R4Mplus printers with firmware version SP920X (R4Mplus U.S. version). For R4Mplus printers with firmware version SP994X (European version): This parameter applies only when the starting block number (parameter b) is 1. <i>Accepted values:</i> 1 to <i>n</i> , where <i>n</i> is the maximum number of bytes for the tag. For UCODE EPC 1.19, <i>n</i> is 32. <i>Default value:</i> 1 For all other supported printers: <i>Accepted values:</i> 1 to <i>n</i> , where <i>n</i> is the maximum number of blocks for the tag type minus the starting block number. For example, if the tag has 8 blocks (starting with block 0) and you start with block 6, <i>n</i> can be 2. This would return block 6 and block 7 information. <i>Default value:</i> 1
f = format	<i>Accepted values:</i> 0 = ASCII 1 = Hexadecimal <i>Default value:</i> 0

Parameters	Details
r = number of retries	Changes the number of times that the printer attempts to read a particular block of a single RFID tag. (Same retry rules as the ^RR command.) <i>Accepted values:</i> 0 to 10 <i>Default value:</i> 0
m = motion	Enables or disables RFID motion for the current field. <i>Accepted values:</i> 0 = Feed label after writing. 1 = No feed after writing. Other ZPL may cause a feed. <i>Default value:</i> 0
S = special mode	 Note • This parameter is used only for the printers referenced here. For R4Mplus printers with firmware version SP920X (U.S. version): Specify actions for mismatched checksums. For EPC Class 1 (Alien reader) only. Not applicable for EPC class 0. <i>Default value:</i> 0 <i>Accepted values:</i> 0 = Do not read if mismatched checksum 1 = Read even if mismatched checksum For R110Xi HF and R2844-Z printers: Specify data order. <i>Default value:</i> 0 <i>Accepted values:</i> 0 = least significant byte first 1 = most significant byte first



Example 1 • This example reads a tag, prints the data on a label, and sends the string Tag Data:xxxxxxx back to the host. The data read will go into the ^FN1 location of the format. The printer will retry the command five times, if necessary.

```

^XA
^F020,120^A0N,60^FN1^FS
^RT1,,,5^FS
^HV1,,Tag Data:^FS
^XZ

```



Example 2 • This example reads from a tag twice and prints the results on a label.

```
^XA
^F020,120^A0N,60^FN1^FS
^F020,100^A0N,20^FN2^FS
^RT1,7,3,,5^FS
^RT2,3,2,,5^FS
^XZ
```

The first ^RT command starts at block 7 and reads three blocks of data in ASCII format. The data read will go into the ^FN1 location of the format. The printer will retry the command five times, if necessary.

The second ^RT command starts at block 2 and reads two blocks of data in ASCII format. The data read will go into the ^FN2 location of the format. The printer will retry the command five times, if necessary.

~RV

Report RFID Encoding Results



Important • This command is not supported by all printers or firmware. See [Printer and Firmware Compatibility on page 383](#) for the list of printers and firmware with which you can use this command.

Description Use this command to tell the printer to send RFID encoding success or failure results to the host computer after each label format completes.

Format ~RVa

This table identifies the parameters for this format:

Parameters	Details
a = enable/disable	Enables or disables the results reporting feature. <i>Accepted Values:</i> E = Enable D = Disable <i>Default Value:</i> D



Example 1 • Assume that the following code is sent and that there is no RFID tag in the field.

```
~RVE
^XA
^RS8,0,,3
^RMY
^RFR,H
^XZ
```

The printer attempts to program a tag three times and then returns the following to the host:

```
_-,3_
```

The minus sign indicates that the programming attempt failed entirely and voided three labels.



Example 2 • Assume that the same code is sent and that the first two attempts at programming a tag are unsuccessful. The third attempt succeeds.

```
~RVE
^XA
^RS8,0,,3
^RMY
^RFR,H
^XZ
```

The printer attempts to program a tag three times and then returns the following to the host:

```
_+,2_
```

The plus sign indicates that the programming attempt was successful and voided two labels.



Set RF Power Levels for Read and Write



Important • This command is not supported by all printers or firmware. See [Printer and Firmware Compatibility on page 383](#) for the list of printers and firmware with which you can use this command.

Description Use this command to set the RFID read and write power levels. This function is useful when using different tag types or transponders that require different power levels to obtain the best read and write abilities. If not enough power is applied, the transponder may not have sufficient power for programming, and tag data will fail to encode. If too much power is applied, the extra power may disable the tag may or cause data communication errors.

Format ^RW*r*,*w*

This table identifies the parameters for this format:

Parameters	Details
<i>r</i> = read power	Sets the power level to match the desired output as calibrated in the factory. R53.16.3 and later: <i>Values:</i> 0 to 30 <i>Default Value:</i> 16 R60.16.4, R62.16.4, R63.16.4, SP994Q, SP999G, SP1027G, SP1056F, SP1082G, and later: <i>Values:</i> 0 to 30, H (high), M (medium), L (low) <i>Default Value:</i> L R65.X and older versions of other firmware: <i>Accepted Values:</i> H = high M = medium L = low <i>Default Value:</i> L

Parameters	Details
w = write power	 Note • This parameter is ignored on the R110Xi HF printer 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. Sets the power level to match the desired output as calibrated in the factory. R53.16.3 and later: <i>Values:</i> 0 to 30 <i>Default Value:</i> 16 R60.16.4, R62.16.4, R63.16.4, SP994Q, SP999G, SP1027G, SP1056F, SP1082G, and later: <i>Values:</i> 0 to 30, H (high), M (medium), L (low) <i>Default Value:</i> L R65.X and older versions of other firmware: <i>Accepted Values:</i> H = high M = medium L = low <i>Default Value:</i> L
a = RFID antenna selection	 Note • This parameter applies only to the R110Xi HF printer. Selects the antenna port that provides the best results for reading and writing. <i>Accepted Values:</i> 1 = antenna port 1 2 = antenna port 2 <i>Default Value:</i> 1



Set RFID Tag Password and Lock Tag

Description Use this command to define a password for a tag during writing.

With Gen 2 tags, you can lock a tag's memory bank with an access password or define a kill password that can be used to permanently disable the tag. If you do not set access or kill passwords for a tag, the tag ignores any read or write commands that try to use these functions.



Note • The printer can set a kill password, but the printer cannot kill a tag.

Format ^RZp,m,l

The following table identifies the parameters for this format.

Parameters	Details
p = password	Sets a password. For tag types other than Gen 2: The password is 8 bits. The memory bank and lock style parameters are ignored. The password must be 2 hexadecimal characters long. <i>Accepted Values:</i> 00 to FF (hexadecimal) <i>Default Value:</i> 00 For Gen 2 tag type only: Gen 2 tags use a 32-bit password and specify the memory bank and lock style. The password must be 8 hexadecimal characters long. Use ^RF on page 395 to read the passwords. <i>Accepted Values:</i> 00000000 to FFFFFFFF (hexadecimal) <i>Default Value:</i> none
m = memory bank	Note • This parameter applies to Gen 2 tags only. <i>Accepted Values:</i> - K = kill password - A = access password - E = EPC - T = tag identifier (TID) - U = user <i>Default Value:</i> none

Parameters	Details
<code>l</code> = lock style	 Note • This parameter applies to Gen 2 tags only. <i>Accepted Values:</i> U = unlocked L = locked O = permanently unlocked (open) P = permanently locked (protected) W = write value (used only for the kill password memory bank) <i>Default Value:</i> none

➔ **Example 1** • On a tag that is not Gen 2, this example encodes 5A as the tag password.

```
^XA
^RZ5A^FS
^XZ
```

➔ **Example 2** • On a Gen 2 tag, this example encodes EPC data 112233445566778899001122 to the tag in Hex format, write protects the tag's EPC data with password 1234ABCD, and leaves the tag's access password unlocked.

```
^XA
^RFW,H^FD112233445566778899001122^FS
^RZ1234ABCD,E,L^FS
^XZ
```

➔ **Example 3** • On a Gen 2 tag, this example encodes EPC data 112233445566778899001122 to the tag in Hex format, write protects the tag's EPC data with password 1234ABCD, and makes the tag's access password unreadable.

```
^XA
^RFW,H^FD112233445566778899001122^FS
^RZ1234ABCD,E,L^FS
^RZ1234ABCD,A,L^FS
^XZ
```

The following code unprotects EPC data 112233445566778899001122 using the password 1234ABCD, encodes EPC data `newdata` to the tag in ASCII format, and then write protects the tag's new EPC data. The access password and its lock state are not changed, so the access password remains unreadable.

```
^XA
^RZ1234ABCD,E,U^FS
^RFW,A^FDnewdata^FS
^RZ1234ABCD,E,L^FS
^XZ
```



Example 4 • On a Gen 2 tag, this example unlocks the locked access password from the previous example.

```
^XA  
^RZ1234ABCD, A, U^FS  
^XZ
```

^WF

Encode AFI or DSFID Byte



Important • This command is not supported by all printers or firmware. See [Printer and Firmware Compatibility on page 383](#) for the list of printers and firmware with which you can use this command.

Description Use this command to encode the AFI or DSFID byte to a tag. Error handling is set by the ^RS command.

Format ^WFr,m,w,f,b

The following table identifies the parameters for this format.

Parameters	Details
r = number of retries	<i>Accepted values:</i> 0 to 10 <i>Default value:</i> 0
m = motion	<i>Accepted values:</i> 0 = Feed label after writing. 1 = No Feed after writing. Other ZPL may cause a feed. <i>Default value:</i> 0
w = write protect	<i>Accepted values:</i> 0 = Not write protected 1 = Write protect <i>Default value:</i> 0
f = data format	<i>Accepted values:</i> 0 = ASCII 1 = Hexadecimal <i>Default value:</i> 0
b = type of byte to read	<i>Accepted values:</i> A = AFI byte D = DSFID byte <i>Default value:</i> A



Example 1 • This example encodes data “R” (hex 52) as the AFI byte. The printer will try the command up to five times, if necessary.

```
^XA
^WF5^FDR
^XZ
```

→ **Example 2** • This example encodes data hex 66 as the AFI byte. The printer will try the command up to three times, if necessary.

```
^XA  
^WF3, , , 1^FD66  
^XZ
```

→ **Example 3** • This example encodes data hex 77 as the DSFID byte. The printer will try the command up to four times, if necessary.

```
^XA  
^WF4, , , 1, D^FD77  
^XZ
```



Write (Encode) Tag



Note • The ^WT command is provided only for backward-compatibility with label formats that were developed for older Zebra RFID printers. See [Printer and Firmware Compatibility on page 383](#) for the list of printers and firmware with which you should not use this command.

Description Use this command to encode the current RFID tag. Check the amount of data memory available for the tag that you will be using. If you send more data than the memory can hold, the printer truncates the data.

Format ^WT**b**,*r*,*m*,*w*,*f*,*v*

This table identifies the parameters for this format.

Parameters	Details
b = block number	Specifies the block number to encode. This parameter is tag-dependent. - For most tags, use block 0. - For EPC Class 0 Plus, block 0 is EPC data, and block 1 is user data. - For the R4Mplus, this parameter does not apply to printers with firmware version SP902X. With other versions of firmware in this printer, you can encode 12 bytes (96 bits) to block 0, but you can encode only the first byte of block 1. <i>Accepted values:</i> 0 to <i>n</i>, where <i>n</i> is the maximum number of blocks for the tag. <i>Default value:</i> 0
r = number of retries	Changes the number of times that the printer attempts to encode a particular block of a single RFID tag. (Same function as the ^RR command.) <i>Accepted values:</i> 0 to 10 <i>Default value:</i> 0
m = motion	Enables or disables RFID motion. (Same function as the ^RM command.) <i>Accepted values:</i> 0 = Feed label after writing 1 = No feed after writing (other ZPL may cause a feed) <i>Default value:</i> 0
w = write protect	<i>Accepted values:</i> 0 = Not write protected 1 = Write protected <i>Default value:</i> 0

Parameters	Details
f = data format	<i>Accepted values:</i> 0 = ASCII 1 = Hexadecimal <i>Default value:</i> 0
For the R110Xi HF: v = reverse the data order	This parameter is not used in the R2844-Z. For the R110Xi HF: Reverses the data order. <i>Accepted values:</i> N = Do not reverse the data order (Most significant byte first for Tag*It and PicoTag. Last significant byte first for I*code and ISO 15693) Y = Reverse the data order <i>Default value:</i> N
For other supported printers: v = verify valid data	For other supported printers: For reliability, some manufacturers encode tags with known data (such as A5A5). This parameter flags whether the preprogrammed data is verified. (Same function as the ^WV command.) <i>Accepted values:</i> N = Do not verify Y = Verify valid data before writing <i>Default value:</i> N



Example • This sample encodes data “RFIDRFID” and will try writing up to five times, if necessary.

```

^XA
^WT,5^FDRFIDRFID^FS
^XZ

```



Verify RFID Encoding Operation



Important • This command is not supported by all printers or firmware. See [Printer and Firmware Compatibility on page 383](#) for the list of printers and firmware with which you can use this command.

Description Use this command to enable or disable the write verify function. When write verify is enabled, this command verifies the RFID encoding operation to ensure that the tag about to be programmed contains the hex data “A5A5” in the first two bytes. This parameter is not persistent (carried over from label to label).

Format ^WVe

This table identifies the parameters for this format.

Parameters	Details
e = enable	<i>Accepted values:</i> Y or N <i>Default value:</i> N



ZPL Wireless Commands

This section contains new or modified ZPL commands for the Wireless and Wireless Plus print servers.

^KC

Set Client Identifier (Option 61)

Description The ^KC command allows the print server to have its own client identifier (CID).

Format ^KCa, b, c, d

The following table identifies the parameters for this format.

Parameters	Details
a = enable or disable	<i>Accepted Values:</i> 0 = disable (default) 1 = enabled, use MAC address 2 = enabled, ASCII value 3 = enabled, HEX value <i>Default Value:</i> 0
b = device	<i>Accepted Values:</i> 0 = all devices 1 = wireless 2 = external wired ^a 3 = internal wired <i>Default Value:</i> 1
c = prefix (optional)	<i>Accepted Values:</i> 11 ASCII characters or 22 hexadecimal values. The prefix can be cleared by defaulting the network settings on the printer.
d = identifier	<i>Accepted Values:</i> 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.

^a. This applies only to the Xi4, 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.



Search for Wired Print Server during Network Boot

Description Use this command to tell the printer whether to search for a wired print server at bootup. This command is ignored on the Xi4, ZM400, ZM600, RZ400, and RZ600 printers.



Note • Only one print server can be installed in the S4M at one time, so this check does not occur.

Table 20 identifies which device becomes the active print server under different conditions.

Table 20 • 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 ^a	
Skip	X	X	Wireless
	X		Wired
		X	Wireless
Check	X	X	Wired
	X		Wired
		X	Wireless

a. A wireless option board must have an active radio that can properly associate to an access point.

Format ^NBa

The following table identifies the parameters for this format.

Parameters	Details
a = check for wired print server at boot time	<i>Accepted Values:</i> C (CHECK), S (SKIP CHECK) <i>Default Value:</i> S

^NN

Set SNMP

Description Use this command to set the Simple Network Management Protocol (SNMP) parameters.

Format ^NNa,b,c,d,e,f

The following table identifies the parameters for this format.

Parameters	Details
a = system name	Same as printer name. <i>Accepted Values:</i> Up to 17 alphanumeric characters
b = system contact	Any contact information as desired (such as a name or phrase) <i>Accepted Values:</i> Up to 50 alphanumeric characters
c = system location	The printer's model information. <i>Accepted Values:</i> Up to 50 alphanumeric characters
d = get community name	<i>Accepted Values:</i> Up to 19 alphanumeric characters <i>Default Value:</i> public
e = set community name	<i>Accepted Values:</i> Up to 19 alphanumeric characters <i>Default Value:</i> public
f = trap community name	<i>Accepted Values:</i> Up to 20 alphanumeric characters <i>Default Value:</i> public

^NP

Set Primary/Secondary Device

Description 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.

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 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 427).

Format ^NP`a`

The following table identifies the parameters for this format.

Parameters	Details
<code>a</code> = device to use as primary	<i>Accepted Values:</i> P = PRINTER M = MPS/PRINTSERVER <i>Default Value:</i> P

^NT

Set SMTP

Description Use this command to set the Simple Mail Transfer Protocol (SMTP) parameters. This allows you to set the e-mail settings for alerts.

Format ^NTa,b

The following table identifies the parameters for this format.

Parameters	Details
a = SMTP server address	<i>Accepted Values:</i> Any properly formatted server address in the xxx.xxx.xxx.xxx format
b = print server domain	<i>Accepted Values:</i> 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



Set Web Authentication Timeout Value

Description 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.

Format ^NWa

The following table identifies the parameters for this format.

Parameters	Details
a = timeout value	The timeout value in minutes for an IP address to be authenticated to the printer web pages. <i>Accepted Values:</i> 0 (no secure pages can be accessed without entering the printer password) to 255 minutes <i>Default Value:</i> 5

^WA

Set Antenna Parameters

Description Use this command to set the values for the receive and transmit antenna.

Format ^WAAa, b

The following table identifies the parameters for this format.

Parameters	Details
a = receive antenna	Accepted Values: D (Diversity), L (Left), R (Right) Default Value: D
b = transmit antenna	Accepted Values: D (Diversity), L (Left), R (Right) Default Value: D

^WE

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 445 to set the security type and related parameters.

Description 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).

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

The following table identifies the parameters for this format.

Parameters	Details
a = encryption mode	<i>Accepted Values:</i> OFF, 40 (40-bit encryption), 128 (128-bit encryption) <i>Default Value:</i> OFF
b = encryption index	Tells the printer which encryption key to use. <i>Accepted Values:</i> 1 (Key 1), 2 (Key 2), 3 (Key 3), 4 (Key 4) <i>Default Value:</i> 1
c = authentication type	<i>Accepted Values:</i> O (Open System), S (Shared Key) <i>Default Value:</i> O  Note • If you enable Shared Key authentication with Encryption Mode set to OFF, this value resets to Open.
d = encryption key storage	<i>Accepted Values:</i> H (Hex key storage), S (string key storage) <i>Default Value:</i> H
e, f, g, h = encryption keys 1 through 4	<i>Accepted Values:</i> 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 1** • 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 2** • 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 3** • 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.



Important • Make sure that you include the exact number of commas required to get to the slot for encryption key 4 (parameter h).



Change Wireless Network Settings

Description Use this command to change the wireless network settings.

Zebra recommends using [^ND on page 310](#) for these settings if you have any of the following printers:

- Xi4 printers with firmware V53.17.1Z or later
- ZM400, ZM600, or S4M printers with firmware V53.15.xZ or later
- RZ400/RZ600 printers with firmware R53.16.x or later

Format ^WIa,b,c,d,e,f,g,h,i

The following table identifies the parameters for this format.

Parameters	Details
a = IP resolution	<i>Accepted Values:</i> 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 <i>Default Value:</i> A
b = IP address	<i>Accepted Values:</i> Any properly formatted IP address in the xxx.xxx.xxx.xxx format. <i>Default Value:</i> 000.000.000.000
c = subnet mask	<i>Accepted Values:</i> Any properly formatted subnet mask in the xxx.xxx.xxx.xxx format. <i>Default Value:</i> 000.000.000.000
d = default gateway	<i>Accepted Values:</i> Any properly formatted gateway in the xxx.xxx.xxx.xxx format. <i>Default Value:</i> 000.000.000.000
e = WINS server address	<i>Accepted Values:</i> Any properly formatted WINS server in the xxx.xxx.xxx.xxx format. <i>Default Value:</i> 000.000.000.000
f = connection timeout checking	<i>Accepted Values:</i> Y = yes N = no <i>Default Value:</i> Y

Parameters	Details
g = timeout value	Time, in seconds, before the connection times out. <i>Accepted Values:</i> 0 through 9999 <i>Default Value:</i> 300
h = ARP broadcast interval	Time, in minutes, that the broadcast is sent to update the device's ARP cache. <i>Accepted Values:</i> 0 through 30 <i>Default Value:</i> 0 (no ARP sent)
i = base raw port number	The port number that the printer should use for its RAW data. <i>Accepted Values:</i> 1 through 65535 <i>Default Value:</i> 9100



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 445 to set the security type and related parameters.

Description 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.

Format ^WLa, b, c

The following table identifies the parameters for this format.

Parameters	Details
a = mode	<i>Accepted Values:</i> OFF, ON <i>Default Value:</i> OFF
b = user name	<i>Accepted Values:</i> Any 1 to 32 alphanumeric including special characters <i>Default Value:</i> user
c = password	<i>Accepted Values:</i> Any 1 to 32 alphanumeric including special characters <i>Default Value:</i> password

~WL

Print Network Configuration Label

Description Use this command to generate a network configuration label (Figure 5).

Format ~WL

Figure 5 • Network Configuration Label

Wireless Print Server

Network Configuration	
Zebra Technologies ZTC 140XiIIIPlus-200dpi ZBR325B042	
NO.....	WIRED PS CHECK?
Printer.....	LOAD LAN FROM?
Wired	
ALL.....	IP PROTOCOL
000.000.000.000.....	IP ADDRESS
000.000.000.000.....	SUBNET MASK
000.000.000.000.....	DEFAULT GATEWAY
000.000.000.000.....	WINS SERVER IP
YES.....	TIMEOUT CHECKING
0300.....	TIMEOUT VALUE
0000.....	ARP INTERVAL
9100.....	BASE RAW PORT
Wireless*	
ALL.....	IP PROTOCOL
010.003.015.030.....	IP ADDRESS
255.255.255.000.....	SUBNET MASK
000.000.000.000.....	DEFAULT GATEWAY
000.000.000.000.....	WINS SERVER IP
YES.....	TIMEOUT CHECKING
0300.....	TIMEOUT VALUE
0000.....	ARP INTERVAL
9100.....	BASE RAW PORT
YES.....	CARD INSERTED
015FH.....	CARD MFG ID
000AH.....	CARD PRODUCT ID
5.02.19.....	CARD FIRMWARE
000B2131b6ba.....	MAC ADDRESS
YES.....	DRIVER INSTALLED
INFRASTRUCTURE.....	OPERATING MODE
125.....	ESSID
100.....	TX POWER
ON.....	1 Mb/s
ON.....	2 Mb/s
ON.....	5.5 Mb/s
ON.....	11 Mb/s
11 Mb/s.....	CURRENT TX RATE
DIVERSITY.....	RECEIVE ANTENNA
DIVERSITY.....	XMIT ANTENNA
YES.....	ASSOCIATED
NONE.....	WLAN SECURITY
OPEN.....	WEP TYPE
1.....	WEP INDEX
LONG.....	PREAMBLE
020.....	POOR SIGNAL

FIRMWARE IN THIS PRINTER IS COPYRIGHTED

ZM and RZ Printers

Network Configuration	
Zebra Technologies PRINTER NAME ZBR2834792	
0.0.0 *.....	OPTION FIRMWARE
Wired.....	PRIMARY NETWORK
NO.....	LOAD FROM EXT?
Internal Wired.....	ACTIVE PRINTSRVR
External Wired	
ALL.....	IP PROTOCOL
000.000.000.000.....	IP ADDRESS
255.255.255.000.....	SUBNET MASK
000.000.000.000.....	DEFAULT GATEWAY
000.000.000.000.....	WINS SERVER IP
YES.....	TIMEOUT CHECKING
0300.....	TIMEOUT VALUE
0000.....	ARP INTERVAL
9100.....	BASE RAW PORT
Internal Wired*	
ALL.....	IP PROTOCOL
010.003.004.116.....	IP ADDRESS
255.255.255.000.....	SUBNET MASK
010.003.004.001.....	DEFAULT GATEWAY
010.003.001.098.....	WINS SERVER IP
YES.....	TIMEOUT CHECKING
300.....	TIMEOUT VALUE
000.....	ARP INTERVAL
9100.....	BASE RAW PORT
00074d2b416b.....	MAC ADDRESS
Wireless	
ALL.....	IP PROTOCOL
000.000.000.000.....	IP ADDRESS
255.255.255.000.....	SUBNET MASK
000.000.000.000.....	DEFAULT GATEWAY
000.000.000.000.....	WINS SERVER IP
YES.....	TIMEOUT CHECKING
300.....	TIMEOUT VALUE
000.....	ARP INTERVAL
9100.....	BASE RAW PORT
NO.....	CARD INSERTED
0000H.....	CARD MFG ID
0000H.....	CARD PRODUCT ID
000000000000.....	MAC ADDRESS
YES.....	DRIVER INSTALLED
INFRASTRUCTURE.....	OPERATING MODE
125.....	ESSID
100.....	TX POWER
ON.....	1 Mb/s
ON.....	2 Mb/s
ON.....	5.5 Mb/s
ON.....	11 Mb/s
11 Mb/s.....	CURRENT TX RATE
DIVERSITY.....	RECEIVE ANTENNA
DIVERSITY.....	XMIT ANTENNA
OPEN.....	WEP TYPE
NONE.....	WLAN SECURITY
1.....	WEP INDEX
LONG.....	PREAMBLE
020.....	POOR SIGNAL
ON.....	PULSE ENABLED
15.....	PULSE RATE
OFF.....	INTL MODE
07FFH.....	CHANNEL MASK

FIRMWARE IN THIS PRINTER IS COPYRIGHTED

Wireless Plus Print Server and Internal Wireless Plus Print Server

Other Printers/Print Engines

Network Configuration	
Zebra Technologies ZTC 105SL-200dpi ZBR12908662	
1.10.1729.....	OPTION FIRMWARE
NO.....	WIRED PS CHECK?
Printer.....	LOAD LAN FROM?
Wired	
ALL.....	IP PROTOCOL
000.000.000.000.....	IP ADDRESS
000.000.000.000.....	SUBNET MASK
000.000.000.000.....	DEFAULT GATEWAY
000.000.000.000.....	WINS SERVER IP
YES.....	TIMEOUT CHECKING
0300.....	TIMEOUT VALUE
0000.....	ARP INTERVAL
9100.....	BASE RAW PORT
Wireless*	
ALL.....	IP PROTOCOL
010.003.015.034.....	IP ADDRESS
255.255.255.000.....	SUBNET MASK
000.000.000.000.....	DEFAULT GATEWAY
000.000.000.000.....	WINS SERVER IP
YES.....	TIMEOUT CHECKING
0300.....	TIMEOUT VALUE
0000.....	ARP INTERVAL
9100.....	BASE RAW PORT
YES.....	CARD INSERTED
015FH.....	CARD MFG ID
000AH.....	CARD PRODUCT ID
5.60.21.....	CARD FIRMWARE
000e84c4f876.....	MAC ADDRESS
YES.....	DRIVER INSTALLED
INFRASTRUCTURE.....	OPERATING MODE
125.....	ESSID
100.....	TX POWER
ON.....	1 Mb/s
ON.....	2 Mb/s
ON.....	5.5 Mb/s
ON.....	11 Mb/s
11 Mb/s.....	CURRENT TX RATE
DIVERSITY.....	RECEIVE ANTENNA
DIVERSITY.....	XMIT ANTENNA
YES.....	ASSOCIATED
NONE.....	WLAN SECURITY
OPEN.....	WEP TYPE
1.....	WEP INDEX
LONG.....	PREAMBLE
020.....	POOR SIGNAL
ON.....	PULSE ENABLED
15.....	PULSE RATE
ON.....	INTL MODE
07FFH.....	CHANNEL MASK

FIRMWARE IN THIS PRINTER IS COPYRIGHTED

Network Configuration	
Zebra Technologies ZTC 220XiIIIPlus-200dpi ZBR3051375	
Wired.....	PRIMARY NETWORK
NO.....	LOAD FROM EXT?
Internal Wired.....	ACTIVE PRINTSRVR
External Wired	
ALL.....	IP PROTOCOL
000.000.000.000....	IP ADDRESS
255.255.255.000....	SUBNET MASK
000.000.000.000....	DEFAULT GATEWAY
000.000.000.000....	WINS SERVER IP
YES.....	TIMEOUT CHECKING
300.....	TIMEOUT VALUE
000.....	ARP INTERVAL
9100.....	BASE RAW PORT
Internal Wired*	
ALL.....	IP PROTOCOL
010.003.004.098....	IP ADDRESS
255.255.255.000....	SUBNET MASK
010.003.004.001....	DEFAULT GATEWAY
010.003.001.098....	WINS SERVER IP
YES.....	TIMEOUT CHECKING
300.....	TIMEOUT VALUE
000.....	ARP INTERVAL
9100.....	BASE RAW PORT
00074d2e8f6f.....	MAC ADDRESS
Wireless	
ALL.....	IP PROTOCOL
000.000.000.000....	IP ADDRESS
255.255.255.000....	SUBNET MASK
000.000.000.000....	DEFAULT GATEWAY
000.000.000.000....	WINS SERVER IP
YES.....	TIMEOUT CHECKING
300.....	TIMEOUT VALUE
000.....	ARP INTERVAL
9100.....	BASE RAW PORT
NO.....	CARD INSERTED
0000H.....	CARD MFG ID
0000H.....	CARD PRODUCT ID
000000000000.....	MAC ADDRESS
YES.....	DRIVER INSTALLED
INFRASTRUCTURE.....	OPERATING MODE
125.....	ESSID
100.....	TX POWER
ON.....	1 Mb/s
ON.....	2 Mb/s
ON.....	5.5 Mb/s
ALL.....	ALL



Set Wireless Password



Note • This command does not apply to the S4M.

Description 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.

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 ^WP*a*,*b*

The following table identifies the parameters for this format.

Parameters	Details
<i>a</i> = old wireless password	<i>Accepted Values:</i> 0000 through 9999 <i>Default Value:</i> 0000
<i>b</i> = new wireless password	<i>Accepted Values:</i> 0000 through 9999 <i>Default Value:</i> 0000



Set Transmit Rate

Description Use this command to change the transmission parameters.

Format ^WRa,b,c,d,e

The following table identifies the parameters for this format.

Parameters	Details
a = rate 1	Sets the 1 Mb/s transmit rate. <i>Accepted Values:</i> Y (On), N (Off)
b = rate 2	Sets the 2 Mb/s transmit rate. <i>Accepted Values:</i> Y (On), N (Off)
c = rate 5	Sets the 5.5 Mb/s transmit rate. <i>Accepted Values:</i> Y (On), N (Off)
d = rate 11	Sets the 11 Mb/s transmit rate. <i>Accepted Values:</i> Y (On), N (Off)
e = transmit power	<i>Accepted Values:</i> 1, 5, 20, 30, 50, 100

~WR

Reset Wireless Radio Card and Print Server

Description 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.

Format ~WR



Set Wireless Radio Card Values

Description Use this command to set the wireless radio card values for ESSID, Operating Mode, and Card Preamble.

Format ^WSe,o,p,h,i,j,k

The following table identifies the parameters for this format.

Parameters	Details
e = ESSID value	<i>Accepted Values:</i> 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. <i>Default Value:</i> 125
o = operating mode	<i>Accepted Values:</i> I (Infrastructure), A (Adhoc) <i>Default Value:</i> I
p = wireless radio card preamble	<i>Accepted Values:</i> L = long S = short <i>Default Value:</i> L
h = wireless pulse  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. <i>Accepted Values:</i> 0 = disabled 1 = enabled <i>Default Value:</i> 1
i = wireless pulse interval  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. <i>Accepted Values:</i> 5 to 300 seconds <i>Default Value:</i> 15

Parameters	Details
j = channel mask  This parameter is supported in firmware version X60.15.x, V50.15.x, or later.	For commonly used channel masks, see Table 21 . <i>Accepted Values:</i> 4 Hexadecimal digits preceded by “0x” (0x0000 to 0xFFFF) <i>Default Value:</i> 0x7FF
k = international mode  This parameter is supported in firmware version X60.15.x, V50.15.x, or later.	<i>Accepted Values:</i> 0 (Disabled), 1 (Enabled) <i>Default Value:</i> 0

Table 21 • Channel Mask Settings

Region	Channel Mask
United States, Canada, Latin America	0x7FF
Europe, Middle East, Africa, other	0x1FFF
Japan	0x3FFF



Configure Wireless Securities

Description 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.



This command applies to printers with firmware version X60.15.x, V53.15.x, V50.15.x, or later.



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 ^WX command replaces individual ZPL commands for different security types.

Format ^WXa,[zero or more supporting parameters]



Note • The supporting parameters that are required vary based on the security type that you select. See [Supporting Parameters for Different Security Types on page 449](#) for instructions for each security type.

The following table identifies the parameters for this format.

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 Different Security Types on page 449.  Note • Configuring the printer for WPA also allows the printer to be used in WPA2 environments. <i>Accepted Values:</i> 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-F AST 07 = PEAP 08 = LEA P 09 = WPA PSK (R6x15.x, R53.15.x, ZSPx, and later.) 10 = WP A EAP-TLS 11 = WP A EAP-TTLS 12 = WP A EAP-FAST 13 = WP A PEAP 14 = WP A LEAP 15 = Kerberos <i>Default Value:</i> 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. <i>Accepted Values:</i> 1, 2, 3, 4 <i>Default Value:</i> 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. <i>Accepted Values:</i> 0 or S 0 = open system - S = shared key <i>Default Value:</i> 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. <i>Accepted Values:</i> H or S - H = hex key storage - S = string key storage <i>Default Value:</i> S

* Not used for all security types

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.  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. <i>Accepted Values:</i> The actual value for the encryption key <i>Default Value:</i> 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 EAP • WPA EAP-TTLS • Kerberos <i>Accepted Values:</i> The actual value for the user ID. <i>Default Value:</i> user

* Not used for all security types

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 EAP • WPA EAP-TTLS • Kerberos <i>Accepted Values:</i> The actual value for the password. <i>Default Value:</i> 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 <i>Accepted Values:</i> The actual value for the optional private key. <i>Default Value:</i> None
l = realm*	Specifies the realm for security types that require it. A value must be specified if using Kerberos. <i>Accepted Values:</i> The actual value for the realm. <i>Default Value:</i> kerberos
m = Key Distribution Center (KDC)*	Specifies the KDC for security types that require it. A value must be specified if using Kerberos. <i>Accepted Values:</i> The actual value for the KDC. <i>Default Value:</i> 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.  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. <i>Accepted Values:</i> a minimum of 64 hexadecimal digits <i>Default Value:</i> 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, 0, H, A1B2C3D4F5, , ,
^JUS
^XZ
```

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, 0, H, 001122334455667788, 112233445566778899, 223344
556677889900, 334455667788990011
^JUS
^XZ
```

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
```



Notes • _____



ZBI Commands

This section explains the Zebra Basic Interpreter, its commands, descriptions, formats, and parameters.

Command and Function Reference Format

This section describes how commands and functions are presented in this document.

Command/Function NAME

Description This heading has a description of 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>,. The <A> and 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.

➔ **Example •** 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.

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.

Introduction to Zebra Basic Interpreter (ZBI)

What is ZBI and why is it for me?

ZBI is an "on-the-printer" programming 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). The following printers support the ZBI 1.x firmware:

- *LP/TLP 284x-Z and 384x-Z*
- *S300/S400/S500/S600*
- *Z4000/Z6000*
- *Z4M/Z6M*
- *Z4Mplus/Z6Mplus*
- *105SL*
- *PAX3*
- *XiII*
- *XiIII*

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. The following printers support the ZBI 2.x firmware:

- *XiIII Plus*
- Z4Mplus/Z6Mplus
- 105SL
- S4M
- PAX4
- ZM400/ZM600
- G-Series
- HC100
- Xi4
- LP/TLP2824+

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.

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:**

```
NEW
AUTONUM 10,5
10 REM "Hello World" Application
15 PRINT "Hello World" ! comment...
20
LIST
10 REM "Hello World" Application
15 PRINT "Hello World"
NEW
LIST
```

Entered automatically when AUTONUM is used →

← A blank line stops AUTONUM

→ **Preview when viewed in ZBI-Developer**

```
AUTONUM 10,5
REM "Hello World" Application
PRINT "Hello World" ! comment...
```

NEW

Description 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

Description 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)

Description 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

Description 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
 = 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

Description This command automatically generates sequential program line numbers.

Format AUTONUM <A>,

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

Description 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>,

Parameters

<A> = the number to start the renumbering sequence

 = 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

Description 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

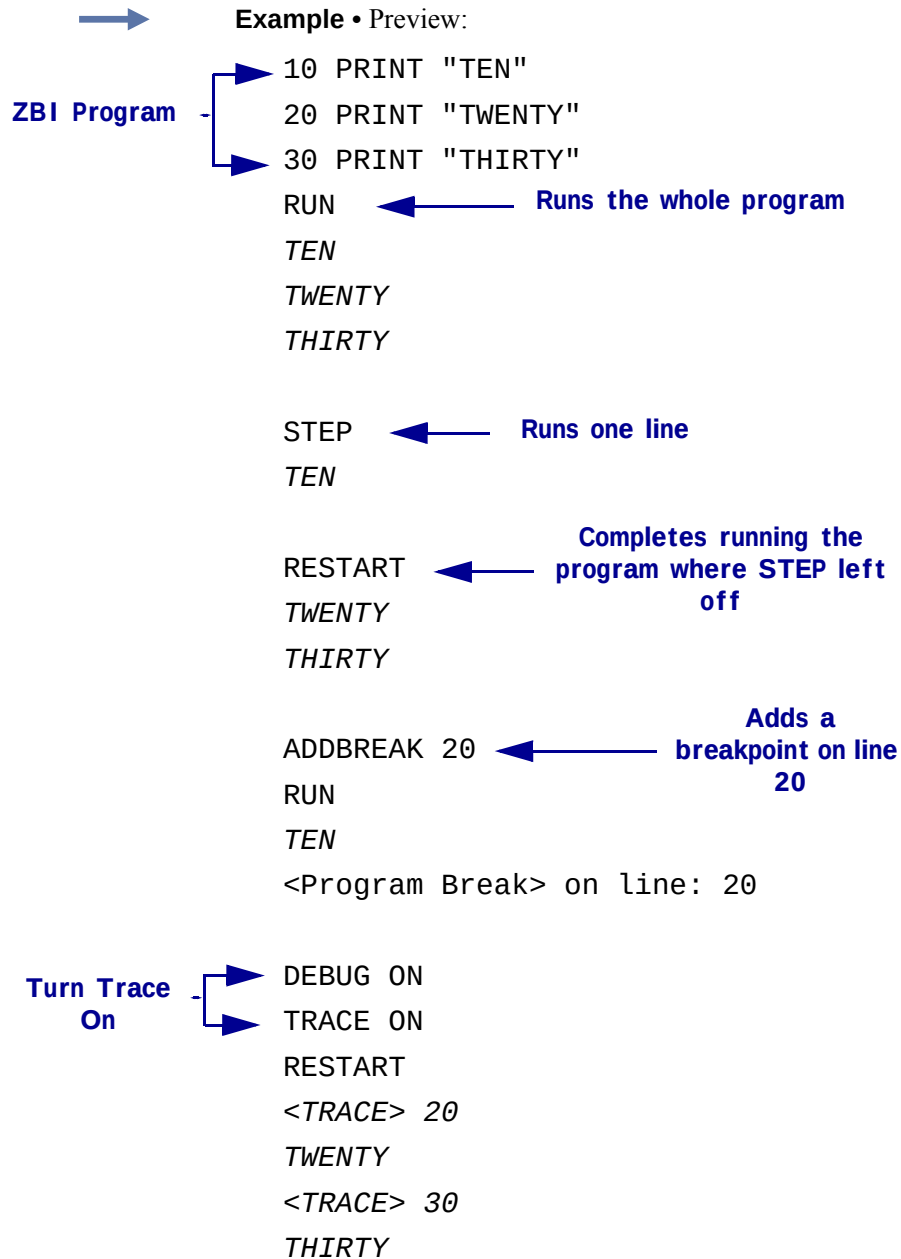
BREAK – Stops the currently running program

ADDBREAK – Adds a break to an existing line

DELBREAK – Deletes an existing break

ZPL Terminates and exits the ZBI environment.

This example shows many of the Running and Debug Commands in practice.



RUN

Description 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

Description 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

Description 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

Description 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

Description 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 • See [TRACE on page 477](#) and [BREAK on page 478](#).

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

Description 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

Description 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



Description 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



Description This command allows you to remove existing breakpoints.

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

Description 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

Description 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. See [Assignment on page 484](#).

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:
 $\langle A \rangle + \langle B \rangle$
 $5+2$ result = 7
 $\text{VAL}("25") + 2$ result = 27
2. - (subtraction) Subtraction expressions use this format:
 $\langle A \rangle - \langle B \rangle$
 $5-2$ result = 3
 $\text{VAL}("25") - 2$ result = 23
3. * (multiplication) Multiplication expressions use this format:
 $\langle A \rangle * \langle B \rangle$
 $5*2$ result = 10
 $\text{VAL}("25") * 2$ result = 50
4. / (division) Division expressions use this format:
 $\langle A \rangle / \langle B \rangle$
 $5/2$ result = 2
 $\text{VAL}("25") / 2$ result = 12
5. ^ (exponentiation) Exponentiation expressions use this format:
 $\langle A \rangle ^ \langle B \rangle$
 5^2 result = 25
 $\text{VAL}("25") ^ 2$ 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:

Functions
Parenthetical Expressions ()
^
* and /
+ 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

Description 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

 = 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
```

Example • 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

Description 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:

1. < (less than)

Expression	Result
1< 2	true
2<2 fal	se
2<1	false

2. <= (less than or equal to)

Expression	Result
1<=2	true
2<=2	true
2<=1	false

3. > (greater than)

Expression	Result
1> 2	false
2>2 fal	se
2>1	true

4. >= (greater than or equal to)

Expression	Result
1>=2	false
2>=2	true
2>=1	true

5. = (equal to)

Expression	Result
1=2	false
2=2	true
"A"="AA "	false
"A"="A "	true

6. <> (not equal to)

Expression	Result
1<>2	true
2<>2	false
"A"<>"AA "	true
"A"<>"A "	false

Combined Boolean Expressions

AND, OR, and NOT can be used in conjunction with base Boolean expressions to recreate expanded Boolean expressions.

1. NOT — Negate the target expression.

Expression	Result
NOT 1=2	true
NOT 1=1	false

2. 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

3. 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:

Expressions and Functions
Parenthetical expressions ()
<, <=, <>, =, =>, >
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 GOSUBs and GOTOs.

EXIT Used to exit the DO and FOR loops.

END Terminates any program currently running.

IF Statements

Description 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-LOOPS (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$="0" THEN
20 PRINT "ZBI IS FUN"
30 ELSE IF A$="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

Description 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

Description 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

 = 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

Description 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



Description 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

Description 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

Description 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 outlines 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 quick 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

Description 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

Accepted Values: 0 to 9

Default Value: a port must be specified

<PORT\$> = port name to open . See [Available Ports on page 500](#).



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

Description 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

Accepted 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:

```
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

Description 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



Description 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 520](#) and [UDP Server on page 522](#).

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



Description 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



Description 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 520](#) and [UDP Server on page 522](#).

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



Description 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 520](#) and [UDP Server on page 522](#).

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

Description 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 • The INPUT command does not accept control characters or the delete character. If these characters need to be processed, please 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

Description 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

Description 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.

Accepted 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

Description 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$>
```

Parameters

<CHANNEL> = reads from this port. Default = 0.

<A> = integer value is set to the byte received.

<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



Description 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



Description 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\$

Description 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)

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" or "USB" depending on the application.

```
10 CLOSE ALL
20 LET INPORT = 1
30 OPEN #INPORT: NAME "SER"
40 PRINT #INPORT: "Enter your name:";
50 INPUT #INPORT: YOURNAME$
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$)
90 ON ERROR GOTO 70
100 LET PACKET_SIZE = READ(INPORT,PACKET$,2000)
110 PRINT #ZPLPORT: "^XA^F0100,100^A0N,40,40^FDPACKET
FROM:";
115 PRINT #ZPLPORT: CLIENT_INFO$; "^FS"
120 PRINT #ZPLPORT: "^F0100,150^A0N,40,40^FDPACKET
SIZE:";
125 PRINT #ZPLPORT: PACKET_SIZE; "^FS"
130 PRINT #ZPLPORT: "^F0100,200^A0N,40,40^FDPACKET
DATA:";
135 PRINT #ZPLPORT: PACKET$; "^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 • Here is an 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^F0100,100^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

Description 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

Description 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

Description 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\$>]

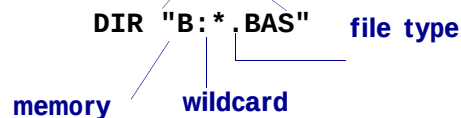
Parameters [<filter\$>] = the name of the file to be accessed (optional). Drive location and file name must be in quotation marks.

Default = "* : * .bas"



Important • Quotes must be around what you are doing. This shows you how to use the wildcard (*) to search for all .BAS files in B: memory:

Quotes surrounding action



Example N/A

Comments This is an interactive command that takes effect as soon as it is received by the printer.

DELETE

Description 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)

Accessing Comma Separated Value (CSV) and Text File Functions

This section describes the functions to access CSV files and ASCII plain-text files. Here is a quick list of these commands:

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



Description 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



Description 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



Description 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



Description 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's 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 front 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 front 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

ZBI Event ID	ZBI Event
18	spare unused
19	previous key
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
55	clean printhead clear
56	media low set
57	media low clear
58	ribbon low set

ZBI Event ID	ZBI Event
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

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.

Xi4/XiiiPlus/PAX4/105SL

<i>Xi4/XiiiPlus/PAX4</i> Front Panel Key	<i>105SL</i> 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		pause key
Feed		feed key
Eject		eject key

ZM400/ZM600/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
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

2824 Plus Series

Front Panel Key	ZBI Event ID	ZBI Name
Feed key	3	Feed key

KR403

Front Panel Key	ZBI Event ID	ZBI Name
Feed key	3	Feed key

REGISTEREVENT



Description 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.



Important • 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: "^F030,30^A0N,50,50^FDZebra Technologies^FS"
PRINT #ZPLPORT: "^F030,85^A0N,35,35^FDwww.zebra.com^FS"
PRINT #ZPLPORT: "^F030,125^A0N,35,35^FDsupport.zebra.com^FS"
PRINT #ZPLPORT: "^F030,165^A0N,35,35^FDVersion: "
PRINT #ZPLPORT: GETVAR$("appl.name") & "^FS"
PRINT #ZPLPORT: "^F030,205^A0N,35,35^FDPrinter Unique ID:"
PRINT #ZPLPORT: GETVAR$("device.unique_id") & "^FS"
PRINT #ZPLPORT: "^F030,245^A0N,35,35^FDActive Network: "
PRINT #ZPLPORT: GETVAR$("ip.active_network") & "^FS"
PRINT #ZPLPORT: "^F030,285^A0N,35,35^FDZBI Memory Usage: "
PRINT #ZPLPORT: GETVAR$("zbi.start_info.memory_alloc") & "^FS"
PRINT #ZPLPORT: "^F030,325^A0N,35,35^FDODometer: "
PRINT #ZPLPORT: GETVAR$("odometer.total_print_length") & "^FS"
PRINT #ZPLPORT: "^XZ"
```

Comments None

UNREGISTEREVENT



Description 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



Description 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: "^F030,30^A0N,50,50";
PRINT #ZPLPORT: "^FDZebra Technologies^FS"
```

```
PRINT #ZPLPORT: "^F030,85^A0N,35,35";
PRINT #ZPLPORT: "^FDwww.zebra.com^FS"
PRINT #ZPLPORT: "^F030,125^A0N,35,35";
PRINT #ZPLPORT: "^FDsupport.zebra.com^FS"
PRINT #ZPLPORT: "^F030,165^A0N,35,35";
PRINT #ZPLPORT: "^FDFW Version: ";
PRINT #ZPLPORT: GETVAR$("appl.name") & "^FS"
PRINT #ZPLPORT: "^F030,205^A0N,35,35";
PRINT #ZPLPORT: "^FDPrinter Unique ID:";
PRINT #ZPLPORT: GETVAR$("device.unique_id") & "^FS"
PRINT #ZPLPORT: "^F030,245^A0N,35,35";
PRINT #ZPLPORT: "^FDActive Network: ";
PRINT #ZPLPORT: GETVAR$("ip.active_network") & "^FS"
PRINT #ZPLPORT: "^F030,285^A0N,35,35";
PRINT #ZPLPORT: "^FDZBI Memory Usage: ";
PRINT #ZPLPORT: GETVAR$("zbi.start_info.memory_alloc") &
"^FS"
PRINT #ZPLPORT: "^F030,325^A0N,35,35";
PRINT #ZPLPORT: "^FD0dometer: ";
PRINT #ZPLPORT: GETVAR$("odometer.total_print_length") &
"^FS"
PRINT #ZPLPORT: "^XZ"
```

Comments None

TRIGGEREVENT



Description 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/XiIIIPlus/PAX4/105SL on page 540](#)
- [105SL Front Panel Key on page 540](#)
- [ZM400/ZM600/Z4Mplus/Z6Mplus on page 541](#)
- [S4M on page 541](#)

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

Description 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

Description 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

Description 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

Description 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^SX0,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

Description 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 553](#).

Comments This is a program command that is preceded by a line number.

ON ERROR

Description 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
...
```

Example • See [TCP Server on page 520](#) or [UDP Server on page 522](#).

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



Description 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



Description 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 556.

Comments If this pin is not controlled via ZBI (power pin), this function will return -1. See [AUXPORT_STEALPIN](#) on page 556.

AUXPORT_GETPIN



Description 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 556](#).

Comments If this pin is not controlled via ZBI (power pin), this function will return -1. See [AUXPORT_STEALPIN on page 556](#).

AUXPORT_RELEASEPIN



Description 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 556.

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\$

Description 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\$

Description 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\$

Description 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\$

Description 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\$

Description 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



Description 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 (forexample, **DECLARE STRING MYARRAY\$(2, 3)**) will be filled from (0,0), then (0,1) up to (2,3).

SPLITCOUNT



Description 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\$

Description 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\$

Description 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.



Important • 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

Description 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

Description 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

Description 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\$

Description 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

Description 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

Description 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

Description 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

Description 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

Description 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\$



Description 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.

```
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)
```

1E0F4

```
PRINT INTTOHEX$(-5)
```

0

```
PRINT INTTOHEX$(-99)
```

0

Comments Negative values will be returned as 0.

HEXTOINT



Description 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



Description 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.
<COLUMNS> = new number of columns in a two dimensional array.



Example • This example shows how to change a one dimensional numeric array.

```
10 DECLARE NUMERIC SCORES(3)  
20 LET SCORES(1) = 85  
30 LET SCORES(2) = 92  
40 LET SCORES(3) = 98  
50 REDIM SCORES(2) ! Discard the last one
```



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



Description 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



Description 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



Description 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



Description 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



Description 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 606](#) 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\$

Description 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\$

Description 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
```

Comments None

DATE

Description 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
```

Comments None

TIME

Description 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.



This is an example of how to use the **TIME** command [assuming the time is one minute past midnight]:

Example •

```
10 PRINT TIME
RUN
60
```

Comments None

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, see the ZPL Manual, Set-Get-Do Command chapters, or 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



Description SETVAR allows the direct setting of printer parameters.

Format SETVAR (PARAM\$, VALUE\$)

Parameters

PARAM\$ = The printer parameter to set. (See the ZPL Manual for specific parameters.)

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\$



Description This function retrieves printer parameters.

Format GETVAR\$ (PARAM\$)

Parameters

PARAM\$ = the printer parameter to get. (See the ZPL Manual for specific parameters.)

Returns The value of the parameter. See the [SGD Printer Setting Commands on page 619](#) 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$(1,I))
NEXT I
OPEN #1: NAME "ZPL"
PRINT #1: "^XA"
FOR I = 1 TO SGDCOUNT
PRINT #1: "^F050, ";50*I;"^A0N,25,25^FD";SGDQUERY$(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: <http://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 then prompted to enter an address, which is then added to an array of all addresses entered. After the user enters a total of 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^F030,30^A0N,30,30^FD"&NAME$(INDEX)&"^FS";
1 rem *****
1 rem ampersand used to concatenate data into strings
1 rem *****
220   print #1: "^F030,70^A0N,30,30^FD"&ADDRESS$(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 CSV 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
```



Example • This is an example of a CSV program (continued).

```
*****
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
```




Example • This is an example of a CSV program (continued).

```
*****
***
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
```



Example • This is an example of a CSV program (continued).

```
*****
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) = "^F0"
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$(8) = "F0"
140 let find$(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$ = "^F0" or in$ = "F0" 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$;
500   end if
510 loop
1 rem *****
1 rem convert the ^F0 and ^LH commands from 200 to 300 dpi
1 rem *****
520 inbyte #1: a$
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 ^A@ 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$a$; ",";
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
1 rem 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:"^F030,30^A0N,30,30^FD"&extracted_data1$&"^FS";
190       print #2:"^F030,70^A0N,30,30^FD"&extracted_data2$&"^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$(1,"","")
180     print #2:"^F030,30^A0N,30,30^FD" &extracted_data1$& "^FS";
190     print #2:"^F030,70^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 or ZM400/ZM600. 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^F0100,100^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 : "^F056,47^A0N,69,58^FDThis weighs^FS";
1 rem *****
1 rem print weight on label; & character concatenates strings
1 rem *****
200   print # 1 : "^F056,150^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

```




SGD Printer Commands

This chapter provides a high-level overview of printer setting Set / Get / Do (SGD) commands. For printer support of these SGD commands, see [SGD Command Support on page 997](#).



SGD commands are available in printers with firmware version V60.16.2Z, V60.15.xZ, V50.15.xZ, V61.15.xZ, V56.15.xZ, V53.16.x, V53.15.2Z, 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.



Note • SGD commands must be terminated by a carriage return or a space and line feed.

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**

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)



Important • The setvar command and attributes must be specified in lower case.

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



Important • The printer settings and attributes must be specified in lower case.

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.

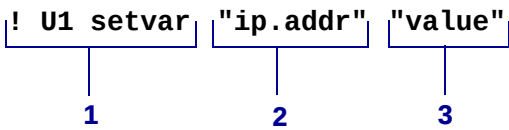


Important • The values must be specified in lower case.

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.

➔ **Example** • This is an example of a command structure illustration:



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).

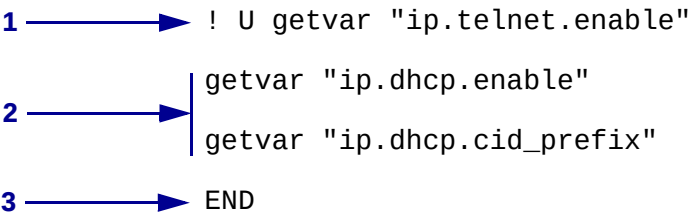


Note • Some RF cards do not support all of the SGD commands.

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.

➔ **Example** • This syntax shows how you can send multiple `getvar` commands:



1	The command portion of the string does not use the "1" after the "! U".
2	Commands issued after the first command do not require the "! U".
3	The string of commands is terminated by the word "END" with a space after the word, and by a carriage return/ line feed.

appl.option_board_version

Description This command returns the version number of the firmware running on the wireless option board. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command returns the version number of the firmware running on the wireless option board. <i>Format: ! U1 getvar "appl.option_board_version"</i>



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 *"
```

appl.bootblock

Description This command refers to the bootblock version. On the configuration label, the bootblock number is identified as the hardware ID. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command returns the bootblock version number that appears on the configuration label . <i>Format: ! U1 getvar "appl.bootblock"</i>



Example • In this example, the getvar returns the bootblock version number.

! U1 getvar "appl.bootblock"

appl.name

Description This command refers to the printer’s firmware version. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command returns the printer’s firmware version. <i>Format: ! U1 getvar "appl.name"</i>



Example • In this example, the `getvar` returns the printer’s firmware version.

```
! U1 getvar "appl.name"
```

CISDFCRC16

Download Files

The CISDFCRC16 command downloads supported files types to the printer. For printer support, see [SGD Command Support on page 997](#).



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.

Type

```
! CISDFCRC16
<crc>
<filename>
<size>
<checksum>
<data>
```



Note • For details on SGD command structure, see [Command Structure on page 628](#).



Important • Each line should be terminated with a CR/LF.



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.

This table identifies the parameters for this format:

Parameters	Details
<crc> = CRC value	<i>Accepted Values:</i> a four digit CRC value in hexadecimal. If 0000 is entered, then the CRC validation is ignored. For examples, see below.
<filename> = file name	<i>Accepted Values:</i> file name that is stored on the printer's file system. An extension must be specified. Files must be saved to the E: drive.
<size> = file size	<i>Accepted Values:</i> an eight digit file size specified in hexadecimal which indicates the number of bytes in the <data> section.

Parameters	Details
<checksum> = checksum value	<i>Accepted Values:</i> a four digit checksum value in hexadecimal. If 0000 is entered, then 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> = data	<i>Accepted Values:</i> Binary data saved on the printer's file system as <filename>. Number of bytes in this field must match the <size> parameter.



Example • 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/tHaU7A4SK08GeghX5hnp/mt4tuQEvsXkCrcgS1puz
z5db07ThhuzxYClnr7uiXPvSRXawgwDTPas+0q/6gHeUSXtA0EofuIyv7wIDAQAB
AoGBAJPnf3wn6wT5pE59DJIyakRiLmkt1wK0zv0bJfgS7i2Yv1EbeAy9PnPe3vKG
Bovm6A+oi2/qTSTLUTiFc7QHxJPVxLmRiHmbf1Q8j+VJkGTpwt8EY/Px+HSM2HAP
jqd+Im0IiE9RQPsxWQH9UaauF6nl5gIfMF74BIPsVzFXLffxAkEA6zSrCKCycE/P
14cjZibnLiWxdL3U3I9eWuhmIS37RB6UJFBCWUPWr26HlHz0KqhOUMbFf5h0mvkZ
gciN9A8kxwJBA0LK7Gyorre8iK9IMMwc70IJc7H8pH1y/N20tyaC1XuPfqz0H4PH
w2W2m3BhZ7ggHJLLiiFVF+Hr5X7cibFDo5kCQQDFe5lHSzXHWxvViN/N+0gL1RYk
Q0cisTW1+n8VyLe5wDr+Km0q6eytq44mvIuWAW6QH/TfZxBIynICKFQX4UctAkAm
P80iAkz9RfnTfhxjp7S35poxoYdodPU6tLAK+ZnhrfDSYJXUFuPYirSqfnMMtbw7
+EICnyRZAP0CqVU7pUm5AkeAnH206dKvUvw0EX+CscVATRrejKLCEJ+6YZWqiD9X
0XGJgrHNXGpDtQiVSGM59p0XnHTZJYjvVNdNOMnhg333nQ==
-----END RSA PRIVATE KEY-----
```



Examples • These are examples of CRC and checksum values:

CRC example

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.

checksum example

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.

This is the checksum byte.

cutter.clean_cutter

Description This command determines if the clean cutter option is enabled or disabled. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the status of the clean cutter option. <i>Format:</i> ! U1 getvar "cutter.clean_cutter"
setvar	This command instructs the printer to set the clean cutter option. <i>Format:</i> ! U1 setvar "cutter.clean_cutter" <i>Values:</i> "on" = turns on clean cutter "off" = turns off clean cutter <i>Default:</i> "on"



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".

device.download_connection_timeout

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the connection time out value (in seconds). <i>Format:</i> ! U1 getvar "device.download_connection_timeout"
setvar	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. <i>Format:</i> ! U1 setvar "device.download_connection_timeout" "value" <i>Values:</i> "0" through "65535" <i>Default:</i> "0" ("0" disables this feature)



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.friendly_name

Description This command shows the name assigned to the printer. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the name assigned to the printer. <i>Format:</i> ! U1 getvar "device.friendly_name"
setvar	This command sets the printer's name. <i>Format:</i> ! U1 setvar "device.friendly_name" "value" <i>Default:</i> "xxxxxxxxxx" ("xxxxxxxxxx" represents the main logic board serial number)



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.key_press

Description This command instructs the printer to press a button on the front panel. For printer support, see [SGD Command Support on page 997](#).

Type setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details	
setvar	This command instructs the printer to press a button on the front panel. <i>Format:</i> ! U1 setvar "device.frontpanel.key_press" <i>Values:</i> The values vary per printer, as follows:	
	ZM400 & Z4M/Z6M: "A" = Pause "B" = Feed "C" = Cancel "D" = Setup/Exit "E" = Minus "F" = Select "G" = Plus XiIIplus: "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: "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

Description 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. For printer support, see [SGD Command Support on page 997](#). For details on the supported character set, see [Character Set on page 994](#).

Type `getvar;setvar`



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the content that is shown on line one of the front panel. <i>Format:</i> <code>! U1 getvar "device.frontpanel.line1"</code>
setvar	This command instructs the printer to set the content that is shown on line one of the front panel. <i>Format:</i> <code>! U1 setvar "device.frontpanel.line1" "value"</code> <i>Values:</i> The maximum amount of alphanumeric ASCII characters available for line 1 on the printer's front panel <i>Default:</i> <code>" "</code>



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

Description 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 printer support, see [SGD Command Support on page 997](#). For details on the supported character set, see [Character Set on page 994](#).

Type `getvar;setvar`



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the content that shows on line two of the front panel. <i>Format:</i> <code>! U1 getvar "device.frontpanel.line2"</code>
setvar	This command instructs the printer to set the content that shows on line two of the front panel. <i>Format:</i> <code>! U1 setvar "device.frontpanel.line2" "value"</code> <i>Values:</i> The maximum amount of alphanumeric ASCII characters available for line two on the printer's front panel <i>Default:</i> <code>" "</code>



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

Description This command retrieves the current content of the front panel in an XML format. For printer support, see *SGD Command Support* on page 997.

Type getvar



Note • For details on SGD command structure, see *Command Structure* on page 628.

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the file that determines the representation of the front panel. <i>Format:</i> ! 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.jobs_print

Description This command identifies the number of jobs to be printed. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the number of jobs to be printed. <i>Format: ! U1 getvar "device.jobs_print"</i>



Example • In this example, the `getvar` retrieves the jobs currently being printed or last printed.

```
! U1 getvar "device.jobs_print"
```

```
"1"
```

device.languages

Description This command identifies the programming language that the printer is currently using. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- Printers running V60.15.8Z, V53.15.2Z, and later.

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the programming language that the printer is currently using. Table 22 shows the possible response values. <i>Format: ! U1 getvar "device.languages"</i>
setvar*	This command instructs the printer to set the printer to the required programming language . <i>Format: ! U1 setvar "device.languages"</i> <i>Values:</i> "ep1" = Eltron Programming Language "zpl" = Zebra Programming Language "ep1_zpl" = Eltron Programming Language and Zebra Programming Language "hybrid_xml_zpl" = XML and ZPL Programming Languages <i>Default Value: "ep1_zpl"</i>

* The setvar command is supported only on Zebra G-Series™ printers.



Example • This example, the getvar result is the current programming language that the printer is using.

```
! U1 getvar "device.languages"
```

Table 22 • Programming Languages

zpl (Zebra Programming Language)
ep1 (Eltron Programming Language)
ep1_zpl (Eltron and Zebra Programming Languages)
hybrid_xml_zpl (both XML and ZPL)

device.orientation

Description 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.

Supported Devices

- KR403

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the parameters for this format:

Commands	Details
getvar	This command instructs the printer to respond with the currently set presenter loop length. <i>Format:</i> ! U1 getvar "device.orientation"
setvar	This command instructs the printer to change the presenter loop length. <i>Format:</i> ! U1 setvar "device.orientation" "value" <i>Values:</i> 0 = printer is installed horizontally 1 = printer is installed vertically <i>Default:</i> 0 = printer is installed horizontally

device.pnp_option

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the Plug and Play option setting. <i>Format: ! U1 getvar "device.pnp_option"</i>
setvar	This command instructs the printer to select the desired Plug and Play response option. <i>Format: ! U1 setvar "device.pnp_option" "value"</i> <i>Values:</i> "epl" = Eltron Programming Language "zpl" = Zebra Programming Language <i>Default: "zpl"</i>



Example • This setvar example shows the value set to "zpl".

```
! U1 setvar "device.pnp_option" "epl"
```

When the setvar value is set to "epl", the getvar result is "epl".

device.reset

Description This command instructs the printer to perform a soft reset. For printer support, see [SGD Command Support on page 997](#).

Type setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the command for this format:

Commands	Details
setvar	This command instructs the printer to perform a soft reset. <i>Format:</i> ! U1 setvar "device.reset" ""



Example • In this example, the setvar performs a soft reset.

```
! U1 setvar "device.reset" ""
```

device.restore_defaults

Description This command restores to the default of all settings within the specified SGD branch. For printer support, see [SGD Command Support on page 997](#).

Type do;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
do	This command restores the default setting for all items within the specified branch. <i>Format:</i> ! U1 do "device.restore_default" "value" <i>Values:</i> "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
setvar	This command restores to the default of all settings within the specified branch. <i>Format:</i> ! U1 setvar "device.restore_default" "value" <i>Values:</i> "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.

```
do    ! U1 do "device.restore_default" "wlan"

setvar ! U1 setvar "device.restore_default" "wlan"
```

device.unique_id

Description This command retrieves the printer identifier. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the established printer identifier. <i>Format:</i> ! 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.uptime

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the amount of time the print has been powered on. It responds in the following format (days, hours, minutes, and seconds). <i>Format:</i> ! 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

Description This command saves and retrieves user specified values. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command saves and retrieves user specified parameters. <i>Format: ! U1 getvar "device.user_p1"</i>
setvar	This command instructs the printer to set user parameters. <i>Format: ! U1 setvar "device.user_p1" "value"</i> <i>Values: alphanumeric text string (1 - 20)</i> <i>Default Value: ""</i>



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

Description This command saves and retrieves user specified values. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command saves and retrieves user specified parameters. <i>Format:</i> ! U1 getvar "device.user_p2"
setvar	This command instructs the printer to set user parameters. <i>Format:</i> ! U1 setvar "device.user_p2" "value" <i>Values:</i> alphanumeric text string (1 - 20) <i>Default:</i> ""



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.xml.enable

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type setvar;getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command enables and disables language parsing support for XML. <i>Format:</i> ! U1 getvar "device.xml.enable"
setvar	This command instructs the printer to disable or enable the language parsing support for XML. <i>Format:</i> ! U1 getvar "device.xml.enable" "value" <i>Values:</i> "on" = enables language parsing support for XML "off" = disables language parsing support for XML <i>Default Value:</i> on



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".

display.text

Description This command retrieves the text data that is being used on the printer’s LCD. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the text data that appears on the printer’s LCD. <i>Format: ! U1 getvar "display.text"</i>



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"
```

file.delete

Description This command instructs the printer to delete specified files. For printer support, see [SGD Command Support on page 997](#).

Type do



Note • For details on SGD command structure, see [Command Structure on page 718](#).



Important • Be sure to always specify the memory location.

This table identifies the command for this format:

Commands	Details
do	This command instructs the printer to delete specified files. <i>Format:</i> ! U1 do "file.delete" "value" <i>Values:</i> file name



Example • This do example shows the specified file to delete.
! U1 do "file.delete" "e:abcd.zpl"

file.dir

Description This command displays a directory listing on the same port the command was received. For printer support, see [SGD Command Support on page 997](#).

Type do;getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).



Important • Be sure to always specify the memory location.

This table identifies the command for this format:

Commands	Details
do	This command sets the directory name from which to retrieve files. <i>Format:</i> ! U1 do "file.dir" "value" <i>Values:</i> directory letter
getvar	This command retrieves a directory listing of the specified directory. <i>Format:</i> ! U1 getvar "file.dir"
setvar	This command sets the directory name from which to retrieve files. <i>Format:</i> ! U1 setvar "file.dir" "value" <i>Values:</i> directory letter



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.type

Description This command displays the contents of the specified file. For printer support, see [SGD Command Support on page 997](#).

Type do;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).



Important • Be sure to always specify the memory location.

This table identifies the command for this format:

Commands	Details
do	This command displays the content of a file on the same port the command was received. <i>Format:</i> ! U1 do "file.type" "value" <i>Values:</i> the drive letter, file name, file extension, such as R:TEST.ZPL
setvar	This command instructs the printer to display the content of a file on the same port the command was received. <i>Format:</i> ! U1 setvar "file.type" "value" <i>Values:</i> the drive letter, file name, file extension, such as R:TEST.ZPL



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

Description This command instructs the printer to send a specified file to the parser. For printer support, see [SGD Command Support on page 997](#).

Type do; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).



Important • Be sure to always specify the memory location.

This table identifies the command for this format:

Commands	Details
do	This command instructs the printer to send a specified file to the parser. <i>Format:</i> ! U1 do "file.run" "value" <i>Values:</i> drive:filename.extension
setvar	This command instructs the printer to send a specified file to the parser. <i>Format:</i> ! U1 setvar "file.run" "values" <i>Values:</i> drive:filename.extension



Example • This setvar example will send the file "text.zpl" stored in RAM to the parser.

```
! U1 setvar "file.run" "R:text.zpl"
```

head.latch

Description This command identifies if the printhead is open or closed. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the status of the printhead, open or closed. <i>Format:</i> ! U1 getvar "head.latch" <i>Values:</i> "ok" = closed "open" = open



Example • In this example, the `getvar` retrieves the status of the print head.

```
! U1 getvar "head.latch"
```

```
"ok"
```

interface.network.active.gateway

Description This command retrieves the gateway address of the active print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the gateway address of the active print server. <i>Format:</i> ! 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

Description This command retrieves the IP address of the active print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the IP address of the active print server. <i>Format:</i> ! 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.mac_addr

Description This command retrieves the MAC address of the active print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the MAC address of the active print server. <i>Format:</i> ! 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

Description This command identifies the RAW MAC address of the active print server. The raw mac address is the mac address without the colons (" : "). For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the RAW MAC address of the active print server. <i>Format:</i> ! 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

Description This command retrieves the netmask of the active print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the netmask of the active print server. <i>Format:</i> ! 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

Description This command retrieves IP protocol of the active print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the IP protocol of the active print server. <i>Format:</i> ! U1 getvar "interface.network.active.protocol"



Example • In this example, the getvar retrieves the IP protocol of the active print server.
! U1 getvar "interface.network.active.protocol"

Possible values include:

"bootp"
"dhcp"
"rarp"
"glean"
"permanent"

media.cut_now

Description 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 669](#).

Supported Devices

- KR403

Type do; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the parameters for this format:

Commands	Details
do	This command instructs the printer to cycle the media cutter. <i>Format:</i> ! U1 do "media.cut_now" ""
setvar	This command instructs the printer to cycle the media cutter. <i>Format:</i> ! U1 setvar "media.cut_now" ""  Note • See media.present.cut_amount on page 669.

media.darkness_mode

Description

This printer setting enables or disables the Supplies Warning system. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves setting for the Supplies Warning system <i>Format:</i> ! U1 getvar "media.media_low.warning"
setvar	This command enables or disables the Supplies Warning system. <i>Format:</i> ! U1 setvar "media.media_low.warning" "value" <i>Values:</i> "Disabled" = not active "Enabled" = active <i>Default:</i> "Disabled" Values outside of the given rane will be ignored and the value will be left unchanged.



Example • This setvar example shows the darkness mode set to "cartridge".
! U1 setvar "media.darkness_mode" "cartridge"

media.media_low.external

Description This printer setting gets the status of the external media_low warning. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- KR403

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the parameters for this format:

Commands	Details
getvar	This command instructs the printer to respond with the currently set media print mode. <i>Format:</i> <pre>! U1 getvar "media.media_low.external"</pre> <i>Values:</i> 0 = Paper present at sensor position 1 = 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

Description This command retrieves the value of, or enables or disables the Supplies Warning system. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- Xi4

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the setting for the Supplies Warning system. <i>Format:</i> ! U1 getvar "media.media_low.warning"
setvar	This command enables or disables the Supplies Warning System. <i>Format:</i> ! U1 setvar "media.media_low.warning" "value" <i>Values:</i> "disabled" = not active "enabled" = active <i>Default:</i> "disabled"



Example • This setvar example disables the Supplies Warning system.

```
! U1 setvar "media.media_low.warning" "disabled"
```



Example • This setvar example enables the Supplies Warning system.

```
! U1 setvar "media.media_low.warning" "enabled"
```

media.present.cut_amount

Description 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. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- KR403

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the parameters for this format:

Commands	Details
getvar	This command instructs the printer to respond with the currently set media cut amount. <i>Format:</i> ! U1 getvar "media.present.cut_amount"
setvar	This command instructs the printer to change the media cut amount. <i>Format:</i> ! U1 setvar "media.present.cut_amount" "value" <i>Values:</i> 0 = normal cut 10-40 = partial cut, value = mm of cut from each side

media.present.eject

Description 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. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- KR403

Type do; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the parameters for this format:

Commands	Details
do	This command instructs the printer to eject the document through the presenter module. <i>Format:</i> ! U1 do "media.present.eject" "value"
setvar	This command instructs the printer to eject the document through the presenter module. <i>Format:</i> ! U1 setvar "media.present.eject" "value" <i>Values:</i> 0 - 255 = amount of media to eject in mm  Note • See media.present.length_addition on page 671 .

media.present.length_addition

Description 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. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- KR403

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the parameters for this format:

Commands	Details
getvar	This command instructs the printer to respond with the currently set media present length addition. Format: : ! U1 getvar "media.present.length_addition"
setvar	This command instructs the printer to change the media present length addition. Format: ! U1 setvar "media.present.length_addition" "value" Values: 0-255 = additional mm of media to eject

media.present.loop_length

Description 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. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- KR403

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the parameters for this format:

Commands	Details
getvar	This command instructs the printer to respond with the currently set presenter loop length. <i>Format:</i> ! U1 getvar "media.present.loop_length"
setvar	This command instructs the printer to change the presenter loop length. <i>Format:</i> ! U1 setvar "media.present.loop_length" "value" <i>Values:</i> 0 = paper is fed straight through the presenter 3-1023 = loop length in mm. <i>Default:</i> 50 = gives a loop of approximately 50mm

media.present.loop_length_max

Description This printer setting determines the maximum allowed length of the presenter loop. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- KR403

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the parameters for this format:

Commands	Details
getvar	This command instructs the printer to respond with the currently set presenter loop length. <i>Format:</i> ! U1 getvar "media.present.loop_length_max"
setvar	This command instructs the printer to change the presenter loop length. <i>Format:</i> ! U1 setvar "media.present.loop_length_max" "value" <i>Values:</i> 0 = paper is fed straight through the presenter 3-1023 = loop length in mm. <i>Default:</i> 600 = gives a loop of approximately 50mm

media.present.cut_margin

Description This printer setting determines the margin between the cutter and the printhead. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- KR403

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the parameters for this format:

Commands	Details
getvar	This command instructs the printer to respond with the currently set media cut margin amount. <i>Format:</i> <pre>! U1 getvar "media.present.cut_margin"</pre>
setvar	This command instructs the printer to change the media cut amount. <i>Format:</i> <pre>! U1 setvar "media.present.cut_margin" "value"</pre> <i>Values:</i> 2 - 9 = mm of distance <i>Default: :</i> 9 = mm of distance

media.present.present_timeout

Description This printer setting determines how long the printer will wait after a present event to clear the label. See ^KV ZPL command. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- KR403

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the parameters for this format:

Commands	Details
getvar	This command instructs the printer to respond with the currently set presenter function mode. <i>Format:</i> <pre>! U1 getvar "media.present.present_timeout" a</pre>
setvar	This command instructs the printer to change the presenter function mode. <i>Format:</i> <pre>! U1 setvar "media.present.present_timeout" "value"</pre> <i>Values:</i> 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.

media.present.present_type

Description This printer setting determines the way that the printer performs a present command. See ^KV ZPL command. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- KR403

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the parameters for this format:

Commands	Details
getvar	This command instructs the printer to respond with the currently set presenter function mode. Format: ! U1 getvar "media.present.present_type"
setvar	This command instructs the printer to change the presenter function mode. Format: ! U1 setvar "media.present.present_type" "value" 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

media.printmode

Description This printer setting determines the action the printer takes after a label or group of labels has printed. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the the currently set media print mode. <i>Format:</i> ! U1 getvar "media.printmode"
setvar	This command instructs the printer to change the media print mode. <i>Format:</i> ! U1 setvar "media.printmode" "value" <i>Values:</i> tear off = "T" ^a peel off = "P" ^a rewind = "R" applicator = "A" ^a cutter = "C" delayed cutter = "D" ^a reserved = "L" ^{a, b} reserved = "U" ^a kiosk = "K" ^c

- a.** This value is not supported on the KR403 printer.
- b.** This value is supported only on the ZM400 and ZM600 printers.
- c.** This value is supported only on the KR403 printer.



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".

For more details on how each setvar value relates to the getvar responses, see [Table 23, Setvar / Getvar Relation on page 678](#).

Table 23 • Setvar / Getvar Relation

If the setvar is set to ...	Then the getvar response and control panel display is ...
"T"	TEAR OFF
"P"	PEEL OFF
"R"	REWIND
"A"	APPLICATOR
"C"	CUTTER
"D"	DELAYED CUT
"L"	RESERVED
"U"	RESERVED
"K"	KIOSK

media . speed

Description This command specifies media print speed in inches per second (ips). For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the currently set media print speed. <i>Format: ! U1 getvar "media.speed"</i>
setvar	This command instructs the printer to set the media print speed. <i>Format: ! U1 setvar "media.speed" "value"</i> <i>Values:</i> 2-12 ips "up" = increments the printer speed by one unit "down" = decrements the speed by one unit <i>Default: "2"</i>

- ➔

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".
- ➔

Example • This setvar example shows the value set to "up".

```
! U1 setvar "media.speed" "up"
```

If the current print speed is 2: When the setvar value is set to "up", the getvar result is "3".
- ➔

Example • This setvar example shows the value set to "down".

```
! U1 setvar "media.speed" "down"
```

If the current print speed is 2: When the setvar value is set to "down", the getvar result is "1".

odometer.headclean

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to retrieve the values for the head clean counter. <i>Format:</i> ! U1 getvar "odometer.headclean"
setvar	This command instructs the printer to reset the head clean counter. <i>Format:</i> ! U1 setvar "odometer.headclean" "value" <i>Values:</i> "0" = reset the head clean counter <i>Default:</i> must be an accepted value or it is ignored



Example • This example shows how to get the odometer head clean, how to reset it, and how to confirm the settings changed.

1. To see the current settings, type:
! U1 getvar "odometer.headclean"
Something similar to this is shown:
"1489 INCHES, 3784 CENTIMETERS"
2. To reset the these values to 0, type:
! U1 setvar "odometer.headclean" "0"
3. 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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to retrieve the values for the head new counter. <i>Format:</i> ! U1 getvar "odometer.headnew"
setvar	This command instructs the printer to reset the head new counter. <i>Format:</i> ! U1 setvar "odometer.headnew" "value" <i>Values:</i> "0" = resets the head new counter <i>Default:</i> must be an accepted value or it is ignored



Example • This example shows how to get the odometer head new, how to reset it, and how to confirm the settings changed:

1. To see the current settings, type:
! U1 getvar "odometer.headnew"
Something similar to this is shown:
"1489 INCHES, 3784 CENTIMETERS"
2. To reset the these values to 0, type:
! U1 setvar "odometer.headnew" "0"
3. 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

Description This command returns the length of the last label printed or fed (in dots). For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command returns the length of the last label printed or fed (in dots). <i>Format:</i> ! 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.

1. To change the odometer label dot length, type:

```
^XA
```

```
^LL500
```

```
^XZ
```

2. 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_count1

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on the command structure of SGD commands, see [Command Structure on page 628](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to return the current value of the first (count1) user resettable counter in both inches and centimeters. <i>Format:</i> ! U1 getvar "odometer.media_marker_count1"
setvar	This command instructs the printer to reset the first user resettable counter. <i>Format:</i> ! U1 setvar "odometer.media_marker_count1" "value" <i>Values:</i> "0" = reset the counter <i>Default:</i> must be an accepted value or it is ignored



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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to return the current value of the second (count2) user resettable counter in both inches and centimeters. <i>Format:</i> ! U1 getvar "odometer.media_marker_count2"
setvar	This command instructs the printer to reset the second user resettable counter. <i>Format:</i> ! U1 setvar "odometer.media_marker_count2" "value" <i>Values:</i> "0" = reset the counter <i>Default:</i> must be an accepted value or it is ignored



Example • This example shows how to get the second user resettable counter, how to reset it, and how to confirm the settings have changed:

1. To see the current settings, type:
! U1 getvar "odometer.media_marker_count2"
Something similar to this is shown:
"8516 INCHES, 21632 CENTIMETERS"
2. To reset the these values to 0, type:
! U1 setvar "odometer.media_marker_count2" "0"
3. 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.retracts_count

Description This printer value records the number of times a label has been retracted since the last time the counter has been reset.

Supported Devices

- KR403

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the parameters for this format:

Commands	Details
getvar	This command instructs the printer to respond with the current number of retractions that have happened since the last time the counter was reset. <i>Format:</i> ! U1 getvar "odometer.retracts_count"
setvar	This command instructs the printer to reset the current count of retractions. <i>Format:</i> ! U1 setvar "odometer.retracts_count" "value" <i>Values:</i> 0 = reset the counter <i>Default:</i> none

odometer.total_print_length

Description This command tracks the total length of media that printed over the life of the printer. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command returns the value of the total length of media that printed over the life of the printer. <i>Format: ! U1 getvar "odometer.total_print_length"</i>



Example • This example shows how to get the total length of media that printed over the life of the printer.

1. To get the total length of media that has printed to date, type:
! U1 getvar "odometer.total_print_length"

Something similar to this is shown:

"8560 INCHES, 21744 CENTIMETERS"

print.tone

Description This command specifies the printer darkness. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the printer’s current darkness setting. <i>Format: ! U1 getvar "print.tone"</i>
setvar	This command instructs the printer to set the darkness and relative darkness. <i>Format: ! U1 setvar "print.tone" "value"</i> <i>Values:</i> "0.0" to "30.0" = darkness "-0.1" to "-30.0" and "+0.1" to "+30.0" = incremental adjustments <i>Default: "4.0"</i>



Example • This setvar example shows the value set to "4.0".

```
! U1 setvar "print.tone" "4.0"
```

When the setvar value is set to "4.0", the getvar result is "4.0".

rfid.error.response

Description During an error condition, an error message shows on the second line of the display. This command can be used to retrieve that error message. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command instructs the printer to respond with any active RFID error messages. <i>Format: ! U1 getvar "rfid.error.response"</i>



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.position.program

Description This command sets the read/write position of the transponder in vertical (Y axis) dot rows from the top of the label. Set to 0 (no movement) if the transponder is already in the effective area without moving the media. For printer support, see [SGD Command Support on page 997](#).



Important • If a label format specifies a value for this parameter, this value will be used for the programming position for all labels until a new position is specified or until the printer is turned off (O) and then back on (I).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command instructs the printer to respond with the current programming position. <i>Format:</i> ! U1 getvar "rfid.position.program"
setvar	This command instructs the printer to set the programming position. <i>Format:</i> ! U1 setvar "rfid.position.program" "value" <i>Values:</i> 0 to label length <i>Default value:</i> For the R110PAX4 and R2844-Z: 0 For all other supported printers: label length minus 1 mm (1/16 in.) <i>Default:</i> 1



Example • This setvar example shows the programming position being set at 15 dot rows from the top of the label.

```
! U1 setvar "rfid.position.program" "15"
```

When the setvar value is set to "15", the getvar result is "15".

rfid.reader_1.antenna_port

Description This command selects the RFID antenna port. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command instructs the printer to respond with the current antenna port. <i>Format:</i> ! U1 getvar "rfid.reader_1.antenna_port"
setvar	This command instructs the printer to set the antenna port. <i>Format:</i> ! U1 setvar "rfid.reader_1.antenna_port" "value" <i>Values:</i> 1 = antenna port 1 2 = antenna port 2 <i>Default:</i> 1



Example • This setvar example shows the selection of antenna port 2.

```
! U1 setvar "rfid.reader_1.antenna_port" "2"
```

When the setvar value is set to "2", the getvar result is "2".

rfid.reader_1.power.read

Description This command sets the RFID reader power level for reading RFID tags. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command instructs the printer to respond with the antenna’s current read power level. <i>Format: ! U1 getvar "rfid.reader_1.power.read"</i>
setvar	This command instructs the printer to set the antenna’s read power level. <i>Format:</i> ! U1 setvar "rfid.reader_1.power.read" "value" R53.16.3Z: <i>Values:</i> 0 to 30 <i>Default Value:</i> 16 R53.16.4Z and later: <i>Values:</i> 0 to 30, up, down (up and down change the current value by 1) <i>Default Value:</i> 16 R60.16.x, R62.16.x, R63.16.x, R65.16.x, SP994Q, SP999G, SP1027G, SP1056F, SP1082G, and later: <i>Values:</i> 0 to 30, high, medium, low <i>Default Value:</i> low Older firmware: <i>Values:</i> high medium low <i>Default:</i> low



Example • This setvar example sets the antenna to high power for reading RFID tags.

```
! U1 setvar "rfid.reader_1.power.read" "high"
```

When the setvar value is set to "high", the getvar result is "high".

rfid.reader_1.power.single_power

Description This command sets the RFID reader power level for reading and writing to RFID tags for readers with a single power level. For printer support, see [SGD Command Support on page 997](#).



Note • This command applies only to the R110Xi HF printer, firmware version R65.X.

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command instructs the printer to respond with the current power level. <i>Format:</i> ! U1 getvar "rfid.reader_1.power.single_power"
setvar	This command instructs the printer to set the power level for reading and writing. <i>Format:</i> ! U1 setvar "rfid.reader_1.power.single_power" "value" <i>Values:</i> high medium low <i>Default:</i> low



Example • This setvar example sets the antenna to high power for writing to RFID tags.

```
! 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.power.write

Description This command sets the RFID reader power level for writing to RFID tags. For printer support, see [SGD Command Support on page 997](#).



Note • This parameter is ignored on the R110Xi HF printer because read and write powers cannot be specified separately. See [rfid.reader_1.power.single_power on page 692](#) to set the power level for the R110Xi HF printer.

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command instructs the printer to respond with the antenna’s current write power level. <i>Format: ! U1 getvar "rfid.reader_1.power.write"</i>
setvar	This command instructs the printer to set the write power level on the RFID reader. <i>Format:</i> ! U1 setvar "rfid.reader_1.power.write" "value" R53.16.3Z: <i>Values:</i> 0 to 30 <i>Default Value:</i> 16 R53.16.4Z and later: <i>Values:</i> 0 to 30, up, down (up and down change the current value by 1) <i>Default Value:</i> 16 R60.16.x, R62.16.x, R63.16.x, R65.16.x, SP994Q, SP999G, SP1027G, SP1056F, SP1082G, and later: <i>Values:</i> 0 to 30, high, medium, low <i>Default Value:</i> low Older firmware: <i>Values:</i> high medium low <i>Default:</i> low



Example • This setvar example sets the antenna to high power for writing to RFID tags.

```
! U1 setvar "rfid.reader_1.power.write" "high"
```

When the setvar value is set to "high", the getvar result is "high".

zbi.control.add_breakpoint

Description This command instructs the printer to set a ZBI program break point. For printer support, see [SGD Command Support on page 997](#).

Type setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
setvar	This command instructs the printer to set a ZBI program break point. <i>Format:</i> <pre>! U1 setvar "zbi.control.add_breakpoint" "value"</pre> <i>Values:</i> 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

Description This command breaks the execution of the ZBI 2.0 program that is currently running. For printer support, see [SGD Command Support on page 997](#).

Type setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
setvar	This command breaks the execution of the ZBI program that is currently running. Format: ! U1 setvar "zbi.control.break" "" Values: ""



Example • This setvar example shows the value set to "".

```
! U1 setvar "zbi.control.break" ""
```

zbi.control.clear_breakpoints

Description This command deletes all breakpoints in the current ZBI 2.0 program. For printer support, see [SGD Command Support on page 997](#).

Type setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
setvar	This command instructs the printer to delete all breakpoints. <i>Format:</i> <pre>! U1 setvar "zbi.control.clear_breakpoints" ""</pre> <i>Values:</i> ""



Example • This setvar example shows the value set to "".

```
! U1 setvar "zbi.control.clear_breakpoints" ""
```


zbi.control.delete_breakpoint

Description This command deletes a breakpoint in the current ZBI 2.0 program. For printer support, see [SGD Command Support on page 997](#).

Type setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
setvar	This command instructs the printer to delete the breakpoint at the line indicated by the value parameter. <i>Format:</i> ! U1 setvar "zbi.control.delete_breakpoint" "value" <i>Values:</i> 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

Description This command gives you control and information about which line of a stopped ZBI 2.0 program is being executed. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command returns the line number that is currently being executed in the ZBI 2.0 program. <i>Format:</i> ! U1 getvar "zbi.control.line_number"
setvar	This command sets which line of the current ZBI 2.0 program should be executed. <i>Format:</i> ! U1 setvar "zbi.control.line_number" "value" <i>Values:</i> Any line number of the currently stopped ZBI program. <i>Default:</i> "0"



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

Description This command restarts a ZBI 2.0 program that is currently stopped. For printer support, see [SGD Command Support on page 997](#).

Type setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
setvar	This command restarts a ZBI 2.0 program that is currently stopped. <i>Format:</i> ! U1 setvar "zbi.control.restart" "value" <i>Values:</i> " "



Example • This setvar example shows the value set to " ".

```
! U1 setvar "zbi.control.restart" " "
```

zbi.control.run

Description This command runs the current ZBI 2.0 program that is loaded in the interpreter. For printer support, see [SGD Command Support on page 997](#).

Type setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
setvar	This command runs the ZBI 2.0 program that is loaded in the interpreter. <i>Format:</i> ! U1 setvar "zbi.control.run" "" <i>Values:</i> ""



Example • This setvar example shows the value set to "".

```
! U1 setvar "zbi.control.run" ""
```

zbi.control.step

Description This command restarts the execution of the currently stopped ZBI 2.0program for one line. For printer support, see [SGD Command Support on page 997](#).

Type setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
setvar	This command instructs the printer to restart the execution of the currently stopped ZBI 2.0 program for one line. <i>Format:</i> ! U1 setvar "zbi.control.step" "" <i>Values:</i> "" <i>Default:</i> ""



Example • This setvar example shows the value set to "".

```
! U1 setvar "zbi.control.step" ""
```

When the setvar value is set to "".

zbi.control.terminate

Description This command instructs the ZBI 2.0 program to terminate and shuts down the interpreter. For printer support, see [SGD Command Support on page 997](#).

Type setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
setvar	This command instructs the ZBI 2.0 program to terminate and shuts down the interpreter. <i>Format:</i> ! U1 setvar "zbi.control.terminate" "value" <i>Values:</i> ""



Example • This setvar example shows the value set to "".

```
! U1 setvar "zbi.control.terminate" ""
```

zbi.control.variable_name

Description This command sets the name of the variable that is to be read or modified through `variable_value`. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the variable value that is to show on the front panel. <i>Format:</i> ! U1 getvar "zbi.control.variable_name"
setvar	This command sets the variable that is to show on the front panel. <i>Format:</i> ! U1 setvar "zbi.control.variable_name" "value" <i>Values:</i> Any ZBI variable in the program that is currently being debugged. <i>Default:</i> ""



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

Description This command identifies the variable name. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the variable name that is loaded into the variable_name. <i>Format:</i> ! U1 getvar "zbi.control.variable_value"
setvar	This command sets a value to the variable referenced by a variable_name. <i>Format:</i> ! U1 setvar "zbi.control.variable_value" "value" <i>Values:</i> A string or integer that is dependent on the variable type in variable_name. <i>Default:</i> The current value of the variable referenced via variable_name



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.key

Description This command identifies if the ZBI 2.0 option is enabled or disabled on the printer. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the status of the ZBI 2.0 option on the printer. <i>Format: ! U1 getvar "zbi.key"</i>



Example • In this example, the `getvar` shows the status of ZBI on the printer.

```
! U1 getvar "zbi.key"
"ENABLED"
```

zbi.last_error

Description This command identifies the last error that the ZBI 2.0 interpreter encountered. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command shows the last error that the ZBI 2.0 interpreter encountered. <i>Format: ! U1 getvar "zbi.last_error"</i>



Example • This example demonstrates how to make the ZBI 2.0 interpreter return the last error it encountered.

```
! U1 getvar "zbi.last_error"
```

zbi.reseller_key

Description This command allows programs that are encrypted with this key in ZBI-Developer to run. For printer support, see [SGD Command Support on page 997](#).

Type setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
setvar	This command allows programs that are encrypted with this key in ZBI developer to run. <i>Format:</i> ! U1 setvar "zbi.reseller_key" "value" <i>Values:</i> 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

Description This command identifies the current ZBI version. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the current ZBI version. <i>Format:</i> ! U1 getvar "zbi.revision"



Example • In this example, the `getvar` shows the current ZBI version.

```
! U1 getvar "zbi.revision"
```

zbi.running_program_name

Description This command identifies the name of the ZBI 2.0 program that is currently running. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the name of the currently running ZBI 2.0 program. <i>Format:</i> ! 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

Description This command instructs the ZBI 2.0 environment to execute the program listed in the file_name. For printer support, see [SGD Command Support on page 997](#).

Type setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
setvar	This command instructs the ZBI 2.0 environment to execute the program listed in the file_name. <i>Format: ! U1 setvar "zbi.start_info.execute"</i>



Example • This setvar example executes the "choices.bas" program.
! U1 setvar "zbi.start_info.execute" "choices.bas"

zbi.start_info.file_name

Description This command prepares a program to run when the `zbi.start_info.execute` command is used. This command does not run the program. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command is used to return the file path and file name of a ZBI 2.0 program to run using the <code>zbi.start_info.execute</code> command. <i>Format:</i> ! U1 getvar "zbi.start_info.file_name" <i>Default:</i> The last program run. If nothing has been run, "*:\.BAZ".
setvar	This command is used to prepare a ZBI 2.0 program to be executed using the <code>zbi.start_info.execute</code> command. <i>Format:</i> ! U1 setvar "zbi.start_info.file_name" "value" <i>Values:</i> a file name or path of a basic program



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

Description This command identifies the amount of memory currently in use in a ZBI 2.0 program. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command shows the amount of memory currently in use in bytes. <i>Format:</i> ! 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

Description This command shows the current state of the ZBI 2.0 program. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the current state of ZBI. <i>Format:</i> ! U1 getvar "zbi.state" <i>Values:</i> "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"
```



Notes • _____



SGD Wired Commands

This chapter provides a high-level overview of the wired Set / Get / Do (SGD) commands. For printer support of these SGD commands, see [SGD Command Support on page 997](#).



SGD commands are available in printers with firmware version V60.15.x, V50.15.x, V61.15.x, V56.15.x, V53.15.xZ, 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.



Note • SGD commands must be terminated by a carriage return or a space and line feed.

SGD commands are commands that allow you to configure all printers with firmware version V60.15.x, V50.15.x, V61.15.x, V56.15.x, V53.15.xZ, or later. The printer performs the specified function immediately after receiving the command. The commands are:

- **setvar**
- **getvar**
- **do**

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)



Important • The setvar command and attributes must be specified in lower case.

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



Important • The printer settings and attributes must be specified in lower case.

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.

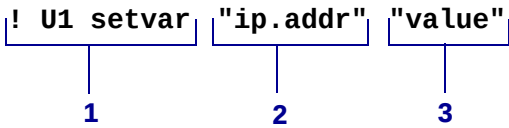


Important • The values must be specified in lower case.

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.

➔ **Example** • This is an example of a command structure illustration:



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).

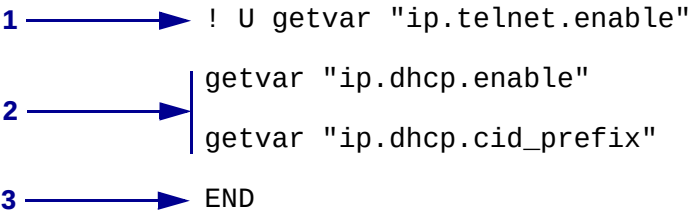


Note • Some RF cards do not support all of the SGD commands.

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.

➔ **Example** • This syntax shows how you can send multiple **getvar** commands:



1	The command portion of the string does not use the "1" after the "! U".
2	Commands issued after the first command do not require the "! U".
3	The string of commands is terminated by the word "END" with a space after the word, and by a carriage return/ line feed.

external_wired.check

Description This command controls whether to check for external print server during the network interface search. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the status of the network interface search. <i>Format:</i> ! U1 getvar "external_wired.check"
setvar	This command instructs the printer to set the network interface search. <i>Format:</i> ! U1 setvar "external_wired.check" "value" <i>Values:</i> "on" = turn on external wired network interface search "off" = turn off external wired network interface search <i>Default:</i> "on" = If wireless option board is not installed "off" = If wireless option board is installed



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

Description This command allows you to get or set the external wired print servers’s IP address. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Important • For a set IP address to take affect, the IP protocol must be set to permanent and the print server must be reset.



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar *	This command instructs the printer to respond with its current external wired print server IP address. <i>Format:</i> ! U1 getvar "external_wired.ip.addr"
setvar	This command instructs the printer to change its current external wired print server IP address upon powering the printer on. <i>Format:</i> ! U1 setvar "external_wired.ip.addr" "value" <i>Values:</i> any valid IP address <i>Default:</i> "0.0.0.0"

* 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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar *	This command instructs the printer to respond with the ARP interval or the ARP cache time out value for the external wired print server. <i>Format:</i> ! U1 getvar "external_wired.ip.arp_interval"
setvar	This command instructs the printer to change the ARP interval or the ARP cache time out for the external wired print server. <i>Format:</i> ! U1 setvar "external_wired.ip.arp_interval" "value" <i>Values:</i> 0 - 30 <i>Default:</i> "0"

* 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

Description This command allows you to default the external wired print server’s IP address. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Important • For a set IP address to take affect, the IP protocol must be set to permanent and the print server must be reset.



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar *	This command instructs the printer to show the status of the setting of external wired print server’s default IP address feature. <i>Format:</i> <pre>! U1 getvar "external_wired.ip.default_addr_enable"</pre>
setvar	This command tells the printer to use it’s default address, 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. <i>Format:</i> <pre>! U1 setvar "external_wired.ip.default_addr_enable" "value"</pre> <i>Values:</i> <pre>"on" = enabled "off" = disabled</pre> <i>Default: "on"</i>

* 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

Description 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". For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar *	This command instructs the printer to respond with the client identifier prefix and suffix of the external wired print server. <i>Format:</i> <pre>! U1 getvar "external_wired.ip.dhcp.cid_all"</pre>
setvar	This command instructs the printer to change the client identifier prefix and suffix of the external wired print server. The prefix gets cleared and the suffix contains the entire client identifier. <i>Format:</i> <pre>! U1 setvar "external_wired.ip.dhcp.cid_all" "value"</pre> <i>Values:</i> A maximum length of 60 characters if the CID type is ASCII, or 120 characters if the CID type is hexadecimal. <i>Default Value:</i> ""

* 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

Description This command determines if DHCP (option 61) on the external wired print server is turned on or off. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar *	This command instructs the printer to respond with the status of the client identifier of the external wired print server. <i>Format:</i> <pre>! U1 getvar "external_wired.ip.dhcp.cid_enable"</pre>
setvar	This command instructs the printer to set the status of the client identifier of the external wired print server. <i>Format:</i> <pre>! U1 setvar "external_wired.ip.dhcp.cid_enable" "value"</pre> <i>Values:</i> <pre>"off" = client identifier is turned off "on" = client identifier is turned on</pre> <i>Default:</i> "off"

* 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

Description 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". For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar *	This command instructs the printer to respond with the client identifier prefix of the external wired print server. <i>Format:</i> <pre>! U1 getvar "external_wired.ip.dhcp.cid_prefix"</pre>
setvar	This command instructs the printer to change the CID prefix of the external wired print server. <i>Format:</i> <pre>! U1 setvar "external_wired.ip.dhcp.cid_prefix" "value"</pre> <i>Values:</i> Any text string up to 10 characters if the CID type is ASCII, or 20 characters if the CID type is hexadecimal. <i>Default Value:</i> ""

* 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

Description 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 727](#) is set to "0" or "2", not "1". For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar *	This command instructs the printer to respond with the client identifier suffix on the external wired print server. Format: ! U1 getvar "external_wired.ip.dhcp.cid_suffix"
setvar	This command instructs the printer to change the client identifier suffix value. Format: ! 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: ""

* 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

Description 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". For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar *	This command instructs the printer to respond with the client identifier type for the external wired print server. <i>Format:</i> <pre>! U1 getvar "external_wired.ip.dhcp.cid_type"</pre>
setvar	This command instructs the printer to enable "synthetic" Client Identifier for the external wired print server. <i>Format:</i> <pre>! U1 setvar "external_wired.ip.dhcp.cid_type" "value"</pre> <i>Values:</i> <pre>"0" = ASCII string "1" = wired print server's MAC address "2" = HEX value</pre> <i>Default Value:</i> "1"

* 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

Description This command instructs the printer to change the external wired print server’s gateway address. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Important • This setting refers to the gateway address. A set value is ignored if the IP protocol is not set to permanent.



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar *	This command instructs the printer to respond with the external wired printer server’s gateway address. <i>Format:</i> ! U1 getvar "external_wired.ip.gateway"
setvar	This command instructs the printer to change the external wired printer server’s gateway address. <i>Format:</i> ! U1 setvar "external_wired.ip.gateway" "value" <i>Values:</i> Any valid gateway address <i>Default:</i> "0.0.0.0"

* 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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar *	This command instructs the printer to respond with the external wired print server’s subnet mask. <i>Format:</i> ! U1 getvar "external_wired.ip.netmask"
setvar	This command instructs the printer to change the external wired print servers’s subnet mask. <i>Format:</i> ! U1 setvar "external_wired.ip.netmask" "value" <i>Values:</i> Any valid subnet mask. <i>Default:</i> "255.255.255.0"

* 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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar *	This command instructs the printer to respond with the external wired printer server's TCP/UDP port number. <i>Format:</i> ! U1 getvar "external_wired.ip.port"
setvar	This command instructs the printer to set the external wired print server's TCP/UDP port number. <i>Format:</i> ! U1 setvar "external_wired.ip.port" "value" <i>Values:</i> 1 - 65535 (excluding any ports currently used by other services, such as 21, 23, 80, and 515). <i>Default:</i> "9100"

* 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

Description This command configures the IP addressing method used by the external wired print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the command for this format:

Commands	Details
getvar *	This command returns the IP addressing method used by the external print server. <i>Format:</i> ! U1 getvar "external_wired.ip.protocol"
setvar	This command instructs the printer to configure the IP addressing method used by the external wired print server. <i>Format:</i> <pre>! U1 setvar "external_wired.ip.protocol" "value"</pre> <i>Values:</i> "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 <i>Default:</i> "all"

* 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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar *	This command instructs the printer to return whether the timeout checking is enabled on the external wired print server. <i>Format:</i> ! U1 getvar "external_wired.ip.timeout.enable"
setvar	This command instructs the printer to enable or disable the timeout checking on the external wired print server. <i>Format:</i> ! U1 setvar "external_wired.ip.timeout.enable" "value" <i>Values:</i> "off" = turns off the connection checking "on" = turns on the connection checking <i>Default:</i> "on"

* 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

Description This network setting refers to the number of seconds before the connection times out for the external wired print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar *	This command instructs the printer to respond with the time of the external wired print server, in seconds, before the connection times out. <i>Format:</i> <pre>! U1 getvar "external_wired.ip.timeout.value"</pre>
setvar	This command instructs the printer to set the time of the external wired print server, in seconds, before the connection times out. <i>Format:</i> <pre>! U1 setvar "external_wired.ip.timeout.value" "value"</pre> <i>Values:</i> "1" through "3600" <i>Default:</i> "300"

* 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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- 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.

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the IPv6 address of the SEH wired print server. <i>Format:</i> ! U1 getvar "external_wired.ip.v6.addr" <i>Values:</i> 8 group of four hexadecimal digits with a colon delimiter character set = A-F or 0-9 (39-character maximum)



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

Description This command retrieves the IPv6 gateway of the SEH wired print server. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- 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.

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the IPv6 gateway of the SEH wired print server. <i>Format:</i> ! U1 getvar "external_wired.ip.v6.gateway" <i>Values:</i> 8 group of four hexadecimal digits with a colon delimiter character set = A-F or 0-9 (39-character maximum)



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

Description This command retrieves the IPv6 address prefix length of the SEH wired print server. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- 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.

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the IPv6 address prefix length of the SEH wired print server. <i>Format:</i> ! U1 getvar "external_wired.ip.v6.prefix_length" <i>Values:</i> character set = 0-9 (3-character maximum)



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

Description This command retrieves the MAC address of the external wired print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the command for this format:

Commands	Details
getvar	This command instructs the printer to respond with the MAC address of the external wired print server. <i>Format: ! U1 getvar "external_wired.mac_addr"</i>



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

Description This command specifies the RAW MAC address of the external print server. The raw mac address is the mac address without the colons (":"). For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the RAW MAC address of the external print server. <i>Format:</i> ! 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"
```

internal_wired.auto_switchover

Description 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. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- ZM400
- ZM600



Important • For this command to work, be sure:

- you are using a ZM400/ZM600 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 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.

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the current automatic switchover value. <i>Format:</i> ! U1 getvar "internal_wired.auto_switchover"
setvar	This command configures switches between the wireless and wired interfaces. <i>Format:</i> ! U1 setvar "internal_wired.auto_switchover" "value" <i>Values:</i> "on" = switchover enabled "off" = switchover disabled <i>Default:</i> "off"



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".

internal_wired.ip.addr

Description This command allows you to get or set the internal wired print servers’s IP address. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Important • For a set IP address to take affect, the IP protocol must be set to permanent and the print server must be reset.



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with its current internal wired print server IP address. <i>Format:</i> ! U1 getvar "internal_wired.ip.addr"
setvar	This command instructs the printer to change its current internal wired print server IP address upon powering the printer on. <i>Format:</i> ! U1 setvar "internal_wired.ip.addr" "value" <i>Values:</i> any valid IP address <i>Default:</i> "0.0.0.0"



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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the ARP interval or the ARP cache time out value for the internal wired print server. <i>Format:</i> ! U1 getvar "internal_wired.ip.arp_interval"
setvar	This command instructs the printer to change the ARP interval or the ARP cache time out for the internal wired print server. <i>Format:</i> ! U1 setvar "internal_wired.ip.arp_interval" "value" <i>Values:</i> 0 - 30 <i>Default:</i> "0"



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

Description This command allows you to default the internal wired print server’s IP address. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Important • For a set IP address to take affect, the IP protocol must be set to permanent and the print server must be reset.



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to show the status of the setting of internal wired print server’s default IP address feature. <i>Format:</i> ! U1 getvar "internal_wired.ip.default_addr_enable"
setvar	This command tells the printer to use it’s default address, 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. <i>Format:</i> ! U1 setvar "internal_wired.ip.default_addr_enable" "value" <i>Values:</i> "on" = enabled "off" = disabled <i>Default:</i> "on"



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.cache_ip

Description This command enables or disables the IP cache on the internal wired print server. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- ZM400
- ZM600

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the status of the IP cache on the internal wired print server. <i>Format:</i> ! U1 getvar "internal_wired.ip.dhcp.cache_ip"
setvar	This command sets the status of the IP cache. <i>Format:</i> ! U1 setvar "internal_wired.ip.dhcp.cache_ip" "value" <i>Values:</i> "on" = enabled "off" = disabled <i>Default:</i> "off"



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

Description 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". For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the client identifier prefix and suffix of the internal wired print server. <i>Format:</i> ! U1 getvar "internal_wired.ip.dhcp.cid_all"
setvar	This command instructs the printer to change the client identifier prefix and suffix of the internal wired print server. The prefix gets cleared and the suffix contains the entire client identifier. <i>Format:</i> ! U1 setvar "internal_wired.ip.dhcp.cid_all" "value" <i>Values:</i> A maximum length of 60 characters if the CID type is ASCII, or 120 characters if the CID type is hexadecimal. <i>Default Value:</i> ""



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

Description This command determines if DHCP (option 61) is turned on or off of the internal wired print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the status of the client identifier of the internal wired print server. Format: ! U1 getvar "internal_wired.ip.dhcp.cid_enable"
setvar	This command instructs the printer to set the status of the client identifier of the internal wired print server. Format: ! U1 setvar "internal_wired.ip.dhcp.cid_enable" "value" Values: "off" = client identifier is turned off "on" = client identifier is turned on Default: "off"



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

Description 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". For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the client identifier prefix of the internal wired print server. <i>Format:</i> ! U1 getvar "internal_wired.ip.dhcp.cid_prefix"
setvar	This command instructs the printer to change the CID prefix of the internal wired print server. <i>Format:</i> ! U1 setvar "internal_wired.ip.dhcp.cid_prefix" "value" <i>Values:</i> Any text string up to 10 characters if the CID type is ASCII, or 20 characters if the CID type is hexadecimal. <i>Default Value:</i> ""



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

Description 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". For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the client identifier suffix of the internal wired 10/100 print server. <i>Format:</i> <pre>! U1 getvar "internal_wired.ip.dhcp.cid_suffix"</pre>
setvar	This command instructs the printer to change the client identifier suffix value of the internal wired 10/100 print server. <i>Format:</i> <pre>! U1 setvar "internal_wired.ip.dhcp.cid_suffix" "value"</pre> <i>Values:</i> 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. <i>Default:</i> ""



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

Description 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". For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the client identifier type for the internal wired print server. <i>Format:</i> ! U1 getvar "internal_wired.ip.dhcp.cid_type"
setvar	This command instructs the printer to enable "synthetic" Client Identifier for the internal wired print server. <i>Format:</i> ! U1 setvar "internal_wired.ip.dhcp.cid_type" "value" <i>Values:</i> "0" = ASCII string "1" = wired print server's MAC address "2" = HEX value <i>Default Value:</i> "1"



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

Description This command retrieves the last time a DHCP request was sent from the internal wired print server. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- ZM400
- ZM600

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the last time a DHCP request was sent from the internal wired print server. <i>Format:</i> <pre>! U1 getvar "internal_wired.ip.dhcp.lease.last_attempt"</pre>



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

Description This command retrieves the original length (in seconds) of the DHCP lease on the internal wired print server. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- ZM400
- ZM600

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the original length (in seconds) of the DHCP lease on the internal wired print server. <i>Format:</i> ! 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

Description This command retrieves the address of the server that provided the DHCP lease on the internal wired print server. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- ZM400
- ZM600

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the address of the server that provided the DHCP lease on the internal wired print server. <i>Format:</i> ! 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

Description This command retrieves the time (in seconds) left in the current DHCP lease on the internal wired print server. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- ZM400
- ZM600

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the time (in seconds) left in the current DHCP lease on the internal wired print server. <i>Format:</i> ! 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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- ZM400
- ZM600

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the status of the DHCP option 12 (host name) in the discovery packet of the internal wired print server. <i>Format:</i> ! U1 getvar "internal_wired.ip.dhcp.option12"
setvar	This command instructs the printer to set the DHCP option 12 (host name) in the discovery packet of the internal wired print server. <i>Format:</i> ! U1 setvar "internal_wired.ip.dhcp.option12" "values" <i>Values:</i> "on" = turns on option 12 "off" = turns off option 12 <i>Default Value:</i> "on"



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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- ZM400
- ZM600

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves 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. <i>Format:</i> ! U1 getvar "internal_wired.ip.dhcp.option12_format"
setvar	This command instructs 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. <i>Format:</i> ! U1 setvar "internal_wired.ip.dhcp.option12_format" "value" <i>Values:</i> 0 to 109 alphanumeric characters <i>Default Value:</i> ""



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 "ShipPrinter", then the response to following command would be "ShipPrinter":

```
! U1 getvar "internal_wired.ip.dhcp.option12_value"
```

internal_wired.ip.dhcp.option12_value

Description This command retrieves the actual value which will be used in the discovery packet of the internal wired print server. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- ZM400
- ZM600

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the actual value which will be used in the discovery packet of the internal wired print server. <i>Format:</i> ! 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 "ShipPrinter", then the response to following command would be "ShipPrinter":

```
! U1 getvar "internal_wired.ip.dhcp.option12_value"
```

internal_wired.ip.dhcp.request_timeout

Description This command retrieves the maximum amount of time to wait (in seconds) for a response for a DHCP discover requests on the internal print server. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- ZM400
- ZM600

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the currently set maximum amount of time to wait for a response before timing out DHCP discover requests. <i>Format:</i> ! U1 getvar "internal_wired.ip.dhcp.request_timeout"
setvar	This command instructs the printer to set the maximum amount of time to wait for a response before timing out DHCP discover requests. <i>Format:</i> ! U1 setvar "internal_wired.ip.dhcp.request_timeout" "value" <i>Values:</i> 2-30 <i>Default:</i> "2"



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".

internal_wired.ip.dhcp.requests_per_session

Description This command retrieves the maximum amount of DHCP discover requests for a single DHCP session on the internal wired print server. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- ZM400
- ZM600

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the currently set maximum amount of DHCP discover requests for a single DHCP session on the internal wired print server. <i>Format:</i> ! U1 getvar "internal_wired.ip.dhcp.requests_per_session"
setvar	This command instructs the printer to set the maximum amount of DHCP discover requests for a single DHCP session on the internal wired print server. <i>Format:</i> ! U1 setvar "internal_wired.ip.dhcp.requests_per_session" "value" <i>Values:</i> 1-10 <i>Default:</i> "2"



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.dhcp.session_interval

Description This command retrieves how long it will take (in seconds) for a DHCP session to time out before a new DHCP session begins. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- ZM400
- ZM600

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 718](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the current DHCP session time out. <i>Format:</i> ! U1 getvar "internal_wired.ip.dhcp.session_interval"
setvar	This command instructs the printer to set the DHCP session time out. <i>Format:</i> ! U1 setvar "internal_wired.ip.dhcp.session_interval" "value" <i>Values:</i> 0-60 <i>Default:</i> "10"



Example • This setvar example shows the value set to "10".

```
! U1 setvar "internal_wired.ip.dhcp.session_interval" "10"
```

When the setvar value is set to "10", the getvar result is "10".

internal_wired.ip.gateway

Description This command instructs the printer to change the internal wired print servers gateway address. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Important • This setting refers to the gateway address. A set value is ignored if the IP protocol is not set to permanent.



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the internal wired printer servers gateway address. <i>Format:</i> ! U1 getvar "internal_wired.ip.gateway"
setvar	This command instructs the printer to change the internal wired printer servers gateway address. <i>Format:</i> ! U1 setvar "internal_wired.ip.gateway" "value" <i>Values:</i> Any valid gateway address <i>Default:</i> "0.0.0.0"



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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with internal wired print servers subnet mask. <i>Format:</i> ! U1 getvar "internal_wired.ip.netmask"
setvar	This command instructs the printer to change the internal wired print servers subnet mask. <i>Format:</i> ! U1 setvar "internal_wired.ip.netmask" "value" <i>Values:</i> Any valid subnet mask. <i>Default:</i> "255.255.255.0"



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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the internal wired printer servers TCP/UDP port number. <i>Format:</i> ! U1 getvar "internal_wired.ip.port"
setvar	This command instructs the printer to set the internal wired print servers TCP/UDP port number. <i>Format:</i> ! U1 setvar "internal_wired.ip.port" "value" <i>Values:</i> 1 - 65535 (excluding any ports currently used by other services, such as 21, 23, 80, and 515). <i>Default:</i> "9100"



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.protocol

Description This command configures the IP addressing method used by the internal wired print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command returns the IP addressing method used by the internal wired print server. <i>Format:</i> ! U1 getvar "internal_wired.ip.protocol"
setvar	This command instructs the printer to configure the IP addressing method used by the internal wired print server. <i>Format:</i> ! U1 setvar "internal_wired.ip.protocol" "value" <i>Values:</i> "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 <i>Default:</i> "all"



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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to return whether the timeout checking is enabled on the internal wired print server. <i>Format:</i> ! U1 getvar "internal_wired.ip.timeout.enable"
setvar	This command instructs the printer to enable or disable the timeout checking on the internal wired print server. <i>Format:</i> ! U1 setvar "internal_wired.ip.timeout.enable" "value" <i>Values:</i> "off" = turns off the connection checking "on" = turns on the connection checking <i>Default:</i> "on"



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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the time of the internal wired print server, in seconds, before the connection times out. <i>Format:</i> ! U1 getvar "internal_wired.ip.timeout.value"
setvar	This command instructs the printer to set the time of the internal wired print server, in seconds, before the connection times out. <i>Format:</i> ! U1 setvar "internal_wired.ip.timeout.value" "value" <i>Values:</i> "1" through "3600" <i>Default:</i> "300"



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.mac_addr

Description This command retrieves the MAC address of the internal wired print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command instructs the printer to respond with the MAC address of the internal wired print server. <i>Format: ! U1 getvar "internal_wired.mac_addr"</i>



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

Description This command identifies the RAW MAC address of the internal wired print server. The raw mac address is the mac address without the colons (" : "). For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- ZM400
- ZM600

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the RAW MAC address of the internal wired print server. <i>Format: ! U1 getvar "internal_wired.mac_raw"</i>



Example • In this example, the `getvar` retrieves the RAW MAC address of the internal wired print server.

```
! U1 getvar "internal_wired.mac_raw"
```



Notes • _____



SGD Wireless Commands

This chapter provides a high-level overview of the wireless Set / Get / Do (SGD) commands and details on each SGD command. For printer support of these SGD commands, see [SGD Command Support on page 997](#).



SGD commands are available in printers with firmware version V60.15.x, V50.15.x, V61.15.x, V56.15.x, V53.15.xZ, or later.



Note • The 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.



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.



Note • SGD commands must be terminated by a carriage return or a space and line feed.

SGD commands are commands that allow you to configure all printers with firmware version V60.15.x, V50.15.x, V61.15.x, V56.15.x, V53.15.xZ, or later. The printer performs the specified function immediately after receiving the command. The commands are:

- **setvar**
- **getvar**
- **do**

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)



Important • The setvar command and attributes must be specified in lower case.

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



Important • The printer settings and attributes must be specified in lower case.

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.

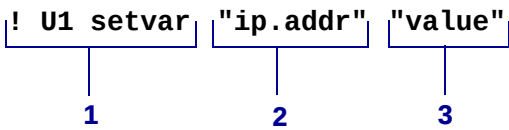


Important • The values must be specified in lower case.

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.

➔ **Example** • This is an example of a command structure illustration:



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).

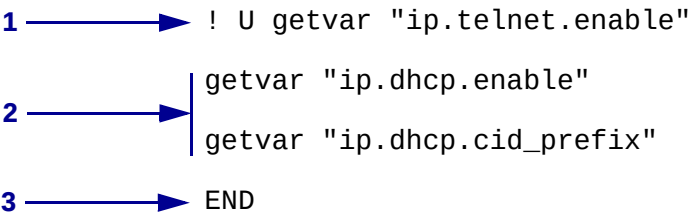


Note • Some RF cards do not support all of the SGD commands.

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.

➔ **Example** • This syntax shows how you can send multiple `getvar` commands:



1	The command portion of the string does not use the "1" after the "! U".
2	Commands issued after the first command do not require the "! U".
3	The string of commands is terminated by the word "END" with a space after the word, and by a carriage return/ line feed.

bluetooth.address

Description This command returns the printer’s Bluetooth device address. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the printer’s Bluetooth address. <i>Format: ! U1 getvar "bluetooth.address"</i>



Example • In this example, the `getvar` command causes the printer to return the printer’s Bluetooth address.

```
! U1 getvar "bluetooth.address"
```

bluetooth.afh_map

Description This command sets or retrieves the default AFH (adaptive frequency hopping) channel map (Bluetooth radios 1.2 and later); 20 bytes. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves default AFH channel map. <i>Format:</i> ! U1 getvar "bluetooth.afh_map"
setvar	This command selectively enables or disables individual Bluetooth channels for use when AFH mode is set to "on". <i>Format:</i> ! U1 setvar "bluetooth.afh_map" "value" <i>Values:</i> 20-byte string of hexadecimal characters <i>Default:</i> "7FFFFFFFFFFFFFFFFFFFFFFF"



Example • This setvar example shows the value set to "7FFFFFFFFFFFFFFFFFFFFFFF".
! U1 setvar "bluetooth.afh_map" "7FFFFFFFFFFFFFFFFFFFFFFF"

What the setvar value is set to is the getvar result. In this example, the getvar result is "7FFFFFFFFFFFFFFFFFFFFFFF".

bluetooth.afh_map_curr

Description This command retrieves the current AFH (adaptive frequency hopping) channel map (Bluetooth radios 1.2 and later). For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the current AFH channel map. <i>Format: ! U1 getvar "bluetooth.afh_map_curr"</i>



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

Description This command sets or retrieves AFH (adaptive frequency hopping) mode setting (Bluetooth radios 1.2 and later). For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the current setting of the AFH mode. <i>Format:</i> ! U1 getvar "bluetooth.afh_mode"
setvar	This command enables and disables AFH mode. <i>Format:</i> ! U1 setvar "bluetooth.afh_mode" "value" <i>Values:</i> "on" = enables AFH mode "off" = disables AFH mode <i>Default:</i> "off"



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.authentication

Description This command sets or retrieves Bluetooth authentication mode and works in combination with the `bluetooth.bluetooth_pin`. For printer support, see [SGD Command Support on page 997](#).

Type `getvar;setvar`



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the current Bluetooth authentication mode. <i>Format:</i> <code>! U1 getvar "bluetooth.authentication"</code>
setvar	This command enables and disables Bluetooth authentication. <i>Format:</i> <code>! U1 setvar "bluetooth.authentication" "value"</code> <i>Values:</i> "off" = disables authentication (can connect to master device without PIN) "setpin" = enables authentication (requires PIN or passkey to connect to a master device) <i>Default:</i> "off"



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

Description This command is used to connect to the printer only when the command `bluetooth.authentication` is set to "setpin". For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the current Bluetooth pin. <i>Format:</i> ! U1 getvar "bluetooth.bluetooth_pin"
setvar	This command sets the Bluetooth pin value. <i>Format:</i> ! U1 setvar "bluetooth.bluetooth_pin" "value" <i>Values:</i> Any text string up to 16 characters <i>Default:</i> ""



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.date

Description This command shows the release date of the Bluetooth module. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command shows the release date of the Bluetooth module. <i>Format: ! U1 getvar "bluetooth.date"</i>



Example • In this example, the `getvar` command returns the release date of the Bluetooth module.

```
! U1 getvar "bluetooth.date"
```

bluetooth.discoverable

Description This command enables or disables the Bluetooth discoverable mode. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the current Bluetooth discoverable mode. <i>Format:</i> ! U1 getvar "bluetooth.discoverable"
setvar	This command enables or disables the Bluetooth discoverable mode. <i>Format:</i> ! U1 setvar "bluetooth.discoverable" "value" <i>Values:</i> "on" = enables Bluetooth discoverable mode "off" = disables Bluetooth discoverable mode <i>Default:</i> "on"



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

Description This command enables or disables the Bluetooth radio. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the current status of the Bluetooth radio. <i>Format: ! U1 getvar "bluetooth.enable"</i>
setvar	This command enables or disables the Bluetooth radio. <i>Format: ! U1 setvar "bluetooth.enable"</i> <i>Values:</i> "on" = enables the Bluetooth radio "off" = disables the Bluetooth radio <i>Default: "on"</i>



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.friendly_name

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the current Bluetooth discoverable mode. <i>Format:</i> ! U1 getvar "bluetooth.friendly_name"
setvar	This command sets the Bluetooth discoverable mode. <i>Format:</i> ! U1 setvar "bluetooth.friendly_name" "value" <i>Values:</i> Any text string up to 17 characters



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.local_name

Description This command retrieves the local name that is provided during service discovery. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the local name that is provided during service discovery. <i>Format: ! U1 getvar "bluetooth.local_name"</i>



Example • In this example, the `getvar` command returns the local name that is provided during service discovery.

```
! U1 getvar "bluetooth.local_name"
```

bluetooth.radio_auto_baud

Description This command retrieves the Bluetooth radio data rate. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves Bluetooth radio data rate. <i>Format:</i> ! 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

Description This command returns the version of the currently installed Bluetooth radio. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command returns the version of the currently installed Bluetooth radio. <i>Format: ! U1 getvar "bluetooth.radio_version"</i>



Example • In this example, the `getvar` command returns the currently installed Bluetooth radio.

```
! U1 getvar "bluetooth.radio_version"
```

bluetooth.short_address

Description This command shortens the Bluetooth address by removing the colons (":"). For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the shortened Bluetooth address. <i>Format:</i> ! U1 getvar "bluetooth.short_address"



Example • In this example, the `getvar` retrieves the short Bluetooth address.

! U1 getvar "bluetooth.short_address"

bluetooth.version

Description This command returns the Bluetooth library version number. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command returns the Bluetooth library version number. <i>Format: ! U1 getvar "bluetooth.version"</i>



Example • In this example, the `getvar` command returns the Bluetooth library version number.

```
! U1 getvar "bluetooth.version"
```

card.mac_addr

Description This command retrieves the MAC address of the wireless radio card. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command instructs the printer to respond with the MAC address. <i>Format: ! U1 getvar "card.mac_addr"</i>



Example • In this example, the **getvar** result is the MAC address for the wireless radio card.

```
! U1 getvar "card.mac_addr"
```

card.inserted

Description This command indicates whether the wireless radio card is or is not inserted. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command instructs the printer to respond with the wireless radio card status. It's inserted or it's not inserted. <i>Format: ! U1 getvar "card.inserted"</i>



Example • In this example, the getvar result is "Inserted".
! U1 getvar "card.inserted"

comm.type

Description This printer setting determines the behavior of the serial port interface. It selects one of three serial communication states: DTE, DCE or Autodetect. For printer support, see [SGD Command Support on page 997](#).

Supported Devices

- LP 2824 Plus

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command instructs the printer to respond with the currently set serial port interface type. <i>Format: ! U1 getvar "comm.type"</i>
setvar	This command instructs the printer to change the serial port interface type. <i>Format: ! U1 setvar "comm.type" "value"</i> <i>Values:</i> "auto" = Autodetect "dte" = Force DTP (Tx on pin 2) "dcs" = Force DCE (Rx on pin 2) <i>Default: "auto"</i>

➔ **Example** • In this example, the getvar retrieves the serial port communications state.

```
! U1 getvar "comm.type"
```

➔ **Example** • This setvar example sets the communications port state to Autodetect.

```
! U1 setvar "comm.type" "auto"
```

ip.active_network

Description This command displays if the printer is actively connected to wireless, external wired, or internal wired. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with what the printer is currently connected to internal wired, wireless, external wired, or unknown. Table 24 provides details on the potential return values. <i>Format:</i> ! U1 getvar "ip.active_network"



Example • In this example, the getvar will return the current active network the printer is connected to.

```
! U1 getvar "ip.active_network"
```

Table 24 • Printer Responses

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

Description This command allows you to get or set the printer’s IP address. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Important • For a set IP address to take affect, the IP protocol must be set to permanent and the print server must be reset.



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with its current IP address. <i>Format:</i> ! U1 getvar "ip.addr"
setvar	This command instructs the printer to change its current IP address upon powering the printer on. <i>Format:</i> ! U1 setvar "ip.addr" "value" <i>Values:</i> any valid IP address <i>Default:</i> "0.0.0.0"



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.arp_interval

Description This printer setting allows you to specify the ARP (Address Resolution Protocol) interval or the ARP cache time out. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the ARP interval or the ARP cache time out value in seconds. <i>Format:</i> ! U1 getvar "ip.arp_interval"
setvar	This command instructs the printer to change the ARP interval or the ARP cache time out. <i>Format:</i> ! U1 setvar "ip.arp_interval" "value" <i>Values:</i> 0 - 30 <i>Default:</i> "0"



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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • If you are using static IP addressing, the IP protocol must be set to permanent.



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the current BOOTP setting. <i>Format:</i> ! U1 getvar "ip.bootp.enable"
setvar	This command instructs the printer to turn BOOTP on or off. <i>Format:</i> ! U1 setvar "ip.bootp.enable" "value" <i>Values:</i> "off" = printer does not use BOOTP to get the IP address "on" = printer uses BOOTP to get the IP address <i>Default:</i> "on"



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.cache_ip

Description This command enables or disables the IP caching. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the status of the IP cache. <i>Format:</i> ! U1 getvar "ip.dhcp.cache_ip"
setvar	This command sets the status of the IP cache. <i>Format:</i> ! U1 setvar "ip.dhcp.cache_ip" "value" <i>Values:</i> "on" = enabled "off" = disabled <i>Default:</i> "off"



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

Description 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". For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the client identifier prefix and suffix. <i>Format:</i> ! U1 getvar "ip.dhcp.cid_all"
setvar	This command instructs the printer to change the CID prefix and suffix. <i>Format:</i> ! U1 setvar "ip.dhcp.cid_all" "value" <i>Values:</i> A maximum length of 60 characters if the CID type is ASCII, or 120 characters if the CID type is hexadecimal. <i>Default Value:</i> ""



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

Description This command determines if DHCP (option 61) is turned on or off. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the status of the client identifier. <i>Format:</i> ! U1 getvar "ip.dhcp.cid_enable"
setvar	This command instructs the printer to set the status of the client identifier. <i>Format:</i> ! U1 setvar "ip.dhcp.cid_enable" "value" <i>Values:</i> "off" = client identifier is turned off "on" = client identifier is turned on <i>Default:</i> "off"



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

Description 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". For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the client identifier prefix. <i>Format:</i> ! U1 getvar "ip.dhcp.cid_prefix"
setvar	This command instructs the printer to change the CID prefix. <i>Format:</i> ! U1 setvar "ip.dhcp.cid_prefix" "value" <i>Values:</i> Any text string up to 10 characters if the CID type is ASCII, or 20 characters if the CID type is hexadecimal. <i>Default Value:</i> ""



Example • This setvar example shows the value set to "PRT001".

```
! U1 setvar "ip.dhcp.cid_prefix" "PRT001"
```

What the setvar value is set to is the getvar result. In this example, the getvar result is "PRT001".

ip.dhcp.cid_suffix

Description 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". For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the client identifier suffix. <i>Format:</i> ! U1 getvar "ip.dhcp.cid_suffix"
setvar	This command instructs the printer to change the CID value. <i>Format:</i> ! U1 setvar "ip.dhcp.cid_suffix" "value" <i>Values:</i> 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. <i>Default Value:</i> ""



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.lease.length

Description This command retrieves the original length of the DHCP lease on the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the original length of the DHCP lease on the wireless print server. <i>Format: ! U1 getvar "ip.dhcp.lease.length"</i>



Example • This command retrieves the original length of the DHCP lease on the wireless print server.

```
! U1 getvar "ip.dhcp.lease.length"
```

```
"1296000"
```


ip.dhcp.cid_type

Description 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". For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the client identifier type. <i>Format:</i> ! U1 getvar "ip.dhcp.cid_type"
setvar	This command instructs the printer to enable "synthetic" Client Identifier. <i>Format:</i> ! U1 setvar "ip.dhcp.cid_type" "value" <i>Values:</i> "0" = ASCII string "1" = wireless radio card's MAC address "2" = HEX value <i>Default Value:</i> "1"

➔ **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.enable

Definition 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • If you are using static IP addressing, the IP protocol must be set to permanent.



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the DHCP status. <i>Format:</i> ! U1 getvar "ip.dhcp.enable"
setvar	This command instructs the printer to turn DHCP on or off. <i>Format:</i> ! U1 setvar "ip.dhcp.enable" "value" <i>Values:</i> "off" = printer does not use DHCP to get the IP address "on" = printer uses DHCP to get the IP address <i>Default:</i> "on"



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

Description This command retrieves the time from the DHCP server of when the last DHCP request was sent. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the last time a DHCP request was sent. <i>Format: ! U1 getvar "ip.dhcp.lease.last_attempt"</i>



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

Description This command retrieves the address of the server that provided the DHCP lease on the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the address of the server that provided the DHCP lease on the wireless print server. <i>Format: ! U1 getvar "ip.dhcp.lease.server"</i>



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

Description This command retrieves the time left in the current DHCP lease on the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the time left in the current DHCP lease on the wireless print server. <i>Format: ! U1 getvar "ip.dhcp.lease.time_left"</i>



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.option12

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the status of the DHCP option 12 (host name) in the discovery packet that is sent from the wireless print server. <i>Format:</i> ! U1 getvar "ip.dhcp.option12"
setvar	This command instructs the printer to set the DHCP option 12 (host name) in the discovery packet that is sent from the wireless print server. <i>Format:</i> ! U1 setvar "ip.dhcp.option12" "value" <i>Values:</i> "on" = turns on option 12 "off" = turns off option 12 <i>Default Value:</i> "on"



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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the value which will be used for option 12 (host name) to be used in the DHCP discovery packet of the wireless print server. <i>Format:</i> ! U1 getvar "ip.dhcp.option12_format"
setvar	This command instructs 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. <i>Format:</i> ! U1 setvar "ip.dhcp.option12_format" "value" <i>Values:</i> string <i>Default Value:</i> ""



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

Description This command retrieves the actual value which will be used in the discovery packet of the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the actual value which will be used in the discovery packet of the wireless print server. <i>Format: ! U1 getvar "ip.dhcp.option12_value"</i>



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

Description This command retrieves the maximum amount of time (in seconds) for a DHCP discovery requests on the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the currently set the amount of time (in seconds) to wait before timing out a DHCP discovery request. <i>Format:</i> ! U1 getvar "ip.dhcp.request_timeout"
setvar	This command instructs the printer to set the amount of time (in seconds) to wait before timing out a DHCP discovery request. <i>Format:</i> ! U1 setvar "ip.dhcp.request_timeout" "value" <i>Values:</i> "2" through "30" <i>Default:</i> "2"



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

Description This command retrieves the maximum amount of DHCP discovery requests for a single DHCP session on the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the currently set maximum amount of DHCP discovery requests for a single DHCP session on the wireless print server. <i>Format:</i> ! U1 getvar "ip.dhcp.requests_per_session"
setvar	This command instructs the printer to set the maximum amount of DHCP discovery requests for a single DHCP session on the wireless print server. <i>Format:</i> ! U1 setvar "ip.dhcp.requests_per_session" "value" <i>Values:</i> "1" through "10" <i>Default:</i> "2"



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

Description This command configures the time interval (in seconds) before a new DHCP session is started on the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the current DHCP session time out value (in seconds). <i>Format:</i> ! U1 getvar "ip.dhcp.session_interval"
setvar	This command instructs the printer to set the DHCP session time out value (in seconds). <i>Format:</i> ! U1 setvar "ip.dhcp.session_interval" "value" <i>Values:</i> "0" through "60" <i>Default:</i> "10"



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.dns.domain

Description This command identifies the network domain of the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the network domain of the wireless print server. <i>Format:</i> ! 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

Description This command retrieves a space delimited list of the domain name servers from a wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves a list of space delimited DNS wireless print servers. <i>Format: ! U1 getvar "ip.dns.servers"</i>



Example • In this example, the `getvar` retrieves a list of space delimited DNS wireless print servers.

```
! U1 getvar "ip.dns.servers"
```

ip.ftp.enable

Description This printer setting refers to the FTP protocol setting. This command tells the printer to turn FTP on or off. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the FTP status. <i>Format: ! U1 getvar "ip.ftp.enable"</i>
setvar	This command instructs the printer to turn FTP on or off. <i>Format: ! U1 setvar "ip.ftp.enable" "value"</i> <i>Values:</i> "off" = disables FTP "on" = enables FTP <i>Default: "on"</i>



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

Description This command setting controls the printers' ability to process or not process commands received via the FTP protocol using the printers ZPL engine. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the FTP processing ability status. <i>Format:</i> ! U1 getvar "ip.ftp.execute_file"
setvar	This command instructs the printer to turn FTP processing ability on or off. <i>Format:</i> ! U1 setvar "ip.ftp.execute_file" "value" <i>Values:</i> "off" = disables the printer's ability to process FTP commands "on" = enables the printer's ability to process FTP commands <i>Default:</i> "on"



Example • This setvar example shows the FTP processing ability set to "on".
 ! U1 setvar "ip.ftp.execute_file" "on"

ip.gateway

Description This command instructs the printer to change the gateway address. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Important • This setting refers to the gateway address. A set value is ignored if the IP protocol is not set to permanent.



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the gateway address. <i>Format:</i> ! U1 getvar "ip.gateway"
setvar	This command instructs the printer to change the gateway address. <i>Format:</i> ! U1 setvar "ip.gateway" "value" <i>Values:</i> Any valid gateway address <i>Default:</i> "0.0.0.0"



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.enable

Description This printer setting refers to the HTTP protocol/web server setting. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the HTTP status. <i>Format:</i> ! U1 getvar "ip.http.enable"
setvar	This command instructs the printer to change HTTP to on or off. <i>Format:</i> ! U1 setvar "ip.http.enable" "value" <i>Values:</i> "off" = disables HTTP protocol "on" = enables HTTP protocol <i>Default:</i> "on"



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.lpd.enable

Description This printer setting refers to the LPD (Line Printer Daemon) protocol setting. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Important • LPD communications from the host should be directed to port 515.



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the LPD status. <i>Format:</i> ! U1 getvar "ip.lpd.enable"
setvar	This command instructs the printer to turn LPD on or off. <i>Format:</i> ! U1 setvar "ip.lpd.enable" "value" <i>Values:</i> "off" = disables LPD protocol "on" = enables LPD protocol <i>Default:</i> "on"



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

Description This command specifies the RAW MAC address of the wireless print server. The raw mac address is the mac address without the colons (":"). For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the RAW MAC address of the wireless print server. <i>Format:</i> ! 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.auto



Description This command enables the ability to automatically perform a mirror update (fetch) command on power up. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command will report whether the printer will perform a mirror update (fetch) automatically on power up. <i>Format:</i> ! U1 getvar "ip.mirror.auto"
setvar	This command instructs the printer 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". <i>Format:</i> ! U1 setvar "ip.mirror.auto" "values" <i>Values:</i> "on" = turns on the auto mirroring feature "off" = turns off the auto mirroring feature <i>Default Value:</i> "off"



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



Description This command identifies how many times mirroring is retried when an error occurs. For printer support, see [SGD Wireless Commands on page 769](#).

Type getvar;setvar



Note • For details on SGD command structure, see [SGD Command Support on page 997](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the number of times the printer retries mirroring when an error occurs. <i>Format: ! U1 getvar "ip.mirror.error_retry"</i>
setvar	This command instructs the printer to set the required times that mirroring retries when an error occurs. <i>Format: ! U1 setvar "ip.mirror.error_retry" "value"</i> <i>Values: numeric values (0 - 65535)</i> <i>Default Value: "0"</i>



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



Description This command identifies if a feedback file is pushed to the mirroring server by the printer when a mirroring update (fetch) is complete. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the mirror feedback setting that the printer is currently using. <i>Format:</i> ! U1 getvar "ip.mirror.feedback.auto"
setvar	This command instructs the printer to set the mirror feedback feature to on or off. <i>Format:</i> ! U1 setvar "ip.mirror.feedback.auto" "value" <i>Values:</i> "on" = turns on mirror feedback "off" = turns off mirror feedback <i>Default Value:</i> "off"



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



Description This command specifies the time interval (in minutes) between performing feedback file uploads. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the number of minutes set to wait between feedback file uploads. <i>Format:</i> ! U1 getvar "ip.mirror.feedback.freq"
setvar	This command instructs the printer to set the number of minutes to wait between feedback file uploads. <i>Format:</i> ! U1 setvar "ip.mirror.feedback.freq" "value" <i>Values:</i> numeric value (0 - 65535) <i>Default Value:</i> "0"



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



Description This command instructs the printer to set the mirror feedback odometer. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the mirror feedback odometer. <i>Format:</i> ! U1 getvar "ip.mirror.feedback.odometer"
setvar	This command instructs the printer to set the odometer counter. <i>Format:</i> ! U1 setvar "ip.mirror.feedback.odometer" "values" <i>Values:</i> numeric value between 0 and 2632 <i>Default Value:</i> "0"



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



Description This command identifies where the feedback file is stored on the mirroring server. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the path on the mirroring sever that the printer is currently using to store the feedback file. <i>Format:</i> ! U1 getvar "ip.mirror.feedback.path"
setvar	This command instructs the printer to set the path on the mirroring server that stores the feedback file. <i>Format:</i> ! U1 setvar "ip.mirror.feedback.path" "value" <i>Values:</i> alphanumeric text (1 to 50 characters) <i>Default Value:</i> "Zebra/feedback"



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



Description This command forces a mirroring update sequence. For printer support, see [SGD Command Support on page 997](#).

Type do;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
do	This command forces a mirroring update sequence. <i>Format: ! U1 do "ip.mirror.fetch" ""</i>
setvar	This command forces a mirroring update sequence. <i>Format: ! U1 setvar "ip.mirror.fetch" ""</i>



Example • This setvar example forces a mirroring update sequence.

```
! U1 setvar "ip.mirror.fetch" ""
```

ip.mirror.freq



Description This command defines the frequency of mirroring updates (in minutes). For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).



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.

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the number of minutes to wait before performing another mirror update. <i>Format: ! U1 getvar "ip.mirror.freq"</i>
setvar	This command instructs the printer to set the mirror frequency. <i>Format: ! U1 setvar "ip.mirror.freq" "value"</i> <i>Values: "0" through "65535" (minutes)</i> <i>Default Value: "0" (disables this feature)</i>



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



Description This command defines the frequency of mirroring updates (in hours). For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the frequency of mirroring updates (in hours) that the printer is currently using. <i>Format: ! U1 getvar "ip.mirror.freq_hours"</i>
setvar	This command instructs the printer to set the frequency of mirroring updates (in hours). <i>Format: ! U1 setvar "ip.mirror.freq_hours" "values"</i> <i>Values: "0" through "100"</i> <i>Default Value: "0" (disables this feature)</i>



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.last_error



Description This command retrieves the last error encountered during a mirroring operation. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the error code of the last mirroring update (fetch). <i>Format:</i> ! 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



Description This command retrieves the timestamp, in seconds, of the last time the system attempted a mirror update (fetch). For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the timestamp, in seconds, of the last time the system attempted a mirror update (fetch). <i>Format: ! U1 getvar "ip.mirror.last_time"</i>



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.password



Description This command specifies the user password on the mirroring server assigned for mirroring updates (fetch). For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 628](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the user password the printer is currently using for mirroring updates (fetch). <i>Format: ! U1 getvar "ip.mirror.password"</i>
setvar	This command instructs the printer to use a specific password for mirroring updates (fetch). <i>Format: ! U1 setvar "ip.mirror.password" "value"</i> <i>Values:</i> alphanumeric text string (1 to 20 characters) <i>Default Value:</i> "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



Description This command identifies the base path on the FTP server where the mirror directory resides. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the base path of the FTP server where the mirror directory resides. <i>Format: ! U1 getvar "ip.mirror.path"</i>
setvar	This command sets the base path on the FTP server where the mirror directory resides. <i>Format: ! U1 setvar "ip.mirror.path" "value"</i> <i>Values:</i> alphanumeric text string (1 to 50 characters) <i>Default Value:</i> "zebra"



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.server



Description This command identifies the IP address of the mirroring server. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the IP address of the mirroring server. <i>Format:</i> ! U1 getvar "ip.mirror.server"
setvar	This command sets the IP address of the mirroring server. <i>Format:</i> ! U1 setvar "ip.mirror.server" "value" <i>Values:</i> a valid IP address <i>Default Value:</i> "127.0.0.1"



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



Description This command reports the success or failure of the last mirroring update (fetch). For printer support, see [SGD Wireless Commands on page 769](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the success or failure of the last mirroring update (fetch). <i>Format:</i> ! U1 getvar "ip.mirror.success" <i>Values:</i> "yes" = successful "no" = 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



Description This command provides the timestamp, in seconds, of the last time the system successfully completed a mirror update (fetch). For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the timestamp, in seconds, of the last time the system successfully completed a mirror update (fetch). <i>Format:</i> ! 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



Description This command specifies the user name on the mirroring server assigned for mirroring updates (fetch). For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the user name the printer is currently using for mirroring updates (fetch). <i>Format: ! U1 getvar "ip.mirror.username"</i>
setvar	This command instructs the printer to use a specific user name for mirroring updates (fetch). <i>Format: ! U1 setvar "ip.mirror.username" "value"</i> <i>Values: alphanumeric text string (1 to 20 characters)</i> <i>Default Value: "user"</i>



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



Description This command retrieves the mirror code build date. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the mirror code build date. <i>Format: ! U1 getvar "ip.mirror.version"</i>



Example • In this example, the `getvar` retrieves the mirror code build date.

```
! U1 getvar "ip.mirror.version"
```

ip.netmask

Description This setting refers to the subnet mask address. This value is ignored if the IP protocol is not set to permanent. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with subnet mask. <i>Format:</i> ! U1 getvar "ip.netmask"
setvar	This command instructs the printer to change the subnet mask. <i>Format:</i> ! U1 setvar "ip.netmask" "value" <i>Values:</i> Any valid subnet mask. <i>Default:</i> "255.255.255.0"



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.pop3.enable

Description This printer setting determines if the printer queries a POP3 mailbox for mail. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the POP3 status. <i>Format:</i> ! U1 getvar "ip.pop3.enable"
setvar	This command instructs the printer to turn POP3 on or off. <i>Format:</i> ! U1 setvar "ip.pop3.enable" "value" <i>Values:</i> "off" = disables POP3 "on" = enables POP3 <i>Default:</i> "on"



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

Description This printer setting refers to the POP3 mailbox password. This only applies if "ip.pop3.enable" is set to on. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the POP3 password. <i>Format:</i> ! U1 getvar "ip.pop3.password" For protection a single "*" prints.
setvar	This command instructs the printer to change the POP3 password. <i>Format:</i> ! U1 setvar "ip.pop3.password" "value" <i>Values:</i> A maximum of 20 alphanumeric characters <i>Default:</i> " "



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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



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.



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the POP3 poll frequency (in seconds). <i>Format: ! U1 getvar "ip.pop3.poll"</i>
setvar	This command instructs the printer to change the POP3 poll interval. A value of "0" causes the printer to only query the POP3 mailbox one time, on printer power up, or following a network reset. <i>Format: ! U1 setvar "ip.pop3.poll" "value"</i> <i>Values: "0" through "65535"</i> <i>Default: "0"</i>



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.server_addr

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the POP3 server address. <i>Format:</i> ! U1 getvar "ip.pop3.server_addr"
setvar	This command instructs the printer to change the POP3 server address. <i>Format:</i> ! U1 setvar "ip.pop3.server_addr" "value" <i>Values:</i> Any valid POP3 server address <i>Default:</i> "0.0.0.0"



Example • This setvar example shows the value set to "10.3.5.10".
! U1 setvar "ip.pop3.server_addr" "10.3.5.10"
When the setvar value is set to "10.3.5.10", the getvar result is "10.3.5.10".

ip.pop3.username

Description This printer setting refers to the POP3 user name. This only applies if the "ip.pop3.enable" is set to on. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the POP3 user name. <i>Format:</i> ! U1 getvar "ip.pop3.username"
setvar	This command instructs the printer to change the POP3 user name. <i>Format:</i> ! U1 setvar "ip.pop3.username" "value" <i>Values:</i> A maximum of 20 alphanumeric characters <i>Default:</i> " "



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.port

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the TCP/UDP port number. <i>Format:</i> ! U1 getvar "ip.port"
setvar	This command instructs the printer to set the TCP/UDP port number. <i>Format:</i> ! U1 setvar "ip.port" "value" <i>Values:</i> 1 - 65535 (excluding any ports currently used by other services, such as 21, 23, 80, and 515). <i>Default:</i> "9100"



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.primary_network

Description This command allows you to set the primary network to either wired or wireless. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the name of the current primary network device. <i>Format:</i> ! U1 getvar "ip.primary_network"
setvar	This command instructs the printer to set the primary network device. <i>Format:</i> ! U1 setvar "ip.primary_network" "value" <i>Values:</i> 1 = wired 2 = wireless <i>Default:</i> "1"



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

Description This printer setting refers to the domain name used by the printer in sending email with respect to the SMTP server. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to return the SMTP domain name. <i>Format:</i> ! U1 getvar "ip.smtp.domain"
setvar	This command instructs the printer to change the SMTP domain name. <i>Format:</i> ! U1 setvar "ip.smtp.domain" "value" <i>Values:</i> A maximum of 24 alphanumeric characters <i>Default:</i> "ZBRPrintServer"



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

Description This printer setting refers to the SMTP protocol. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to return the SMTP status. <i>Format: ! U1 getvar "ip.smtp.enable"</i>
setvar	This command instructs the printer to turn SMTP on or off. <i>Format: ! U1 setvar "ip.smtp.enable" "value"</i> <i>Values:</i> "off" = disables SMTP "on" = enables SMTP <i>Default: "on"</i>



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

Description This printer setting refers to the IP address of the SMTP server used for sending email. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the current SMTP server address. <i>Format: ! U1 getvar "ip.smtp.server_addr"</i>
setvar	This command instructs the printer to change the SMTP server address. <i>Format: ! U1 setvar "ip.smtp.server_addr" "value"</i> <i>Values: Any valid IP address.</i> <i>Default: 0.0.0.0</i>



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.get_community_name

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to get the SNMP get community name string. <i>Format:</i> ! U1 getvar "ip.snmp.get_community_name" For protection a single "*" prints.
setvar	This command instructs the printer to set the SNMP get community name string. <i>Format:</i> ! U1 setvar "ip.snmp.get_community_name" "value" <i>Values:</i> A maximum of 19 alphanumeric characters. <i>Default:</i> "public"



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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to return the printer's SNMP set community name string. <i>Format:</i> ! U1 getvar "ip.snmp.set_community_name" For protection a single "*" returns.
setvar	This command instructs the printer to set the SNMP set community name string. <i>Format:</i> ! U1 setvar "ip.snmp.set_community_name" "value" <i>Values:</i> A maximum of 19 alphanumeric characters <i>Default:</i> "public"



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.enable

Description This printer setting refers to the SNMP protocol. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the SNMP status. <i>Format:</i> ! U1 getvar "ip.snmp.enable"
setvar	This command instructs the printer to enable or disable the SNMP protocol. <i>Format:</i> ! U1 setvar "ip.snmp.enable" "value" <i>Values:</i> "on" = enable the SNMP protocol "off" = disable the SNMP protocol <i>Default:</i> "on"



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.telnet.enable

Description This printer setting refers to the TELNET (port 23) protocol. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the TELNET status. <i>Format:</i> ! U1 getvar "ip.telnet.enable"
setvar	This command instructs the printer to turn TELNET on or off. <i>Format:</i> ! U1 setvar "ip.telnet.enable" "value" <i>Values:</i> "off" = disables telnet protocol "on" = enables telnet protocol <i>Default:</i> "on"



Example • This setvar example shows the value set to "on".

```
! U1 setvar "ip.telnet.enable" "on"
```

When the setvar value is set to "on", the getvar result is "on".

ip.tcp.enable

Description This printer setting refers to the TCP socket protocol. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the TCP status. <i>Format: ! U1 getvar "ip.tcp.enable"</i>
setvar	This command instructs the printer to turn the TCP on or off. <i>Format: ! U1 setvar "ip.tcp.enable" "value"</i> <i>Values:</i> "off" = disables TCP protocol "on" = enables TCP protocol <i>Default: "on"</i>



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.udp.enable

Description This printer setting refers to the UDP socket protocol. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the UDP status. <i>Format: ! U1 getvar "ip.udp.enable"</i>
setvar	This command instructs the printer to turn UDP on or off. <i>Format: ! U1 setvar "ip.udp.enable" "value"</i> <i>Values:</i> "off" = disables UDP protocol "on" = enables UDP protocol <i>Default: "off"</i>



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".

wlan.adhocautomode

Description This printer setting refers to enabling or disabling the adhoc auto mode. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the adhoc auto mode status. <i>Format:</i> ! U1 getvar "wlan.adhocautomode"
setvar	This command instructs the printer to set the adhoc auto mode. <i>Format:</i> ! U1 setvar "wlan.adhocautomode" "value" <i>Values:</i> "on" = adhoc auto mode enabled "off" = adhoc auto mode disabled <i>Default:</i> "off"



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

Description This printer setting refers to specifying the wireless channel for adhoc channel. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the wireless channel for adhoc channel mode. <i>Format:</i> ! U1 getvar "wlan.adhocchannel"
setvar	This command instructs the printer to set the wireless channel for adhoc channel mode. <i>Format:</i> ! U1 setvar "wlan.adhocchannel" "value" <i>Values:</i> Decimal value between 1 and 16 inclusive <i>Default:</i> "1"



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

Description This command refers to if the printer is or is not associated with an access point (AP). For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with yes or no, which identifies if it is associated with the AP. <i>Format: ! U1 getvar "wlan.associated"</i>



Example • In this example, the getvar result is "yes".

```
! U1 getvar "wlan.associated"
```

wlan.bssid

Description This command returns the MAC address of the access point (AP) with which the printer is associated. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the MAC address of the access point (AP). <i>Format: ! U1 getvar "wlan.bssid"</i>



Example • In this example, the `getvar` result is the MAC address of the access point.

```
! U1 getvar "wlan.bssid"
```

wlan.channel

Description This command retrieves the current WI-FI channel the printer is using. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the current WI-FI channel the printer is using. <i>Format:</i> ! 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

Description This printer setting refers to specifying the wireless channel masks to enable and disable various channels. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the wireless channel mask value. <i>Format:</i> ! U1 getvar "wlan.channel_mask"
setvar	This command instructs the printer to set the wireless channel mask value. <i>Format:</i> ! U1 setvar "wlan.channel_mask" "value" <i>Values:</i> 4 Hexadecimal digits preceded by "0x" (0x0000 to 0xFFFF). For commonly used channel masks, see Table 25 . <i>Default:</i> "0x7FFF"

Table 25 • Channel Mask Settings

Region	Channel Mask
United States, Canada, Latin America	0x7FFF
Europe, Middle East, Africa, other	0x1FFF
Japan	0x3FFF



Example • This setvar example shows the value set to "0x7FFF".

```
! U1 setvar "wlan.channel_mask" "0x7FFF"
```

When the setvar value is set to "0x7FFF", the getvar result is "0x7FFF".

wlan.current_tx_rate

Description This command retrieves the transmission rate of the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command instructs the printer to retrieve the current transmit rate of the wireless print server. <i>Format:</i> ! 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.essid

Description This printer setting refers to the printer’s stored ESSID. Setting the ESSID to "" will set the printer in a "broadcast" mode. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the stored ESSID value. <i>Format:</i> ! U1 getvar "wlan.essid"
setvar	This command instructs the printer to change the ESSID. <i>Format:</i> ! U1 setvar "wlan.essid" "value" <i>Values:</i> 32 character alphanumeric string <i>Default:</i> "125"



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".

wlan.firmware_version

Description This command refers to the firmware version of the wireless radio card. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the current version of the wireless radio card firmware. <i>Format: ! U1 getvar "wlan.firmware_version"</i>



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

Description This command allows you to get or set the wireless print servers’s IP address. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Important • For a set IP address to take affect, the IP protocol must be set to permanent and the print server must be reset.



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with its current wireless print server IP address. <i>Format:</i> ! U1 getvar "wlan.ip.addr"
setvar	This command instructs the printer to change its current wireless print server IP address upon powering the printer on. <i>Format:</i> ! U1 setvar "wlan.ip.addr" "value" <i>Values:</i> any valid IP address <i>Default:</i> "0.0.0.0"



Note • The setvar value of this command can be affected by the wlan.ip.dhcp.enable command.



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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the ARP interval or the ARP cache time out value (in seconds) for the wireless print server. <i>Format:</i> ! U1 getvar "wlan.ip.arp_interval"
setvar	This command instructs the printer to respond with the ARP interval or the ARP cache time out value for the wireless print server. <i>Format:</i> ! U1 setvar "wlan.ip.arp_interval" "value" <i>Values:</i> 0 to 30 seconds <i>Default:</i> "0"



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

Description This command allows you to default the wireless print server’s IP address. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Important • For a set IP address to take affect, the IP protocol must be set to permanent and the print server must be reset.



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to show the status of the setting of the wireless print server’s default IP address feature. <i>Format:</i> <pre>! U1 getvar "wlan.ip.default_addr_enable"</pre>
setvar	This command tells the printer to use it’s default address, 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. <i>Format:</i> <pre>! U1 setvar "wlan.ip.default_addr_enable" "value"</pre> <i>Values:</i> "on" = enabled "off" = disabled <i>Default:</i> "on"



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.cache_ip

Description This command enables or disables the IP cache of the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the status of the IP cache. <i>Format:</i> ! U1 getvar "wlan.ip.dhcp.cache_ip"
setvar	This command sets the status of the IP cache. <i>Format:</i> ! U1 setvar "ip.dhcp.cache_ip" "value" <i>Values:</i> "on" = enabled "off" = disabled <i>Default:</i> "off"



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

Description 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". For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the client identifier prefix and suffix of the wireless print server. <i>Format:</i> ! U1 getvar "wlan.ip.dhcp.cid_all"
setvar	This command instructs the printer to change the client identifier prefix and suffix of the wireless print server. The prefix gets cleared and the suffix contains the entire client identifier. <i>Format:</i> ! U1 setvar "wlan.ip.dhcp.cid_all" "value" <i>Values:</i> A maximum length of 60 characters if the CID type is ASCII, or 120 characters if the CID type is hexadecimal. <i>Default Value:</i> ""



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

Description This command determines if DHCP (option 61) is turned on or off of the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the status of the client identifier of the wireless print server. <i>Format:</i> ! U1 getvar "wlan.ip.dhcp.cid_enable"
setvar	This command instructs the printer to set the status of the client identifier of the wireless print server. <i>Format:</i> ! U1 setvar "wlan.ip.dhcp.cid_enable" "value" <i>Values:</i> "off" = client identifier is turned off "on" = client identifier is turned on <i>Default:</i> "off"



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

Description 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". For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the client identifier prefix of the wireless print server. <i>Format:</i> ! U1 getvar "wlan.ip.dhcp.cid_prefix"
setvar	This command instructs the printer to change the CID prefix of the wireless print server. <i>Format:</i> ! U1 setvar "wlan.ip.dhcp.cid_prefix" "value" <i>Values:</i> Any text string up to 10 characters if the CID type is ASCII, or 20 characters if the CID type is hexadecimal. <i>Default Value:</i> ""



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

Description 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". For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the client identifier suffix on the wireless print server. <i>Format:</i> ! U1 getvar "wlan.ip.dhcp.cid_suffix"
setvar	This command instructs the printer to change the client identifier suffix value on the wireless print server. <i>Format:</i> ! U1 setvar "wlan.ip.dhcp.cid_suffix" "value" <i>Values:</i> 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. <i>Default:</i> ""



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

Description 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". For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the client identifier type for the wireless print server. <i>Format:</i> ! U1 getvar "wlan.ip.dhcp.cid_type"
setvar	This command instructs the printer to enable "synthetic" client identifier for the wireless print server. <i>Format:</i> ! U1 setvar "wlan.ip.dhcp.cid_type" "value" <i>Values:</i> "0" = ASCII string "1" = wireless radio card's MAC address "2" = HEX value <i>Default Value:</i> "1"



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

Description This command retrieves the last time a DHCP request was sent from the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the last time a DHCP request was sent from the wireless print server. <i>Format:</i> ! U1 getvar "wlan.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 "wlan.ip.dhcp.lease.last_attempt"
```

wlan.ip.dhcp.lease.length

Description This command retrieves the original length (in seconds) of the DHCP lease on the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the original length (in seconds) of the DHCP lease on the wireless print server. <i>Format: ! U1 getvar "wlan.ip.dhcp.lease.length"</i>



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

Description This command retrieves the address of the print server that provided the DHCP lease on the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the address if the print server that provided the DHCP lease on the wireless print server. <i>Format:</i> ! 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

Description This command retrieves the time (in seconds) left in the current DHCP lease on the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the time (in seconds) left in the current DHCP lease on the wireless print server. <i>Format: ! U1 getvar "wlan.ip.dhcp.lease.time_left"</i>



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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the status of the DHCP option 12 (host name) is on or off in the discovery packet that is sent from the wireless print server. <i>Format:</i> ! U1 getvar "wlan.ip.dhcp.option12"
setvar	This command instructs the printer to set the DHCP option 12 (host name) is on or off in the discovery packet that is sent from the wireless print server. <i>Format:</i> ! U1 setvar "wlan.ip.dhcp.option12" "value" <i>Values:</i> "on" = turns on option 12 "off" = turns off option 12 <i>Default Value:</i> "on"



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

Description This command specifies the format of the option 12 value to be used in the discovery packet of the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the format of option 12 value to be used in the discovery packet of the wireless print server. <i>Format:</i> ! U1 getvar "wlan.ip.dhcp.option12_format"
setvar	This command instructs the printer to set the format of option 12 value to be used in the discovery packet of the wireless print server. <i>Format:</i> ! U1 setvar "wlan.ip.dhcp.option12_format" "value" <i>Values:</i> string <i>Default Value:</i> ""



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.option12_value

Description This command retrieves the actual value which will be used in the discovery packet of the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the actual value which will be used in the discovery packet of the wireless print server. <i>Format: ! U1 getvar "wlan.ip.dhcp.option12_value"</i>



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

Description This command sets the maximum time (in seconds) to wait for a response to a DHCP discovery request on the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the maximum time (in seconds) to wait for a response to a DHCP discovery request on the wireless print server. <i>Format:</i> ! U1 getvar "wlan.ip.dhcp.request_timeout"
setvar	This command instructs the printer to set the maximum time (in seconds) to wait for a response to a DHCP discovery request on the wireless print server. <i>Format:</i> ! U1 setvar "wlan.ip.dhcp.request_timeout" "value" <i>Values:</i> "2" through "30" <i>Default:</i> "2"



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

Description This command retrieves the maximum amount of DHCP discover requests for a single DHCP session on the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the currently set maximum amount of DHCP discover requests for a single DHCP session on the wireless print server. <i>Format:</i> ! U1 getvar "wlan.ip.dhcp.requests_per_session"
setvar	This command instructs the printer to set the maximum amount of DHCP discover requests for a single DHCP session on the wireless print server. <i>Format:</i> ! U1 setvar "wlan.ip.dhcp.requests_per_session" "value" <i>Values:</i> "1" through "10" <i>Default:</i> "2"



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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the current DHCP session time out. <i>Format:</i> ! U1 getvar "wlan.ip.dhcp.session_interval"
setvar	This command instructs the printer to set the DHCP session time out. <i>Format:</i> ! U1 setvar "wlan.ip.dhcp.session_interval" "value" <i>Values:</i> "0" through "60" <i>Default:</i> "10"



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.gateway

Description This command instructs the printer to change the wireless print server’s gateway address. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • This setting refers to the gateway address. A set value is ignored if the IP protocol is not set to permanent.



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the wireless printer server’s gateway address. <i>Format:</i> ! U1 getvar "wlan.ip.gateway"
setvar	This command instructs the printer to change the wireless printer server’s gateway address. <i>Format:</i> ! U1 setvar "wlan.ip.gateway" "value" <i>Values:</i> Any valid gateway address <i>Default:</i> "0.0.0.0"



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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with wireless print server’s subnet mask. <i>Format:</i> ! U1 getvar "wlan.ip.netmask"
setvar	This command instructs the printer to change the wireless print servers’s subnet mask. <i>Format:</i> ! U1 setvar "wlan.ip.netmask" "value" <i>Values:</i> Any valid subnet mask. <i>Default:</i> "255.255.255.0"



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

Description 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.For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the wireless printer server’s TCP/UDP port number. <i>Format:</i> ! U1 getvar "wlan.ip.port"
setvar	This command instructs the printer to set the wireless print server’s TCP/UDP port number. <i>Format:</i> ! U1 setvar "wlan.ip.port" "value" <i>Values:</i> 1 - 65535 (excluding any ports currently used by other services, such as 21, 23, 80, and 515). <i>Default:</i> "9100"



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.protocol

Description This command configures the IP addressing method used by the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command returns the value of the currently selected IP protocol used by the wireless print server. <i>Format:</i> ! U1 getvar "wlan.ip.protocol"
setvar	This command instructs the printer to configure the IP addressing method used by the wireless print server. <i>Format:</i> ! U1 setvar "wlan.ip.protocol" "value" <i>Values:</i> "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 <i>Default:</i> "all"



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

Description This network setting refers to enabling the connection timeout on the wireless print server. For this to take effect, the print server must be reset. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to return whether the timeout checking is enabled on the wireless print server. <i>Format:</i> ! U1 getvar "wlan.ip.timeout.enable"
setvar	This command instructs the printer to enable or disable the timeout checking on the wireless print server. <i>Format:</i> ! U1 setvar "wlan.ip.timeout.enable" "value" <i>Values:</i> "off" = turns off the connection checking "on" = turns on the connection checking <i>Default:</i> "on"



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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the time, in seconds, before the connection times out. <i>Format:</i> ! U1 getvar "wlan.ip.timeout.value"
setvar	This command instructs the printer to set the the time value of the wireless print server, in seconds, before the connection times out. <i>Format:</i> ! U1 setvar "wlan.ip.timeout.value" "value" <i>Values:</i> "1" through "3600" <i>Default:</i> "300"



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.keep_alive.enable

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the wlan.keep_alive.enable setting. <i>Format: ! U1 getvar "wlan.keep_alive.enable"</i>
setvar	This command instructs the printer to send a LSAP (link service access point) packet to the access point on an user controllable interval. <i>Format: ! U1 setvar "wlan.keep_alive.enable" "value"</i> <i>Values:</i> "on" = turns on keep_alive "off" = turns off keep_alive <i>Default: "on"</i>



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

Description This printer setting manages the interval at which the LSAP (link service access point) packet is sent. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the wlan.keep_alive.timeout interval value. <i>Format:</i> ! U1 getvar "wlan.keep_alive.timeout"
setvar	This command instructs the printer to configure the frequency at which the printer sends the wlan.keep_alive packet. <i>Format:</i> ! U1 setvar "wlan.keep_alive.timeout" "value" <i>Values:</i> 5 to 300 seconds <i>Default:</i> "15"



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

Description 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). For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the current Kerberos KDC. <i>Format:</i> ! U1 getvar "wlan.kerberos.kdc"
setvar	This command instructs the printer to change the Kerberos KDC. <i>Format:</i> ! U1 setvar "wlan.kerberos.kdc" "value" <i>Values:</i> 0-32 ASCII characters <i>Default:</i> "krbtgt"



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.password

Description This printer setting refers to the Kerberos password. The password must correspond to a user profile established on the Kerberos KDC server in use. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the current Kerberos password. <i>Format:</i> ! U1 getvar "wlan.kerberos.password" For protection a single "*" prints.
setvar	This command instructs the printer to set the Kerberos password. <i>Format:</i> ! U1 setvar "wlan.kerberos.password" "value" <i>Values:</i> 0-32 alphanumeric characters <i>Default:</i> "password"



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

Description This printer setting refers to the Kerberos realm, an administrative domain with its own Kerberos server (KDC). For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Important • If you are using a Windows 2000 Server the realm must be all upper-case. For details, see the Windows 2000 Server example below.



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the current Kerberos realm. <i>Format:</i> ! U1 getvar "wlan.kerberos.realm"
setvar	This command instructs the printer to change the Kerberos realm. <i>Format:</i> ! U1 setvar "wlan.kerberos.realm" "value" <i>Values:</i> 0-64 alphanumeric characters <i>Default:</i> "kerberos"



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".



Example • 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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the current Kerberos user name. <i>Format:</i> ! U1 getvar "wlan.kerberos.username"
setvar	This command instructs the printer to change the Kerberos user name. <i>Format:</i> ! U1 setvar "wlan.kerberos.username" "value" <i>Values:</i> 0-32 alphanumeric characters <i>Default:</i> "user"



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.mac_addr

Description This command retrieves the MAC address of the wireless print server. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command instructs the printer to respond with the MAC address of the wireless print server. <i>Format: ! U1 getvar "wlan.mac_addr"</i>



Example • In this example, the `getvar` result is the MAC address for the wireless print server.

```
! U1 getvar "wlan.mac_addr"
```

wlan.mac_raw

Description This command specifies the RAW MAC address of the wireless print server. The raw mac address is the mac address without the colons (":"). For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the RAW MAC address of the wireless print server. <i>Format:</i> ! 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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar

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".



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the network-mode value. <i>Format:</i> ! U1 getvar "wlan.operating_mode"
setvar	This command instructs the printer to set the network operating mode. <i>Format:</i> ! U1 setvar "wlan.operating_mode" "value" <i>Values:</i> "adhoc" = printer will try to associate with a network device "infrastructure" = printer will try to associate with an access point <i>Default:</i> "infrastructure"



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

Description This printer setting refers to the generic password that is used by the wireless securities that need a password. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Important • Kerberos has its own password field



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with a generic password for wireless securities. <i>Format:</i> ! U1 getvar "wlan.password" For protection a single "*" prints.
setvar	This command instructs the printer to set a generic password for the wireless securities that need a password. <i>Format:</i> ! U1 setvar "wlan.password" "value" <i>Values:</i> A maximum of 32 alphanumeric characters. <i>Default:</i> "password"



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.preamble

Description This printer setting selects the radio preamble length to be used. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the current preamble length. <i>Format:</i> ! U1 getvar "wlan.preamble"
setvar	This command instructs the printer to set the preamble length. <i>Format:</i> ! U1 setvar "wlan.preamble" "value" <i>Values:</i> "long" = enables long preamble "short" = enables short preamble <i>Default:</i> "long"



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

Description This printer setting allows the setting of the optional private key password. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the value of the private key password. <i>Format:</i> ! U1 getvar "wlan.private_key_password" For protection a single "*" prints.
setvar	This command instructs the printer to set the private key password. <i>Format:</i> ! U1 setvar "wlan.private_key_password" "value" <i>Values:</i> A maximum of 32 alphanumeric characters <i>Default:</i> ""



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.roam.interchannel_delay

Description This command sets how long of a delay before scanning the next channel when roaming. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the current set delay time before scanning the next channel when roaming. <i>Format:</i> ! U1 getvar "wlan.roam.interchannel_delay"
setvar	This command sets how long of a delay before scanning the next channel when roaming. The values are in milliseconds. <i>Format:</i> ! U1 setvar "wlan.roam.interchannel_delay" "value" <i>Values:</i> 10 to 30000 <i>Default:</i> "400"



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

Description This printer setting refers to specifying the wireless roam interval. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the specified roam interval. <i>Format:</i> ! U1 getvar "wlan.roam.interval"
setvar	This command instructs the printer to set the wireless roam interval. <i>Format:</i> ! U1 setvar "wlan.roam.interval" "value" <i>Values:</i> Decimal values between 5 and 255 inclusive <i>Default:</i> "20"



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

Description This command sets how long the radio waits on a channel looking for probe responses. For printer support, see [SGD Command Support on page 997](#).

Type getvar;setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command retrieves the current setting for how long the radio waits on a channel looking for probe responses. <i>Format:</i> ! U1 getvar "wlan.roam.max_chan_scan_time"
setvar	This command sets how long the radio waits on a channel looking for probe responses. The values are in milliseconds. <i>Format:</i> ! U1 setvar "wlan.roam.max_chan_scan_time" "value" <i>Values:</i> 10 to 500 <i>Default:</i> "100"



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.signal

Description This printer setting refers to specifying the wireless roam signal. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the specified wireless roam signal. <i>Format:</i> ! U1 getvar "wlan.roam.signal"
setvar	This command instructs the printer to set the wireless roam signal. <i>Format:</i> ! U1 setvar "wlan.roam.signal" "value" <i>Values:</i> Decimal values between 1 and 75 inclusive. <i>Default:</i> "50"



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.security

Description This printer setting allows you to specify both the wireless encryption type and authentication type in one command. For printer support, see [SGD Command Support on page 997](#).



Note • The supporting parameters that are required vary based on the security type that you select. See [Supporting SGD Commands for Different Security Types on page 907](#) for instructions for each security type.

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).



Note • 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.



Note • Configuring the printer for WPA also allows the printer to be used in WPA2 environments.

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to return the name and not the type. If an invalid security mode is entered the printer returns Invalid Mode. <i>Format: ! U1 getvar "wlan.security" "value"</i>

Commands	Details
setvar	This command instructs the printer to set the wireless security value. <i>Format:</i> ! U1 setvar "wlan.security" "value" <i>Values:</i> "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" <i>Default:</i> "1"



Example • 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 SGD Commands 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](#) on page 922)
- WEP authentication type (see [wlan.wep.auth_type](#) on page 921)
- WEP key type (see [wlan.wep.key_format](#) on page 928)
- the actual values of any WEP encryption keys to be used (see [wlan.wep.key1](#) on page 923, [wlan.wep.key2](#) on page 924, [wlan.wep.key3](#) on page 925, or [wlan.wep.key4](#) on page 926)



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](#) on page 922)
- WEP authentication type (see [wlan.wep.auth_type](#) on page 921)
- WEP key type (see [wlan.wep.key_format](#) on page 928)
- the actual values of any WEP encryption keys to be used (see [wlan.wep.key1](#) on page 923, [wlan.wep.key2](#) on page 924, [wlan.wep.key3](#) on page 925, or [wlan.wep.key4](#) on page 926)



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](#) on page 900)



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 on page 920](#))
- password (see [wlan.password on page 898](#))



Example • This example configures the printer for EAP-TTLS authentication, including a user ID 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 on page 920](#))
- password (see [wlan.password on page 898](#))
- optional private key password (see [wlan.private_key_password on page 900](#))



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 on page 920](#))
- password (see [wlan.password on page 898](#))



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 on page 920](#))
- password (see [wlan.password on page 898](#))



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 on page 927](#))



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](#) on page 900)



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](#) on page 920)
- password (see [wlan.password](#) on page 898)



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 on page 920](#))
- password (see [wlan.password on page 898](#))
- optional private key password (see [wlan.private_key_password on page 900](#))



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 on page 920](#))
- password (see [wlan.password on page 898](#))



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](#) on page 920)
- password (see [wlan.password](#) on page 898)



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](#) on page 894)
- Kerberos password (see [wlan.kerberos.password](#) on page 892)
- realm (see [wlan.kerberos.realm](#) on page 893)
- Key Distribution Center (KDC) (see [wlan.kerberos.kdc](#) on page 891)



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

Description This command returns the signal noise on the wireless network. Values above 40% represent a very significant noise, and radio communication is not reliable. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command instructs the printer to return the current signal noise on the wireless network. <i>Format: ! U1 getvar "wlan.signal_noise"</i>



Example • In this example, the getvar result is the current signal_noise value.

```
! U1 getvar "wlan.signal_noise"
```

wlan.signal_quality

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command instructs the printer to return the current signal quality of the wireless network. <i>Format: ! U1 getvar "wlan.signal_quality"</i>



Example • In this example, the getvar result is the current signal_quality value.

```
! U1 getvar "wlan.signal_quality"
```

wlan.signal_strength

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the command for this format:

Commands	Details
getvar	This command instructs the printer to respond with the current signal strength. <i>Format:</i> ! U1 getvar "wlan.signal_strength"



Example • In this example, the getvar result is "93".

```
! U1 getvar "wlan.signal_strength"
```

wlan.station_name

Description This printer setting refers to the station name. For printer support, see [SGD Command Support on page 997](#).

Type setvar;getvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the station name value. <i>Format:</i> ! U1 getvar "wlan.station_name"
setvar	This command instructs the printer to set the station name. <i>Format:</i> ! U1 setvar "wlan.station_name" "value" <i>Values:</i> A maximum of 32 alphanumeric characters <i>Default:</i> "ZEBRA"



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.tx_power

Description This printer setting refers to specifying the wireless transmit power. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the wireless transmit power. <i>Format:</i> ! U1 getvar "wlan.tx_power"
setvar	This command instructs the printer to set the wireless transmit power. <i>Format:</i> ! U1 setvar "wlan.tx_power" "value" <i>Values:</i> Decimal values of 1, 5, 20, 30, 50, 100 <i>Default:</i> "100"



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

Description This printer setting refers to specifying the wireless transmit rate. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the wireless transmit rate. <i>Format:</i> ! U1 getvar "wlan.tx_rate"
setvar	This command instructs the printer to set the wireless transmit rate. <i>Format:</i> ! U1 setvar "wlan.tx_rate" "value" <i>Values:</i> 1, 2, 5.5, 11, all <i>Default:</i> "all"



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.username

Description This printer setting refers to the generic user name that is used by the wireless securities that need a user name. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Important • Kerberos has its own user name field.



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with a generic user name for the wireless securities that need a user name. <i>Format:</i> ! U1 getvar "wlan.username"
setvar	This command instructs the printer to set a generic user name for wireless securities that need a user name. <i>Format:</i> ! U1 setvar "wlan.username" "value" <i>Values:</i> A maximum of 32 alphanumeric characters <i>Default:</i> "user"



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

Description 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to retrieve the current WEP authentication type. <i>Format:</i> ! U1 getvar "wlan.wep.auth_type"
setvar	This command instructs the printer to set the WEP authentication type. <i>Format:</i> ! U1 setvar "wlan.wep.auth_type" "value" <i>Values:</i> "open" = enables the open authentication type "shared" = enables the shared authentication type <i>Default:</i> "open"



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

Description This printer setting refers to 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). For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the encryption key index. <i>Format:</i> ! U1 getvar "wlan.wep.index"
setvar	This command instructs the printer to set the encryption key index. <i>Format:</i> ! U1 setvar "wlan.wep.index" "value" <i>Values:</i> "1" = enables encryption key 1 "2" = enables encryption key 2 "3" = enables encryption key 3 "4" = enables encryption key 4 <i>Default:</i> "1"



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

Description This printer setting refers to 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the encryption key. <i>Format:</i> ! U1 getvar "wlan.wep.key1" For protection a single "*" prints.
setvar	This command instructs the printer to set the encryption key. <i>Format:</i> ! U1 setvar "wlan.wep.key1" "value" <i>Values:</i> 10 hexadecimal characters for 40-bit encryption 26 hexadecimal characters for 128-bit encryption <i>Default:</i> All zeros



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

Description This printer setting refers to 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer respond with the encryption key. <i>Format:</i> ! U1 getvar "wlan.wep.key2" For protection a single "*" prints.
setvar	This command instructs the printer to set the encryption key. <i>Format:</i> ! U1 setvar "wlan.wep.key2" "value" <i>Values:</i> 10 hexadecimal characters for 40-bit encryption 26 hexadecimal characters for 128-bit encryption <i>Default:</i> All zeros



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

Description This printer setting refers to 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the encryption key. <i>Format:</i> ! U1 getvar "wlan.wep.key3" For protection a single "*" prints.
setvar	This command instructs the printer to set the encryption key. <i>Format:</i> ! U1 setvar "wlan.wep.key3" "value" <i>Values:</i> 10 hexadecimal characters for 40-bit encryption 26 hexadecimal characters for 128-bit encryption <i>Default:</i> All zeros



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

Description This printer setting refers to 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. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer respond with the encryption key. <i>Format:</i> ! U1 getvar "wlan.wep.key4" For protection a single "*" prints.
setvar	This command instructs the printer to set the encryption key. <i>Format:</i> ! U1 setvar "wlan.wep.key4" "value" <i>Values:</i> 10 hexadecimal characters for 40-bit encryption 26 hexadecimal characters for 128-bit encryption <i>Default:</i> All zeros



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

Description This printer setting specifies the pre-shared key (PSK) value to use when the WPA authentication is set to PSK. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the pre-shared key. <i>Format:</i> ! U1 getvar "wlan.wpa.psk" For protection a single "*" prints.
setvar	This command instructs the printer to set the pre-shared key. <i>Format:</i> ! U1 setvar "wlan.wpa.psk" "value" <i>Values:</i> 64 hexadecimal digits <i>Default:</i> 64 zeros (00000000...)



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

Description This printer setting specifies the format for the WEP key. For printer support, see [SGD Command Support on page 997](#).

Type getvar; setvar



Important • This printer setting should proceed any of the `wep.key` settings if you select a non-default value.



Note • For details on SGD command structure, see [Command Structure on page 772](#).

This table identifies the commands for this format:

Commands	Details
getvar	This command instructs the printer to respond with the WEP key format. <i>Format:</i> <code>! U1 getvar "wep.key_format"</code>
setvar	This command instructs the printer to set the WEP key format. <i>Format:</i> <code>! U1 setvar "wlan.wep.key_format" "value"</code> <i>Values:</i> <code>"ascii"</code> = WEP key is set by ASCII string <code>"hex"</code> = WEP key is a Hex string <i>Default:</i> <code>"hex"</code>



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"`.



Zebra Code Pages

This section provides you with a visual of the different Zebra 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
	20	32	0	30	48	@	40	64	P	50	80	'	60	96	p	70	112
!	21	33	1	31	49	A	41	65	Q	51	81	a	61	97	q	71	113
"	22	34	2	32	50	B	42	66	R	52	82	b	62	98	r	72	114
#	23	35	3	33	51	C	43	67	S	53	83	c	63	99	s	73	115
\$	24	36	4	34	52	D	44	68	T	54	84	d	64	100	t	74	116
%	25	37	5	35	53	E	45	69	U	55	85	e	65	101	u	75	117
&	26	38	6	36	54	F	46	70	V	56	86	f	66	102	v	76	118
'	27	39	7	37	55	G	47	71	W	57	87	g	67	103	w	77	119
(	28	40	8	38	56	H	48	72	X	58	88	h	68	104	x	78	120
)	29	41	9	39	57	I	49	73	Y	59	89	i	69	105	y	79	121
*	2a	42	:	3a	58	J	4a	74	Z	5a	90	j	6a	106	z	7a	122
+	2b	43	;	3b	59	K	4b	75	[	5b	91	k	6b	107	{	7b	123
,	2c	44	<	3c	60	L	4c	76	¢	5c	92	l	6c	108		7c	124
-	2d	45	=	3d	61	M	4d	77	]	5d	93	m	6d	109	}	7d	125
.	2e	46	>	3e	62	N	4e	78	^	5e	94	n	6e	110	~	7e	126
/	2f	47	?	3f	63	O	4f	79	_	5f	95	o	6f	111	◻	7f	127

CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC
É	90	144	á	a0	160	⋮	b0	176	┐	c0	192	ð	d0	208	Ó	e0	224
æ	91	145	í	a1	161	⋈	b1	177	└	c1	193	Ð	d1	209	ß	e1	225
Æ	92	146	ó	a2	162	⋈	b2	178	┘	c2	194	È	d2	210	Ô	e2	226
ô	93	147	ú	a3	163		b3	179	┘	c3	195	Ê	d3	211	Õ	e3	227
ö	94	148	ñ	a4	164	└	b4	180	—	c4	196	Ë	d4	212	Ö	e4	228
ò	95	149	Ñ	a5	165	Á	b5	181	+	c5	197	Ì	d5	213	Œ	e5	229
û	96	150	ª	a6	166	Â	b6	182	ã	c6	198	Í	d6	214	μ	e6	230
ù	97	151	º	a7	167	Ã	b7	183	Ä	c7	199	Î	d7	215	þ	e7	231
ÿ	98	152	¿	a8	168	©	b8	184	⌌	c8	200	Ï	d8	216	þ	e8	232
Ö	99	153	®	a9	169	≡	b9	185	⌌	c9	201	┘	d9	217	Ú	e9	233
Ü	9a	154	¬	aa	170		ba	186	⌌	ca	202	┘	da	218	Û	ea	234
ø	9b	155	½	ab	171	⌌	bb	187	⌌	cb	203	■	db	219	Ù	eb	235
£	9c	156	¼	ac	172	⌌	bc	188	⌌	cc	204	■	dc	220	Ý	ec	236
Ø	9d	157	¡	ad	173	¢	bd	189	=	cd	205		dd	221	Ý	ed	237
×	9e	158	«	ae	174	¥	be	190	≡	ce	206	Ì	de	222	˘	ee	238
f	9f	159	»	af	175	┘	bf	191	⌌	cf	207	■	df	223	˙	ef	239
																ff	255

Zebra Code Page 1250 (Scalable/Downloaded TTF) — 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
	20	32
!	21	33
"	22	34
#	23	35
\$	24	36
%	25	37
&	26	38
'	27	39
(	28	40
)	29	41
*	2a	42
+	2b	43
,	2c	44
-	2d	45
.	2e	46
/	2f	47

CHR	HEX	DEC
0	30	48
1	31	49
2	32	50
3	33	51
4	34	52
5	35	53
6	36	54
7	37	55
8	38	56
9	39	57
:	3a	58
;	3b	59
<	3c	60
=	3d	61
>	3e	62
?	3f	63

CHR	HEX	DEC
@	40	64
A	41	65
B	42	66
C	43	67
D	44	68
E	45	69
F	46	70
G	47	71
H	48	72
I	49	73
J	4a	74
K	4b	75
L	4c	76
M	4d	77
N	4e	78
O	4f	79

CHR	HEX	DEC
P	50	80
Q	51	81
R	52	82
S	53	83
T	54	84
U	55	85
V	56	86
W	57	87
X	58	88
Y	59	89
Z	5a	90
[	5b	91
\	5c	92
]	5d	93
^	5e	94
_	5f	95

CHR	HEX	DEC
`	60	96
a	61	97
b	62	98
c	63	99
d	64	100
e	65	101
f	66	102
g	67	103
h	68	104
i	69	105
j	6a	106
k	6b	107
l	6c	108
m	6d	109
n	6e	110
o	6f	111

CHR	HEX	DEC
p	70	112
q	71	113
r	72	114
s	73	115
t	74	116
u	75	117
v	76	118
w	77	119
x	78	120
y	79	121
z	7a	122
{	7b	123
	7c	124
}	7d	125
~	7e	126
	7f	127

CHR	HEX	DEC
€	80	128
	81	129
,	82	130
	83	131
,,	84	132
...	85	133
†	86	134
‡	87	135
	88	136
‰	89	137
Š	8a	138
‹	8b	139
Ś	8c	140
Ť	8d	141
Ž	8e	142
Ż	8f	143

CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC
	90	144		a0	160	°	b0	176	Ř	c0	192	Đ	d0	208	í	e0	224
‘	91	145	˘	a1	161	±	b1	177	Á	c1	193	Ň	d1	209	á	e1	225
’	92	146	˘	a2	162	˘	b2	178	Ā	c2	194	Ň	d2	210	â	e2	226
“	93	147	Ł	a3	163	†	b3	179	Ǻ	c3	195	Ó	d3	211	ǻ	e3	227
”	94	148	Ǫ	a4	164	’	b4	180	Ǻ	c4	196	Ō	d4	212	ǻ	e4	228
•	95	149	Ą	a5	165	μ	b5	181	Ĺ	c5	197	Ŏ	d5	213	í	e5	229
—	96	150	ı	a6	166	¶	b6	182	Ć	c6	198	Ŏ	d6	214	ć	e6	230
—	97	151	§	a7	167	•	b7	183	Ç	c7	199	×	d7	215	ç	e7	231
	98	152	”	a8	168	˘	b8	184	Č	c8	200	Ř	d8	216	č	e8	232
™	99	153	©	a9	169	ą	b9	185	É	c9	201	Ů	d9	217	é	e9	233
š	9a	154	§	aa	170	ş	ba	186	Ě	ca	202	Ú	da	218	ẹ	ea	234
›	9b	155	«	ab	171	»	bb	187	Ě	cb	203	Ů	db	219	ě	eb	235
ś	9c	156	¬	ac	172	Ł	bc	188	Ě	cc	204	Ů	dc	220	ě	ec	236
ť	9d	157	-	ad	173	”	bd	189	Í	cd	205	Ý	dd	221	í	ed	237
ž	9e	158	®	ae	174	ı	be	190	İ	ce	206	Ț	de	222	î	ee	238
ž	9f	159	Ž	af	175	ž	bf	191	Đ	cf	207	ß	df	223	d'	ef	239

Zebra Code Page 1250 — (Bitmap Fonts) Central and Eastern European Latin Character Set

This is the Zebra Code Page 1250 that supports bitmap fonts:



Note • Font D was used to display this chart.

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
!	21	33	1	31	49	A	41	65	Q	51	81	a	61	97	q	71	113
"	22	34	2	32	50	B	42	66	R	52	82	b	62	98	r	72	114
#	23	35	3	33	51	C	43	67	S	53	83	c	63	99	s	73	115
\$	24	36	4	34	52	D	44	68	T	54	84	d	64	100	t	74	116
%	25	37	5	35	53	E	45	69	U	55	85	e	65	101	u	75	117
&	26	38	6	36	54	F	46	70	V	56	86	f	66	102	v	76	118
'	27	39	7	37	55	G	47	71	W	57	87	g	67	103	w	77	119
(<	28	40	8	38	56	H	48	72	X	58	88	h	68	104	x	78	120
)	29	41	9	39	57	I	49	73	Y	59	89	i	69	105	y	79	121
*	2a	42	=	3a	58	J	4a	74	Z	5a	90	j	6a	106	z	7a	122
+	2b	43	;	3b	59	K	4b	75	[	5b	91	k	6b	107	(	7b	123
,	2c	44	<	3c	60	L	4c	76	\	5c	92	l	6c	108	!	7c	124
-	2d	45	=	3d	61	M	4d	77	]	5d	93	m	6d	109	)	7d	125
.	2e	46	>	3e	62	N	4e	78	^	5e	94	n	6e	110	~	7e	126
/	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
	90	144		a0	160	°	b0	176		c0	192		d0	208		e0	224
“	91	145		a1	161	±	b1	177	Á	c1	193		d1	209	á	e1	225
”	92	146		a2	162		b2	178	Â	c2	194		d2	210	â	e2	226
”	93	147		a3	163		b3	179		c3	195	Ó	d3	211		e3	227
”	94	148	Ø	a4	164	´	b4	180	Ä	c4	196	Ö	d4	212	ä	e4	228
-	95	149		a5	165	µ	b5	181		c5	197		d5	213		e5	229
—	96	150	!	a6	166	¶	b6	182		c6	198	Ö	d6	214		e6	230
—	97	151	§	a7	167	-	b7	183	Ç	c7	199	×	d7	215	ç	e7	231
	98	152	¨	a8	168	¸	b8	184		c8	200		d8	216		e8	232
™	99	153	©	a9	169		b9	185	É	c9	201		d9	217	é	e9	233
Š	9a	154		aa	170		ba	186		ca	202	Ú	da	218		ea	234
>	9b	155	«	ab	171	»	bb	187	Ë	cb	203		db	219	ë	eb	235
	9c	156	¬	ac	172		bc	188		cc	204	Ü	dc	220		ec	236
	9d	157	-	ad	173		bd	189	İ	cd	205	Ý	dd	221	í	ed	237
Ž	9e	158	®	ae	174		be	190	İ	ce	206		de	222	î	ee	238
	9f	159		af	175		bf	191		cf	207	ß	df	223		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
	20	32	0	30	48	@	40	64	P	50	80	`	60	96	p	70	112
!	21	33	1	31	49	A	41	65	Q	51	81	a	61	97	q	71	113
"	22	34	2	32	50	B	42	66	R	52	82	b	62	98	r	72	114
#	23	35	3	33	51	C	43	67	S	53	83	c	63	99	s	73	115
\$	24	36	4	34	52	D	44	68	T	54	84	d	64	100	t	74	116
%	25	37	5	35	53	E	45	69	U	55	85	e	65	101	u	75	117
&	26	38	6	36	54	F	46	70	V	56	86	f	66	102	v	76	118
'	27	39	7	37	55	G	47	71	W	57	87	g	67	103	w	77	119
(	28	40	8	38	56	H	48	72	X	58	88	h	68	104	x	78	120
)	29	41	9	39	57	I	49	73	Y	59	89	i	69	105	y	79	121
*	2a	42	:	3a	58	J	4a	74	Z	5a	90	j	6a	106	z	7a	122
+	2b	43	;	3b	59	K	4b	75	[	5b	91	k	6b	107	{	7b	123
,	2c	44	<	3c	60	L	4c	76	\	5c	92	l	6c	108		7c	124
-	2d	45	=	3d	61	M	4d	77	]	5d	93	m	6d	109	}	7d	125
.	2e	46	>	3e	62	N	4e	78	^	5e	94	n	6e	110	~	7e	126
/	2f	47	?	3f	63	O	4f	79	_	5f	95	o	6f	111		7f	127

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	Đ	d0	208	à	e0	224
‘	91	145	ì	a1	161	±	b1	177	Á	c1	193	Ñ	d1	209	á	e1	225
’	92	146	ç	a2	162	²	b2	178	Â	c2	194	Ò	d2	210	â	e2	226
“	93	147	£	a3	163	³	b3	179	Ã	c3	195	Ó	d3	211	ã	e3	227
”	94	148	¤	a4	164	´	b4	180	Ä	c4	196	Ô	d4	212	ä	e4	228
•	95	149	¥	a5	165	µ	b5	181	Å	c5	197	Õ	d5	213	å	e5	229
—	96	150		a6	166	¶	b6	182	Æ	c6	198	Ö	d6	214	æ	e6	230
—	97	151	§	a7	167	·	b7	183	Ç	c7	199	×	d7	215	ç	e7	231
~	98	152	”	a8	168	¸	b8	184	È	c8	200	Ø	d8	216	è	e8	232
™	99	153	©	a9	169	¹	b9	185	É	c9	201	Ù	d9	217	é	e9	233
š	9a	154	ª	aa	170	º	ba	186	Ê	ca	202	Ú	da	218	ê	ea	234
›	9b	155	«	ab	171	»	bb	187	Ë	cb	203	Û	db	219	ë	eb	235
œ	9c	156	¬	ac	172	¼	bc	188	Ì	cc	204	Ü	dc	220	ì	ec	236
	9d	157	-	ad	173	½	bd	189	Í	cd	205	Ý	dd	221	í	ed	237
ž	9e	158	®	ae	174	¾	be	190	Î	ce	206	Þ	de	222	î	ee	238
ÿ	9f	159	¯	af	175	¿	bf	191	Ï	cf	207	ß	df	223	ï	ef	239

Zebra Code Page 1253 — Modern Greek Character Set

This is the Zebra Code Page 1253:

CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC			
	0	0		1	1		20	32	0	30	48	@	40	64	P	50	80	'	60	96	p	70	112
	2	2		3	3	!	21	33	1	31	49	A	41	65	Q	51	81	a	61	97	q	71	113
	4	4		5	5	"	22	34	2	32	50	B	42	66	R	52	82	b	62	98	r	72	114
	6	6		7	7	#	23	35	3	33	51	C	43	67	S	53	83	c	63	99	s	73	115
	8	8		9	9	\$	24	36	4	34	52	D	44	68	T	54	84	d	64	100	t	74	116
	a	10		b	11	%	25	37	5	35	53	E	45	69	U	55	85	e	65	101	u	75	117
	c	12		d	13	&	26	38	6	36	54	F	46	70	V	56	86	f	66	102	v	76	118
	e	14		f	15	'	27	39	7	37	55	G	47	71	W	57	87	g	67	103	w	77	119
	10	16		11	17	(	28	40	8	38	56	H	48	72	X	58	88	h	68	104	x	78	120
	12	18		13	19	)	29	41	9	39	57	I	49	73	Y	59	89	i	69	105	y	79	121
	14	20		15	21	*	2a	42	:	3a	58	J	4a	74	Z	5a	90	j	6a	106	z	7a	122
	16	22		17	23	+	2b	43	;	3b	59	K	4b	75	[	5b	91	k	6b	107	{	7b	123
	18	24		19	25	,	2c	44	<	3c	60	L	4c	76	\	5c	92	l	6c	108		7c	124
	1a	26		1b	27	-	2d	45	=	3d	61	M	4d	77	]	5d	93	m	6d	109	}	7d	125
	1c	28		1d	29	.	2e	46	>	3e	62	N	4e	78	^	5e	94	n	6e	110	~	7e	126
	1e	30		1f	31	/	2f	47	?	3f	63	O	4f	79	_	5f	95	o	6f	111		7f	127

CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC			
€	80	128		90	144	€	a0	160	°	b0	176	Τ	c0	192	Π	d0	208	ϒ	e0	224	π	f0	240
	81	129	'	91	145	™	a1	161	±	b1	177	Α	c1	193	Ρ	d1	209	α	e1	225	ρ	f1	241
,	82	130	'	92	146	Α	a2	162	²	b2	178	Β	c2	194		d2	210	β	e2	226	ς	f2	242
f	83	131	"	93	147	£	a3	163	³	b3	179	Γ	c3	195	Σ	d3	211	γ	e3	227	ο	f3	243
„	84	132	"	94	148	α	a4	164		b4	180	Δ	c4	196	Τ	d4	212	δ	e4	228	τ	f4	244
...	85	133	'	95	149	¥	a5	165	μ	b5	181	Ε	c5	197	Υ	d5	213	ε	e5	229	υ	f5	245
†	86	134	—	96	150		a6	166	¶	b6	182	Ζ	c6	198	Φ	d6	214	ζ	e6	230	φ	f6	246
‡	87	135	—	97	151	\$	a7	167	'	b7	183	Η	c7	199	Χ	d7	215	η	e7	231	χ	f7	247
	88	136		98	152	"	a8	168	Ε	b8	184	Θ	c8	200	Ψ	d8	216	θ	e8	232	ψ	f8	248
‰	89	137	™	99	153	©	a9	169	Η	b9	185	Ι	c9	201	Ω	d9	217	ι	e9	233	ω	f9	249
	8a	138		9a	154		aa	170	Ι	ba	186	Κ	ca	202	Ϊ	da	218	κ	ea	234	ϊ	fa	250
'	8b	139	'	9b	155	"	ab	171	»	bb	187	Λ	cb	203	Υ	db	219	λ	eb	235	ϒ	fb	251
	8c	140		9c	156	┐	ac	172	Ο	bc	188	Μ	cc	204	ά	dc	220	μ	ec	236	ό	fc	252
	8d	141		9d	157	'	ad	173	½	bd	189	Ν	cd	205	έ	dd	221	ν	ed	237	ύ	fd	253
	8e	142		9e	158	®	ae	174	Υ	be	190	Ξ	ce	206	ή	de	222	ξ	ee	238	ώ	fe	254
	8f	143		9f	159	—	af	175	Ω	bf	191	Ο	cf	207	Ι	df	223	ο	ef	239		ff	255

Zebra Code Page 1254 — Turkish Character Set

This is the Zebra Code Page 1254:

CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC
	0	0		1	1		20	32	0	30	48	@	40	64	P	50	80	'	60	96	p	70	112
	2	2		3	3	!	21	33	1	31	49	A	41	65	Q	51	81	a	61	97	q	71	113
	4	4		5	5	"	22	34	2	32	50	B	42	66	R	52	82	b	62	98	r	72	114
	6	6		7	7	#	23	35	3	33	51	C	43	67	S	53	83	c	63	99	s	73	115
	8	8		9	9	\$	24	36	4	34	52	D	44	68	T	54	84	d	64	100	t	74	116
	a	10		b	11	%	25	37	5	35	53	E	45	69	U	55	85	e	65	101	u	75	117
	c	12		d	13	&	26	38	6	36	54	F	46	70	V	56	86	f	66	102	v	76	118
	e	14		f	15	'	27	39	7	37	55	G	47	71	W	57	87	g	67	103	w	77	119
	10	16		11	17	(	28	40	8	38	56	H	48	72	X	58	88	h	68	104	x	78	120
	12	18		13	19	)	29	41	9	39	57	I	49	73	Y	59	89	i	69	105	y	79	121
	14	20		15	21	*	2a	42	:	3a	58	J	4a	74	Z	5a	90	j	6a	106	z	7a	122
	16	22		17	23	+	2b	43	;	3b	59	K	4b	75	[	5b	91	k	6b	107	{	7b	123
	18	24		19	25	,	2c	44	<	3c	60	L	4c	76	\	5c	92	l	6c	108		7c	124
	1a	26		1b	27	-	2d	45	=	3d	61	M	4d	77	]	5d	93	m	6d	109	}	7d	125
	1c	28		1d	29	.	2e	46	>	3e	62	N	4e	78	^	5e	94	n	6e	110	~	7e	126
	1e	30		1f	31	/	2f	47	?	3f	63	O	4f	79	_	5f	95	o	6f	111		7f	127

CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC			
€	80	128		90	144		a0	160	°	b0	176	À	c0	192	Ğ	d0	208	à	e0	224	ğ	f0	240
	81	129	'	91	145	ı	a1	161	±	b1	177	Á	c1	193	Ñ	d1	209	á	e1	225	ñ	f1	241
,	82	130	'	92	146	Ç	a2	162	²	b2	178	Â	c2	194	Ò	d2	210	â	e2	226	ò	f2	242
/	83	131	"	93	147	£	a3	163	³	b3	179	Ã	c3	195	Ó	d3	211	ã	e3	227	ó	f3	243
„	84	132	"	94	148	¤	a4	164		b4	180	Ä	c4	196	Ô	d4	212	ä	e4	228	ô	f4	244
...	85	133	'	95	149	¥	a5	165	µ	b5	181	Å	c5	197	Õ	d5	213	å	e5	229	õ	f5	245
†	86	134	—	96	150		a6	166	¶	b6	182	Æ	c6	198	Ö	d6	214	æ	e6	230	ö	f6	246
‡	87	135	—	97	151	\$	a7	167	'	b7	183	Ç	c7	199	×	d7	215	ç	e7	231	÷	f7	247
^	88	136	~	98	152	"	a8	168	,	b8	184	È	c8	200	Ø	d8	216	è	e8	232	ø	f8	248
%	89	137	™	99	153	©	a9	169	¹	b9	185	É	c9	201	Ù	d9	217	é	e9	233	ù	f9	249
§	8a	138	§	9a	154	ª	aa	170	º	ba	186	Ê	ca	202	Ú	da	218	ê	ea	234	ú	fa	250
'	8b	139	'	9b	155	«	ab	171	»	bb	187	Ë	cb	203	Û	db	219	ë	eb	235	û	fb	251
œ	8c	140	œ	9c	156	¬	ac	172	¼	bc	188	Ì	cc	204	Ü	dc	220	ì	ec	236	ü	fc	252
	8d	141		9d	157	•	ad	173	½	bd	189	Í	cd	205	İ	dd	221	í	ed	237	ı	fd	253
	8e	142		9e	158	®	ae	174	¾	be	190	Î	ce	206	Ş	de	222	î	ee	238	ş	fe	254
	8f	143	Ÿ	9f	159		af	175	¿	bf	191	Ï	cf	207	ß	df	223	ï	ef	239	ÿ	ff	255

Zebra Code Page 1255 — Hebrew Character Set

This is the Zebra Code Page 1255:

CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC
	0	0		1	1		20	32	0	30	48	@	40	64	P	50	80
	2	2		3	3	!	21	33	1	31	49	A	41	65	Q	51	81
	4	4		5	5	"	22	34	2	32	50	B	42	66	R	52	82
	6	6		7	7	#	23	35	3	33	51	C	43	67	S	53	83
	8	8		9	9	\$	24	36	4	34	52	D	44	68	T	54	84
	a	10		b	11	%	25	37	5	35	53	E	45	69	U	55	85
	c	12		d	13	&	26	38	6	36	54	F	46	70	V	56	86
	e	14		f	15	'	27	39	7	37	55	G	47	71	W	57	87
	10	16		11	17	(	28	40	8	38	56	H	48	72	X	58	88
	12	18		13	19	)	29	41	9	39	57	I	49	73	Y	59	89
	14	20		15	21	*	2a	42	:	3a	58	J	4a	74	Z	5a	90
	16	22		17	23	+	2b	43	;	3b	59	K	4b	75	[	5b	91
	18	24		19	25	,	2c	44	<	3c	60	L	4c	76	\	5c	92
	1a	26		1b	27	-	2d	45	=	3d	61	M	4d	77	]	5d	93
	1c	28		1d	29	.	2e	46	>	3e	62	N	4e	78	^	5e	94
	1e	30		1f	31	/	2f	47	?	3f	63	O	4f	79	_	5f	95

CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC			
€	80	128		90	144		a0	160	°	b0	176	:	c0	192		d0	208	⌘	e0	224	ן	f0	240
	81	129	'	91	145	ı	a1	161	±	b1	177	ˆ	c1	193		d1	209	ג	e1	225	ו	f1	241
,	82	130	'	92	146	©	a2	162	²	b2	178	˙	c2	194		d2	210	ל	e2	226	י	f2	242
f	83	131	"	93	147	£	a3	163	³	b3	179	˚	c3	195	:	d3	211	ך	e3	227	ף	f3	243
„	84	132	"	94	148	®	a4	164		b4	180	.	c4	196	ן	d4	212	ה	e4	228	צ	f4	244
...	85	133	•	95	149	¥	a5	165	μ	b5	181	˘	c5	197	י	d5	213	י	e5	229	ץ	f5	245
†	86	134	—	96	150		a6	166	¶	b6	182	˙	c6	198	"	d6	214	י	e6	230	ץ	f6	246
‡	87	135	—	97	151	\$	a7	167	•	b7	183	.	c7	199	'	d7	215	ן	e7	231	פ	f7	247
^	88	136	~	98	152	"	a8	168	.	b8	184	.	c8	200	"	d8	216	ו	e8	232	ך	f8	248
0/0	89	137	™	99	153	©	a9	169	¹	b9	185		c9	201		d9	217	'	e9	233	ש	f9	249
	8a	138		9a	154	×	aa	170	÷	ba	186		ca	202		da	218	ך	ea	234	ן	fa	250
'	8b	139	'	9b	155	“	ab	171	”	bb	187	...	cb	203		db	219	ן	eb	235		fb	251
	8c	140		9c	156	⌒	ac	172	¼	bc	188	.	cc	204		dc	220	ל	ec	236		fc	252
	8d	141		9d	157	•	ad	173	½	bd	189	.	cd	205		dd	221	ו	ed	237		fd	253
	8e	142		9e	158	®	ae	174	¾	be	190	.	ce	206		de	222	ן	ee	238		fe	254
	8f	143		9f	159		af	175	ı	bf	191	.	cf	207		df	223	י	ef	239		ff	255



Notes • _____

B

ASCII



This section shows the American Standard Code for Information Interchange (ASCII) code used by Zebra printers.

ASCII Code Chart

Shaded areas in [Table 26](#) indicate characters not recommended for command prefix, format prefix, or delimiter characters.

Table 26 • Zebra Used ASCII Code

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
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 Bar Codes

This section provides information about different fonts (type faces) and bar codes that can be used with the printer.

Contents

Standard Printer Fonts	948
Proportional and Fixed Spacing	950
Scalable Versus Bitmapped Fonts	952
Font Matrices	954
Bar Codes	956

Standard Printer Fonts

Most Zebra printers come standard with 15 bitmapped fonts and one scalable font, see [Figure 6](#). 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 6 • Examples of the Standard Printer Fonts

FONT A — — ABCDwxyz 12345
FONT B — — ABCDXYZ 12345 UPPER CASE ONLY
FONT D — — ABCDwxyz 12345
FONT E — — (OCR-B) ABCDwxyz 12345
FONT F — — ABCDwxyz 12345
FONT G — — AByz 12
FONT H — — (OCR-A) UPPER CASE ONLY
zero — **FONT O** — — (Scaleable) ABCDwxyz 12345
FONT GS — — ® © ™ (U) Accessed with the ^GS command.
FONT P — — ABCDwxyz 12345
FONT Q — — ABCDwxyz 12345
FONT R — — ABCDwxyz 12345
FONT S — — ABCDwxyz 12345
FONT T — — ABCDwxyz 12345
FONT U — — ABCDwxyz 12345
FONT V — — ABCDwxyz 12345

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. [Figure 7](#) shows how these character values are printed.



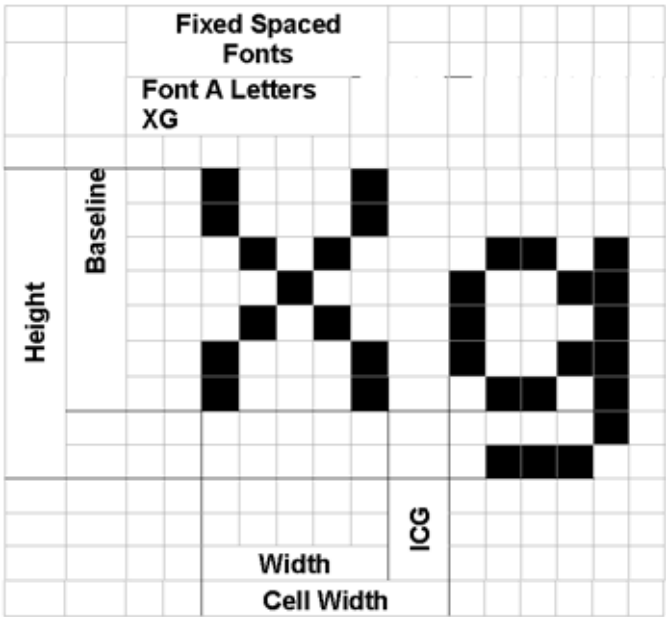
Note • Unidentified characters should default to a space.

Proportional and Fixed Spacing

Proportional spacing is different than fixed spacing. In [Table 27](#), 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 8](#) is an example of fixed space fonts:

Figure 8 • Fixed Space Fonts Proportion



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 27 • 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.

Scalable Fonts

All dot parameters used in the commands to create scalable fonts are translated into a point size because scalable fonts are measured in point sizes, not dots. To determine how many dots to enter to obtain a particular point size, use the following formula. The actual point size will be an approximate value.

$$\text{Dots} = \frac{(\text{Point size}) \times (\text{Dots per inch of Printer})}{72}$$

- For printers using a 6 dot/mm printhead the “dots per inch of printer” value is 152.4
- For printers using a 8 dot/mm printhead the “dots per inch of printer” value is 203.2
- For printers using a 12 dot/mm printhead the “dots per inch of printer” value is 304.8
- For printers using a 24 dot/mm printhead the “dots per inch of printer” value is 609.6

The actual height and width of the character in dots will vary, depending on the font style and the particular character. Therefore, some characters will be smaller and some will be larger than the actual dot size requested. The baselines for all scalable fonts are calculated against the dot size of the cell. The baseline is 3/4 down from the top of the cell. For example, if the size of the cell is 80 dots, the baseline will be 60 dots (3/4) down from the top of the cell.

For more information concerning fonts and related commands, see [~DB on page 163](#) and [~DS on page 172](#).

Bitmapped Fonts

Internal bitmapped fonts can be magnified from 1 to 10 times their normal (default) size. The magnification factor is in whole numbers. Therefore, if the normal size of a bitmapped font is 9 dots high and 5 dots wide, a magnification factor of 3 would produce a character of 27 dots high and 15 dots wide. Height and width can be magnified independently.

Magnification Factor

The font commands contain parameters for entering the height and width of printed characters. The values are always entered in dots. When entering these values for bitmapped fonts, use the following formula:

$$\text{Base Height} \times \text{Magnification Factor} = \text{Height Parameter Value}$$

The same principle applies when calculating width.

Example:

Base height = 9 dots

Base width = 5 dots

To magnify a bitmapped character with the above specifics 3 times its size:

Height parameter = 27 [9 x 3]

Width parameter = 15 [5 x 3]

Changing Bitmapped Font Size

Alphanumeric field command (^A) parameters h and w control the magnification and, therefore, the ultimate size of the font. The parameter is specified in dots, but ZPL II actually uses an integer multiplier times the original height/width of the font. For example, if you specify

^{AD}, 54

you get characters three times their normal size (54 dots high), but if you specify

^{AD, 52}

you receive the same result, not characters 52 dots high.

Defining only the height or width of a bitmapped font forces the magnification to be proportional to the parameter defined. If neither is defined, the ^CF height and width are used. For example, if the height is twice the standard height, the width will be twice the standard width.



Example • If a ^CF command, with height and width parameters defined, is used to set the first font, any ^A commands (to select a different font) that follow must have the height and width parameter filled in.

If this is not done, the newly selected font will be magnified using values for the ^CF height and width parameters. This is an example of what happens:.

ZPL II CODE	GENERATED LABEL
^XA^LL1800 ^FO50,50^CFD,26,10^FDZEBRA....^FS ^FO50,100^FD"Bar Code, Bar None"^FS ^FO50,200^AA^FDZEBRA....^FS ^FO50,250^FD"Bar Code, Bar None"^FS ^XZ	ZEBRA.... "Bar Code, Bar None" ZEBRA.... "Bar Code, Bar None"

Font Matrices

Type Key U = Uppercase, L = Lowercase, D = Descenders

Table 28 • 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 29 • 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
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 30 • 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 31 • 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	

Bar Codes

Every bar code 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 bar code section in the [ZPL Commands on page 39](#) provides a table of applicable characters. Start and stop characters and check digits are used by many, but not all, bar codes. These will be indicated in the specific bar code explanations.

Zebra printers can print the following kinds of bar codes:

Bar code 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 bar codes - 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 bar codes - Aztec - Code 49 - Maxi Code - TLC39 - PDF-417 - QR Code - Codablock - DataMatrix - Micro-PDF417	
Bar code 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 9 • Quiet Zone in a Bar Code



Bar Code Field Instructions

To create a bar code, a bar code field command must be contained in the label format. [Table 32](#) shows the bar code field commands. The number in brackets denotes the print ratio. Each command produces a unique bar code.



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 bar code field commands ratio, see [Table 11 on page 143](#).

Table 32 • Bar Code Field Commands

ZPL Command	Command Description	Ratio
^B0	Aztec Bar Code 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 Bar Code	[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 32 • Bar Code Field Commands

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]
^BO	Aztec Bar Code 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 bar codes	[Fixed]

Additionally, each bar code field command can be issued with a definition parameter string. The parameter string defines field rotation, height, and interpretation line status for all bar codes. For some bar codes, 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 bar codes 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 bar code parameters suit the application, then only the bar code command needs to be entered.

Parameters in bar code 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. Table 33 through Table 36 identify the groups and the bar codes they contain:

Table 33 • Numeric Only Bar Codes

ZPL Command	Command Description
^B0	Aztec Bar Code Parameters
^B1	Code 11
^B5	Planet Code Bar Code
^BI	Industrial 2 of 5
^BJ	Standard 2 of 5
^BK	ANSI Codabar (or NW-7)
^BM	MSI
^B0	Aztec Bar Code Parameters
^BP	Plessey
^BZ	PostNet (*), USPS Intelligent Mail, and Planet bar codes

Table 34 • Retail Labeling Bar Codes

ZPL Command	Command Description
^B0	Aztec Bar Code Parameters
^B8	EAN-8
^B9	UPC-E
^BE	EAN-13
^B0	Aztec Bar Code Parameters
^BS	UPC/EAN extensions
^BU	UPC-A

Table 35 • Alphanumeric Bar Codes

ZPL Command	Command Description
^B0	Aztec Bar Code Parameters
^B3	Code 39
^BA	Code 93
^BC	Code 128
^BL	LOGMARS
^B0	Aztec Bar Code Parameters

Table 36 • Two-Dimensional Bar Codes

ZPL Command	Command Description
^B0	Aztec Bar Code Parameters
^B4	Code 49
^B7	PDF417
^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.

Contents

Mod 10 Check Digit	962
Mod 43 Check Digit	963

Mod 10 Check Digit

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.
 $20 \times 3 = 60$
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 1 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

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	M=22	X=33
1=1	C=12	N=23	Y=34
2=2	D=13	O=24	Z=35
3=3	E=14	P=25	- =36
4=4	F=15	Q=26	. = 37
5=5	G=16	R=27	Space=38
6=6	H=17	S=28	\$=39
7=7	I=18	T=29	/=40
8=8	J=19	U=30	+ =41
9=9	K=20	V=31	% =42
A=10	L=21	W=32	

→ 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.

$$115/43 = 2 \text{ Remainder is } 29$$

3. The “check digit” is the character that corresponds to the value of the remainder.

$$\text{Remainder} = 29$$

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



Notes •



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, please contact your Reseller or Zebra Representative.

Contents

Introduction to B64 and Z64.	966
B64 and Z64 Encoding.	968

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 unenclosed 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.



Example • The following is an example of an existing download command using the new encoding:

```
~DTARIAL,59494,:Z64:H4sICMB8+DMAC0FSSUFMLlRURgDsmnd8VEW7x5+ZO
edsyibZNNJhlyWhbEJIwYSwJDGNkmwghJIgJYEEhQIPSGgKAjEAiIiVaSoIJ
YNBAkIGgGxUBVUUCGU0JQSC0WFnPvBE+SF18+9H+8f973X+3Jm93umzzNznvn
NSSFGRJ6ARAVZvXK7XDaXLyTiR5B7ontuZPQ824I5RKIa6ew+aba8+pU1rVDZ
iciv
```

[multiple lines deleted]

```
/O6DU5wZ7ie2+g4xzDPwCpwm3nqW2GAPcdcLxF4fIP66jHjncmKvKzh/ZUNCx
l9/QQx2HXHYB4m/PkQcdCdx2G70Yt+mszkMh4iZxoifvkh89BFipo87kwD/Bf
/d0cycAAEA:a1b2
```

The parameters are identical to the existing ~DT command:

Table 37 • ~DT Command Parameters

Parameter	Details
O = font name	<i>Accepted Values:</i> any valid TrueType name, up to 8 characters <i>Default Value:</i> if a name is not specified, UNKNOWN is used In this example, Arial is the specified font.
S = font size	<i>Accepted Values:</i> the number of memory bytes required to hold the Zebra-downloadable format of the font <i>Default Value:</i> 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	<i>Accepted Values:</i> a string of ASCII hexadecimal values (two hexadecimal digits/byte). The total number of two-digit values must match parameter S. <i>Default Value:</i> 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

This table identifies the parameters for this format:

Table 38 • Format Parameters

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.



Notes •



Field Interactions

This section provides you with examples that show how commands interact with various justification parameters. The examples are in charts for these orientations:

- Normal
- Rotated
- Bottom-up
- Inverted

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

Table 39 shows you the various normal orientation outputs:

	^FPH	^FPV	^FPR
^FO Left Justified	⊥ABCDE	⊥ A B C D E	EDCBA⊥
^FT Left Justified	⊥ABCDE	⊥ A B C D E	EDCBA⊥
^FO Right Justified	ABCDE⊥	A⊥ B C D E	⊥EDCBA
^FT Right Justified	ABCDE⊥	A⊥ B C D E	⊥EDCBA

Rotated Orientation

Table 40 shows you the various rotated orientation outputs:

Table 40 • Rotated Orientation Examples

	^FPH	^FPV	^FPR
^FO Left Justified	ABCDE	A B C D E	EDCBA
^FT Left Justified	ABCDE	A B C D E	EDCBA
^FO Right Justified	ABCDE	A B C D E	EDCBA
^FT Right Justified	ABCDE	A B C D E	EDCBA

Bottom Up Orientation

Table 41 shows you the various bottom up orientation outputs:

Table 41 • Bottom Orientation Examples			
	^FPH	^FPV	^FPR
^FO Left Justified	ABCDE┐	┐A B C D E	EDCBA┐
^FT Left Justified	┐ABCDE	┐A B C D E	EDCBA┐
^FO Right Justified	ABCDE┐	A B C D E┐	EDCBA┐
^FT Right Justified	ABCDE┐	┐A B C D E	EDCBA┐

Inverted Orientation

Table 42 shows you the various inverted orientation outputs:

	Table 42 • Inverted Orientation Examples		
	^FPH	^FPV	^FPR
^FO Left Justified	+ ABCDE	+ E D C B A	+ EDCBA
^FT Left Justified	+ ABCDE	E D C B + A	+ EDCBA
^FO Right Justified	+ ABCDE	+ E D C B A	+ EDCBA
^FT Right Justified	+ ABCDE	E D C B + A	+ EDCBA



Notes •



Real Time Clock

This appendix contains the information needed to install, program, and operate the Real Time Clock (RTC) option.

Overview

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 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	1. Use the LEFT oval key to select the position to be adjusted 2. Then, use the RIGHT oval key to select the correct value for that position.
Z4Mplus and Z6Mplus printers	1. Press SELECT to select the parameter. 2. Use the MINUS (-) key to select the position to be adjusted 3. Then, use the PLUS (+) key to select the correct value for that position. 4. Press SELECT to accept any changes and deselect the parameter.
S4M printer	1. Press ENTER. The printer displays the current RTC date. 2. 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. 3. Press ENTER to accept the value shown.



Note • Invalid dates, such as 2/29/1999, may be entered, but 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
<i>PAX</i> Series print engines, <i>Xi</i> Series printers, and <i>105SL</i> printers	1. Use the LEFT oval key to select the position to be adjusted 2. Use the RIGHT oval key to select the correct value for that position.
<i>Z4Mplus</i> and <i>Z6Mplus</i> printers	1. Press SELECT to select the parameter. 2. Use the MINUS (-) key to select the position to be adjusted. 3. Use the PLUS (+) key to select the correct value for that position. 4. Press SELECT to accept any changes and deselect the parameter.
<i>S4M</i> printer	1. Press ENTER. The printer displays the current RTC date. 2. 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. 3. 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. See [Table 43](#) for the list of command characters and their functions.

Table 43 • 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*
%W	is replaced by the week# of the year, 00 to 53, Monday is 1st day**
%w	is replaced by the day# of the week, 00 (Sunday) to 06 (Saturday)
%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

*, %U establishes Sunday as the first day of the year.

**, %W establishes Monday as the first day of the year.

The Set Offset **^SO** 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 **^SO** command is required to set the secondary offset; an additional **^SO** 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 **^SL** (see [page 341](#)) 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.



Example • 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.

First Day of the Week	Sun	Mon	Tue	Wed	Thu	Fri	Sat	
							1	
	2	3	4	5	6	7	8	Week 00
	9	10	11	12	13	14	15	Week 01
	16	17	18	19	20	21	22	
	23	24	25	26	27	28	29	
	30	31						

→ **Example • 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.

	Mon	Tue	Wed	Thu	Fri	Sat	Sun	
						1	2	← Week 00
First Day of the Week →	3	4	5	6	7	8	9	← Week 01
	10	11	12	13	14	15	16	
	17	18	19	20	21	22	23	
	24	25	26	27	28	29	30	
	31							

→ **Example • 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.

	Sun	Mon	Tue	Wed	Thu	Fri	Sat	
	No Week 00							
First Day of the Week →	1	2	3	4	5	6	7	← Week 01
	8	9	10	11	12	13	14	
	15	16	17	18	19	20	21	
	22	23	24	25	26	27	28	
	29	30	31					

→ **Example • January, 2006 with Week 00**

Set Monday as the first day of the week using the %W command. In this example, Saturday, January 1st is Week 00 and Sunday, January 2nd begins Week 01.

First Day of the Week	Mon	Tue	Wed	Thu	Fri	Sat	Sun	
							1	← Week 00
	2	3	4	5	6	7	8	← Week 01
	9	10	11	12	13	14	15	
	16	17	18	19	20	21	22	
	23	24	25	26	27	28	29	
	30	31						

Time and Date Precision

As of V60.13.0.1 firmware, the **^CO** 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 **^CO** 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 shown on [page 341](#).

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: (A) _____
Available RAM (in BYTES)
2. Print a Configuration Label and note these values:
Printer "Print Width" (in DOTS) (B) _____
(NOT the Label Width)
Label Length (in DOTS) (C) _____
3. Determine the desired maximum number of queued labels with the (D) _____
same Time and Date value.



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)
 $(B \times C \times D)/8 =$ (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.



Example •

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) =

$(B \times C \times D)/8 = 208000$ BYTES

And —

The **^C0** 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^C0Y,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.



Note • For the 170PAX2 print engine, you must disable the Reprint Function for the Time Now Mode to function properly.

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 10 on page 990](#).

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
    ^S02,3,0,0,1,0,0^FS
    ^S03,0,0,-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 10 on page 990](#) illustrates 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
    ^F010,025^AD^FC%,{,#{^FD1: Mil: %H:%M:%S Civ: %I:%M:%S
    %p^FS
    ^F010,050^AD^FC%,{,#{^FD2: Mil: {H:{M:{S Civ: {I:{M:{S
    {p^FS
    ^F010,075^AD^FC%,{,#{^FD3: Mil: #H:#M:#S Civ: #I:#M:#S
    #p^FS
    ^F010,100^AD^FC%,{,#{^FD1: On %A, %B %d
    , %Y (%a, %m/%d/%y, %d %b %Y).^FS
    ^F010,125^AD^FC%,{,#{^FD2: On {A, {B {d, {Y (
    {a, {m/{d/{y, {d {b {Y).^FS
    ^F010,150^AD^FC%,{,#{^FD3: On #A, #B #d, #Y (
    #a, #m/#d/#y, #d #b #Y).^FS
^XZ

```

Figure 10 • Printed Result of the Above ZPL II Script

```

1: Mil: 14:30:00 Civ: 02:30:00 PM
2: Mil: 15:30:00 Civ: 03:30:00 PM
3: Mil: 14:30:00 Civ: 02:30:00 PM
1: On Wed, April 23, 2005 (Fri, 04/23/05, 23 Apr 2005)
2: On Saturday, July 23, 2005 (Fri, 07/23/05, 23 Jul 2005)
3: On Friday, April 23, 2004 (Fri, 04/23/04, 23 Apr 2004)

```

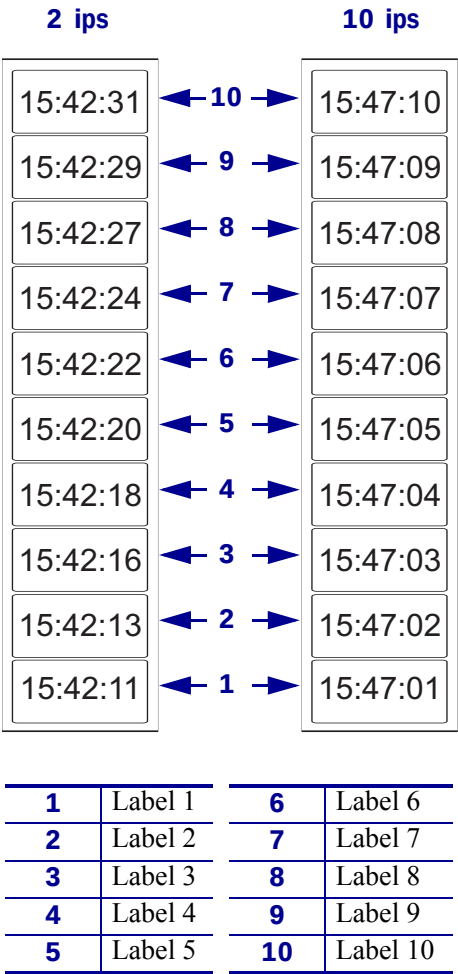
The following are examples of the time stamp using the **^SL1** and **^SL5** at 2 ips and 10 ips for the Enhanced Real Time Clock (V60.13.0.10 and later).



Note • They show the variation of time due to print speed and label complexity.

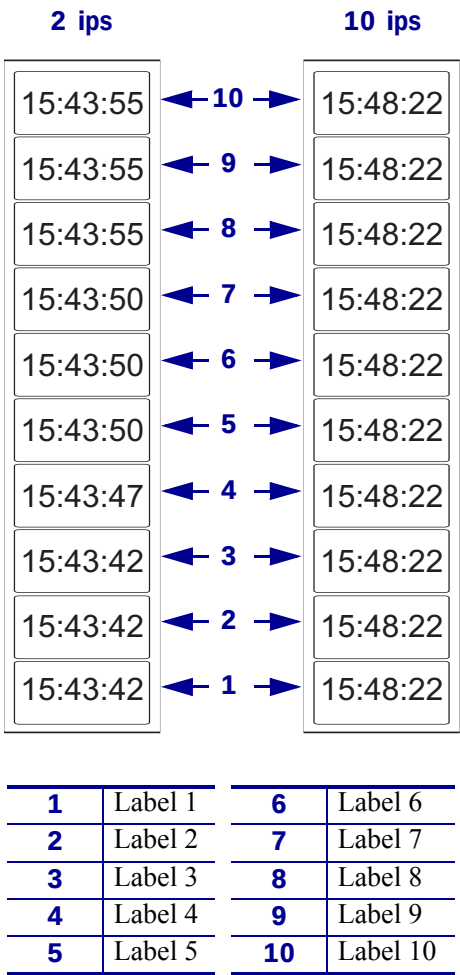
```
^XA
  ^SL1^FS
    ^F0187,184^A0N,101,121^FC%^FD%H:%M:%S^FS
  ^PQ10
^XZ
```

Figure 11 • Example of ^SL1, 2 ips and 10 ips



^XA
 ^SL5^FS
 ^F0187,184^A0N,101,121^FC%^FD%H:%M:%S^FS
 ^PQ10
 ^XZ

Figure 12 • Example of ^SL5, 2 ips and 10 ips



H



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, 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.

Character Set

These are the supported characters for these Zebra printers:

- *XiIIIPlus*
- 105SL
- Z4M/Z6M
- PAX4
- S4M:

Char. code															
	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
	0	0	0	1	1	1	1	0	0	1	1	1	1	1	1
	0	1	1	0	0	1	1	1	1	0	0	1	1	1	1
	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1
xxxx0000			0	0	P	`	F		-	9	3	α	ρ		
xxxx0001		!	1	A	Q	a	q		7	7	4	ä	q		
xxxx0010		"	2	B	R	b	r		「	イ	ツ	×	β	θ	
xxxx0011		#	3	C	S	c	s		」	ウ	テ	モ	ε	ω	
xxxx0100		\$	4	D	T	d	t		、	エ	ト	†	μ	Ω	
xxxx0101		%	5	E	U	e	u		・	オ	ナ	1	σ	ü	
xxxx0110		&	6	F	V	f	v		ヲ	カ	ニ	ヨ	ρ	Σ	
xxxx0111		'	7	G	W	g	w		ア	キ	ヌ	ラ	q	π	
xxxx1000		(	8	H	X	h	x		イ	ク	ネ	リ	」	又	
xxxx1001		)	9	I	Y	i	y		ウ	ケ	ル	「	U		
xxxx1010		*	:	J	Z	j	z		エ	コ	ハ	レ	i	〒	
xxxx1011		+	;	K	[	k	[		オ	サ	ヒ	ロ	*	厶	
xxxx1100		,	<	L	¥	1	l		ハ	シ	フ	ワ	Φ	円	
xxxx1101		-	=	M	]	m	]		ユ	ズ	ハ	ン	も	÷	
xxxx1110		.	>	N	^	n	^		ヨ	セ	ホ	ッ	ん		
xxxx1111		/	?	O	_	o	_		ッ	ッ	マ	°	ö	■	

These are the supported characters for these Zebra printers:

- ZM400 and ZM600
- Xi4

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	



Notes • _____



SGD Command Support

This appendix provides you with details identifying which SGD commands can be used with different printers and firmware versions.

Printer and Firmware Compatibility

This section provides supported printer models and minimum firmware versions for all SGD commands. In order to set and get configurations, the SGD commands can be used through the Ethernet, serial, parallel, and USB interfaces.



Note • The printer responds with the printer setting of "?" if the printer setting:

- does not exist
- has not been configured yet

Printer Type

If you know the printer type that you currently have, answer this question:

What type of printer do you have?

If you have a ...	Then...
Standard Printer	Go to Table 44 on page 999 .
RFID printer	Go to Table 45 on page 1012 .

Standard Printer



Important • All the firmware versions noted in [Table 44](#) refer to the minimum firmware version required. For RFID printer support, see [Table 45](#).

Table 44 • SGD Standard Printer and Firmware Compatibility

SGD Commands	XiIIIPlus , 105SL, PAX4, Z4MPlus, Z6MPlus	S4M	ZM400, ZM600	Xi4	HC100	G-Series	KR403
<i>appl.bootblock</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	V66.17.2Z
<i>appl.name</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>appl.option_board_version</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>bluetooth.address</i>	—	—	—	—	—	V61.15.6Z, V56.15.6Z	—
<i>bluetooth.afh_map</i>	—	—	—	—	—	V61.15.6Z, V56.15.6Z	—
<i>bluetooth.afh_map_curr</i>	—	—	—	—	—	V61.15.6Z, V56.15.6Z	—
<i>bluetooth.afh_mode</i>	—	—	—	—	—	V61.15.6Z, V56.15.6Z	—
<i>bluetooth.authentication</i>	—	—	—	—	—	V61.15.6Z, V56.15.6Z	—
<i>bluetooth.bluetooth_pin</i>	—	—	—	—	—	V61.15.6Z, V56.15.6Z	—
<i>bluetooth.date</i>	—	—	—	—	—	V61.15.6Z, V56.15.6Z	—
<i>bluetooth.discoverable</i>	—	—	—	—	—	V61.15.6Z, V56.15.6Z	—
<i>bluetooth.enable</i>	—	—	—	—	—	V61.15.6Z, V56.15.6Z	—
<i>bluetooth.friendly_name</i>	—	—	—	—	—	V61.15.6Z, V56.15.6Z	—
<i>bluetooth.local_name</i>	—	—	—	—	—	V61.15.6Z, V56.15.6Z	—
<i>bluetooth.radio_auto_baud</i>	—	—	—	—	—	V61.15.6Z, V56.15.6Z	—
<i>bluetooth.radio_version</i>	—	—	—	—	—	V61.15.6Z, V56.15.6Z	—
<i>bluetooth.short_address</i>	—	—	—	—	—	V61.15.6Z, V56.15.6Z	—
<i>bluetooth.version</i>	—	—	—	—	—	V61.15.6Z, V56.15.6Z	—
<i>card.inserted</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	—	V61.15.6Z, V56.15.6Z	—
<i>card.mac_addr</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>CISDFCRC16</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	—	—

* On these printers with ZebraNet™ 10/100 Internal Print Server.

** On these printers with ZebraNet™ 10/100 External Print Server, firmware version 1.1.5

Table 44 • SGD Standard Printer and Firmware Compatibility (Continued)

SGD Commands	XIIIPlus , 105SL, PAX4, Z4MPlus, Z6MPlus	S4M	ZM400, ZM600	Xi4	HC100	G-Series	KR403
<i>comm.type</i>	—	—	—	—	—	V61.17.3Z, V56.17.3Z	—
<i>cutter.clean_cutter</i>	—	—	—	—	—	V61.15.6Z, V56.15.6Z	—
<i>device.download_connection_timeout</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>device.friendly_name</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>device.frontpanel.key_press</i>	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>device.frontpanel.line1</i>	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>device.frontpanel.line2</i>	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>device.frontpanel.xml</i>	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>device.jobs_print</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	—	—
<i>device.languages</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1ZC	V61.15.6Z, V56.15.6Z	—
<i>device.orientation</i>	—	—	—	—	—	—	V66.17.2Z
<i>device.pnp_option</i>	—	—	—	—	—	V61.15.6Z, V56.15.6Z	—
<i>device.reset</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>device.restore_defaults</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>device.unique_id</i>	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>device.uptime</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>device.user_p1</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>device.user_p2</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>device.xml.enable</i>	V60.15.8Z	V53.15.8Z	V53.15.8Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>display.text</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	—	V56.16.5Z	—
<i>external_wired.check</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>external_wired.ip.addr</i>	V60.15.8Z	**V53.15.5Z	**V53.15.5Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—

* On these printers with ZebraNet™ 10/100 Internal Print Server.

** On these printers with ZebraNet™ 10/100 External Print Server, firmware version 1.1.5

Table 44 • SGD Standard Printer and Firmware Compatibility (Continued)

SGD Commands	XIIIPlus , 105SL, PAX4, Z4MPlus, Z6MPlus	S4M	ZM400, ZM600	Xi4	HC100	G-Series	KR403
<i>external_wired.ip.arp_interval</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>external_wired.ip.default_addr_enable</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>external_wired.ip.dhcp.cid_all</i>	V60.15.8Z	**V53.15.5Z	**V53.15.5Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>external_wired.ip.dhcp.cid_enable</i>	V60.15.8Z	**V53.15.5Z	**V53.15.5Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>external_wired.ip.dhcp.cid_prefix</i>	V60.15.8Z	**V53.15.5Z	**V53.15.5Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>external_wired.ip.dhcp.cid_suffix</i>	V60.15.8Z	**V53.15.5Z	**V53.15.5Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>external_wired.ip.dhcp.cid_type</i>	V60.15.8Z	V53.15.5Z	**V53.15.5Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>external_wired.ip.gateway</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>external_wired.ip.netmask</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>external_wired.ip.port</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>external_wired.ip.protocol</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>external_wired.ip.timeout.enable</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>external_wired.ip.timeout.value</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>external_wired.ip.v6.addr</i>	V60.16.5Z	v53.16.5Z	v53.16.5Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>external_wired.ip.v6.gateway</i>	V60.16.5Z	v53.16.5Z	v53.16.5Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>external_wired.ip.v6.prefix_length</i>	V60.16.5Z	v53.16.5Z	v53.16.5Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>external_wired.mac_addr</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>external_wired.mac_raw</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	—	V61.16.5Z, V56.16.5Z	—
<i>file.delete</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>file.dir</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>file.run</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—

* On these printers with ZebraNet™ 10/100 Internal Print Server.

** On these printers with ZebraNet™ 10/100 External Print Server, firmware version 1.1.5

Table 44 • SGD Standard Printer and Firmware Compatibility (Continued)

SGD Commands	XIIIPlus , 105SL, PAX4, Z4MPlus, Z6MPlus	S4M	ZM400, ZM600	Xi4	HC100	G-Series	KR403
<i>file.type</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>head.latch</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>interface.network.active.gateway</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>interface.network.active.ip_addr</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>interface.network.active.mac_addr</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>interface.network.active.mac_raw</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>interface.network.active.netmask</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>interface.network.active.protocol</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	V61.16.5Z, V56.16.5Z	—
<i>internal_wired.auto_switchover</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.addr</i>	—	—	* V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.arp_interval</i>	—	—	* V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.default_addr_enable</i>	—	—	* V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.dhcp.cache_ip</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.dhcp.cid_all</i>	—	—	* V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.dhcp.cid_enable</i>	—	—	* V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.dhcp.cid_prefix</i>	—	—	* V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.dhcp.cid_suffix</i>	—	—	* V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.dhcp.cid_type</i>	—	—	* V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.dhcp.lease.last_attempt</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.dhcp.lease.length</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.dhcp.lease.server</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—

* On these printers with ZebraNet™ 10/100 Internal Print Server.

** On these printers with ZebraNet™ 10/100 External Print Server, firmware version 1.1.5

Table 44 • SGD Standard Printer and Firmware Compatibility (Continued)

SGD Commands	XIIIPlus , 105SL, PAX4, Z4MPlus, Z6MPlus	S4M	ZM400, ZM600	Xi4	HC100	G-Series	KR403
<i>internal_wired.ip.dhcp.lease.time_left</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.dhcp.option12</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.dhcp.option12_format</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.dhcp.option12_value</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.dhcp.request_timeout</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.dhcp.requests_per_session</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.dhcp.session_interval</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.gateway</i>	—	—	* V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.netmask</i>	—	—	* V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.port</i>	—	—	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.protocol</i>	—	—	* V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.timeout.enable</i>	—	—	* V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.ip.timeout.value</i>	—	—	* V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.mac_addr</i>	—	—	* V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>internal_wired.mac_raw</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>ip.active_network</i>	V60.15.8Z		V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.addr</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.arp_interval</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.bootp.enable</i>	V60.15.2Z	V53.15.5Z , V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.dhcp.cache_ip</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.dhcp.cid_all</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—

* On these printers with ZebraNet™ 10/100 Internal Print Server.

** On these printers with ZebraNet™ 10/100 External Print Server, firmware version 1.1.5

Table 44 • SGD Standard Printer and Firmware Compatibility (Continued)

SGD Commands	XIIIPlus , 105SL, PAX4, Z4MPlus, Z6MPlus	S4M	ZM400, ZM600	Xi4	HC100	G-Series	KR403
<i>ip.dhcp.cid_enable</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.dhcp.cid_prefix</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.dhcp.cid_suffix</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.dhcp.cid_type</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.dhcp.enable</i>	V60.15.2Z	V53.15.5Z, V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.dhcp.lease.last_attempt</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.dhcp.lease.length</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.dhcp.lease.server</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.dhcp.lease.time_left</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.dhcp.option12</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.dhcp.option12_format</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.dhcp.option12_value</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.dhcp.request_timeout</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.dhcp.requests_per_session</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.dhcp.session_interval</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.dns.domain</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	—	—
<i>ip.dns.servers</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	—	—
<i>ip.ftp.enable</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.gateway</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.http.enable</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>ip.lpd.enable</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—

* On these printers with ZebraNet™ 10/100 Internal Print Server.

** On these printers with ZebraNet™ 10/100 External Print Server, firmware version 1.1.5

Table 44 • SGD Standard Printer and Firmware Compatibility (Continued)

SGD Commands	XIIIPlus , 105SL, PAX4, Z4MPlus, Z6MPlus	S4M	ZM400, ZM600	Xi4	HC100	G-Series	KR403
<i>ip.mac_raw</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1ZC	—	—
<i>ip.mirror.auto</i>	V60.16.4Z	V53.17.2Z	V53.17.2Z	V53.17.2Z	—	V61.17.3Z, V56.17.3Z	—
<i>ip.mirror.error_retry</i>	V60.17.2Z	V53.17.2Z	V53.17.2Z	V53.17.2Z	—	V61.17.3Z, V56.17.3Z	—
<i>ip.mirror.feedback.auto</i>	V60.17.2Z	V53.17.2Z	V53.17.2Z	V53.17.2Z	—	V61.17.3Z, V56.17.3Z	—
<i>ip.mirror.feedback.freq</i>	V60.17.2Z	V53.17.2Z	V53.17.2Z	V53.17.2Z	—	V61.17.3Z, V56.17.3Z	—
<i>ip.mirror.feedback.odometer</i>	V60.17.2Z	V53.17.2Z	V53.17.2Z	V53.17.2Z	—	V61.17.3Z, V56.17.3Z	—
<i>ip.mirror.feedback.path</i>	V60.17.2Z	V53.17.2Z	V53.17.2Z	V53.17.2Z	—	V61.17.3Z, V56.17.3Z	—
<i>ip.mirror.fetch</i>	V60.17.2Z	V53.17.2Z	V53.17.2Z	V53.17.2Z	—	V61.17.3Z, V56.17.3Z	—
<i>ip.mirror.freq</i>	V60.17.2Z	V53.17.2Z	V53.17.2Z	V53.17.2Z	—	V61.17.3Z, V56.17.3Z	—
<i>ip.mirror.freq_hours</i>	V60.17.2Z	V53.17.2Z	V53.17.2Z	V53.17.2Z	—	V61.17.3Z, V56.17.3Z	—
<i>ip.mirror.last_error</i>	V60.17.2Z	V53.17.2Z	V53.17.2Z	V53.17.2Z	—	V61.17.3Z, V56.17.3Z	—
<i>ip.mirror.last_time</i>	V60.17.2Z	V53.17.2Z	V53.17.2Z	V53.17.2Z	—	V61.17.3Z, V56.17.3Z	—
<i>ip.mirror.password</i>	V60.17.2Z	V53.17.2Z	V53.17.2Z	V53.17.2Z	—	V61.17.3Z, V56.17.3Z	—
<i>ip.mirror.path</i>	V60.17.2Z	V53.17.2Z	V53.17.2Z	V53.17.2Z	—	V61.17.3Z, V56.17.3Z	—
<i>ip.mirror.server</i>	V60.17.2Z	V53.17.2Z	V53.17.2Z	V53.17.2Z	—	V61.17.3Z, V56.17.3Z	—
<i>ip.mirror.success</i>	V60.17.2Z	V53.17.2Z	V53.17.2Z	V53.17.2Z	—	V61.17.3Z, V56.17.3Z	—
<i>ip.mirror.success_time</i>	V60.17.2Z	V53.17.2Z	V53.17.2Z	V53.17.2Z	—	V61.17.3Z, V56.17.3Z	—
<i>ip.mirror.username</i>	V60.17.2Z	V53.17.2Z	V53.17.2Z	V53.17.2Z	—	V61.17.3Z, V56.17.3Z	—
<i>ip.mirror.version</i>	V60.17.2Z	V53.17.2Z	V53.17.2Z	V53.17.2Z	—	V61.17.3Z, V56.17.3Z	—
<i>ip.netmask</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>ip.pop3.enable</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—

* On these printers with ZebraNet™ 10/100 Internal Print Server.

** On these printers with ZebraNet™ 10/100 External Print Server, firmware version 1.1.5

Table 44 • SGD Standard Printer and Firmware Compatibility (Continued)

SGD Commands	XIIIPlus , 105SL, PAX4, Z4MPlus, Z6MPlus	S4M	ZM400, ZM600	Xi4	HC100	G-Series	KR403
<i>ip.pop3.password</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>ip.pop3.poll</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>ip.pop3.server_addr</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>ip.pop3.username</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>ip.port</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>ip.primary_network</i>	V53.15.5Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>ip.smtp.domain</i>	V60.15.2Z	V53.15.5Z, V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>ip.smtp.enable</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>ip.smtp.server_addr</i>	V60.15.2Z	V53.15.5Z, V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>ip.snmp.enable</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>ip.snmp.get_community_name</i>	V60.15.2Z	V53.15.5Z, V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>ip.snmp.set_community_name</i>	V60.15.2Z	V53.15.5Z, V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>ip.tcp.enable</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>ip.telnet.enable</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>ip.udp.enable</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>media.cut_now</i>	—	—	—	—	—	—	V66.17.2Z
<i>media.darkness_mode</i>	—	—	—	—	V54.16.1Z	—	—
<i>media.media_low.external</i>	—	—	—	—	—	—	V66.17.2Z
<i>media.media_low.warning</i>	—	—	—	V53.17.1Z	—	—	—
<i>media.present.cut_amount</i>	—	—	—	—	—	—	V66.17.2Z
<i>media.present.cut_margin</i>	—	—	—	—	—	—	V66.17.2Z

* On these printers with ZebraNet™ 10/100 Internal Print Server.

** On these printers with ZebraNet™ 10/100 External Print Server, firmware version 1.1.5

Table 44 • SGD Standard Printer and Firmware Compatibility (Continued)

SGD Commands	XIIIPlus , 105SL, PAX4, Z4MPlus, Z6MPlus	S4M	ZM400, ZM600	Xi4	HC100	G-Series	KR403
<i>media.present.eject</i>	—	—	—	—	—	—	V66.17.2Z
<i>media.present.length_addition</i>	—	—	—	—	—	—	V66.17.2Z
<i>media.present.loop_length</i>	—	—	—	—	—	—	V66.17.2Z
<i>media.present.loop_length_max</i>	—	—	—	—	—	—	V66.17.2Z
<i>media.present.present_timeout</i>	—	—	—	—	—	—	V66.17.2Z
<i>media.present.present_type</i>	—	—	—	—	—	—	V66.17.2Z
<i>media.printmode</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>media.speed</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.6Z, V56.16.6Z	—
<i>odometer.headclean</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>odometer.headnew</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>odometer.label_dot_length</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>odometer.media_marker_count1</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>odometer.media_marker_count2</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>odometer.retracts_count</i>	—	—	—	—	—	—	V66.17.2Z
<i>odometer.total_print_length</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.15.6Z, V56.15.6Z	—
<i>print.tone</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.6Z, V56.16.6Z	—
<i>wlan.adhocautomode</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.adhocchannel</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.associated</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.bssid</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.channel</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—

* On these printers with ZebraNet™ 10/100 Internal Print Server.

** On these printers with ZebraNet™ 10/100 External Print Server, firmware version 1.1.5

Table 44 • SGD Standard Printer and Firmware Compatibility (Continued)

SGD Commands	XIIIPlus , 105SL, PAX4, Z4MPlus, Z6MPlus	S4M	ZM400, ZM600	Xi4	HC100	G-Series	KR403
<i>wlan.channel_mask</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.current_tx_rate</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.essid</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.firmware_version</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.ip.addr</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.ip.arp_interval</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.ip.default_addr_enable</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.ip.dhcp.cache_ip</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.ip.dhcp.cid_all</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.ip.dhcp.cid_enable</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.ip.dhcp.cid_prefix</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.ip.dhcp.cid_suffix</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.ip.dhcp.cid_type</i>	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.ip.dhcp.lease.last_attempt</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.ip.dhcp.lease.length</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.ip.dhcp.lease.server</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.ip.dhcp.lease.time_left</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.ip.dhcp.option12</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.ip.dhcp.option12_format</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.ip.dhcp.option12_value</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.ip.dhcp.request_timeout</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—

* On these printers with ZebraNet™ 10/100 Internal Print Server.

** On these printers with ZebraNet™ 10/100 External Print Server, firmware version 1.1.5

Table 44 • SGD Standard Printer and Firmware Compatibility (Continued)

SGD Commands	XIIIPlus , 105SL, PAX4, Z4MPlus, Z6MPlus	S4M	ZM400, ZM600	Xi4	HC100	G-Series	KR403
wlan.ip.dhcp.requests_per_session	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.ip.dhcp.session_interval	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.ip.gateway	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.ip.netmask	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.ip.port	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.ip.protocol	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.ip.timeout.enable	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.ip.timeout.value	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.keep_alive.enable	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.keep_alive.timeout	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.kerberos.kdc	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.kerberos.password	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.kerberos.realm	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.kerberos.username	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.mac_addr	V60.15.8Z	V53.15.5Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.mac_raw	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.operating_mode	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.password	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.preamble	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.private_key_password	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
wlan.roam.interchannel_delay	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—

* On these printers with ZebraNet™ 10/100 Internal Print Server.

** On these printers with ZebraNet™ 10/100 External Print Server, firmware version 1.1.5

Table 44 • SGD Standard Printer and Firmware Compatibility (Continued)

SGD Commands	XIIIPlus , 105SL, PAX4, Z4MPlus, Z6MPlus	S4M	ZM400, ZM600	Xi4	HC100	G-Series	KR403
<i>wlan.roam.interval</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.roam.max_chan_scan_time</i>	V60.16.4Z	V53.16.4Z	V53.16.4Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.roam.signal</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.security</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.signal_noise</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.signal_quality</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.signal_strength</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.station_name</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.tx_power</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.tx_rate</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.username</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.wep.auth_type</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.wep.index</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.wep.key_format</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.wep.key1</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.wep.key2</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.wep.key3</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.wep.key4</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>wlan.wpa.psk</i>	V60.15.2Z	V50.15.2Z	V53.15.5Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>zbi.control.add_breakpoint</i>	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
<i>zbi.control.break</i>	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—

* On these printers with ZebraNet™ 10/100 Internal Print Server.

** On these printers with ZebraNet™ 10/100 External Print Server, firmware version 1.1.5

Table 44 • SGD Standard Printer and Firmware Compatibility (Continued)

SGD Commands	XIIIPlus , 105SL, PAX4, Z4MPlus, Z6MPlus	S4M	ZM400, ZM600	Xi4	HC100	G-Series	KR403
zbi.control.clear_breakpoints	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
zbi.control.delete_breakpoint	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
zbi.control.line_number	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
zbi.control.restart	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
zbi.control.run	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
zbi.control.step	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
zbi.control.terminate	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
zbi.control.variable_name	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
zbi.control.variable_value	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
zbi.key	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
zbi.last_error	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
zbi.reseller_key	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
zbi.revision	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
zbi.running_program_name	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
zbi.start_info.execute	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
zbi.start_info.file_name	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
zbi.start_info.memory_alloc	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—
zbi.state	V60.16.2Z	V53.16.2Z	V53.16.2Z	V53.17.1Z	V54.16.1Z	V61.16.5Z, V56.16.5Z	—

* On these printers with ZebraNet™ 10/100 Internal Print Server.

** On these printers with ZebraNet™ 10/100 External Print Server, firmware version 1.1.5

RFID Printers



Important • All the firmware versions noted in [Table 45](#), refer to minimum firmware version required. For non-RFID printers, see [Table 44](#). Once an RFID Ready printer has been reconfigured as a RFID Printer, [Table 45](#) should be used

Table 45 • SGD RFID Printer and Firmware Compatibility

SGD Commands	R110Xi (UHF) and R170Xi	R110PAX4	R4Mplus	R110Xi HF
<i>card.inserted</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>card.mac_addr</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.addr</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.arp_interval</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.bootp.enable</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.dhcp.cid_all</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.dhcp.cid_enable</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.dhcp.cid_prefix</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.dhcp.cid_suffix</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.dhcp.cid_type</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.dhcp.enable</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.ftp.enable</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.gateway</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z

* On these printers with ZebraNet™ 10/100 Internal Print Server.

** On these printers with ZebraNet™ 10/100 External Print Server, firmware version 1.1.5

Table 45 • SGD RFID Printer and Firmware Compatibility

SGD Commands	R110Xi (UHF) and R170Xi	R110PAX4	R4Mplus	R110Xi HF
<i>ip.http.enable</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.lpd.enable</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.netmask</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.pop3.enable</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.pop3.password</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.pop3.poll</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.pop3.server_addr</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.pop3.username</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.port</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.primary_network</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.smtp.domain</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.smtp.enable</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.smtp.server_addr</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.snmp.enable</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.snmp.get_community_name</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.snmp.set_community_name</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z

* On these printers with ZebraNet™ 10/100 Internal Print Server.

** On these printers with ZebraNet™ 10/100 External Print Server, firmware version 1.1.5

Table 45 • SGD RFID Printer and Firmware Compatibility

SGD Commands	R110Xi (UHF) and R170Xi	R110PAX4	R4Mplus	R110Xi HF
<i>ip.tcp.enable</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.telnet.enable</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>ip.udp.enable</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>media.printmode</i>				
<i>rfid.error.response</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>rfid.position.program</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>rfid.reader_1.antenna_port</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>rfid.reader_1.power.read</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>rfid.reader_1.power.single_power</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>rfid.reader_1.power.write</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.adhocautomode</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.adhocchannel</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.associated</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.channel_mask</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.essid</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.firmware_version</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z

* On these printers with ZebraNet™ 10/100 Internal Print Server.

** On these printers with ZebraNet™ 10/100 External Print Server, firmware version 1.1.5

Table 45 • SGD RFID Printer and Firmware Compatibility

SGD Commands	R110Xi (UHF) and R170Xi	R110PAX4	R4Mplus	R110Xi HF
<i>wlan.keep_alive.enable</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.keep_alive.timeout</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.kerberos.kdc</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.kerberos.password</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.kerberos.realm</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.kerberos.username</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.operating_mode</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.password</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.preamble</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.private_key_password</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.roam.interval</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.roam.signal</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.security</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.signal_noise</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.signal_quality</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.signal_strength</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z

* On these printers with ZebraNet™ 10/100 Internal Print Server.

** On these printers with ZebraNet™ 10/100 External Print Server, firmware version 1.1.5

Table 45 • SGD RFID Printer and Firmware Compatibility

SGD Commands	R110Xi (UHF) and R170Xi	R110PAX4	R4Mplus	R110Xi HF
<i>wlan.station_name</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.tx_power</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.tx_rate</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.username</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.wep.auth_type</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.wep.index</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.wep.key_format</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.wep.key1</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.wep.key2</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.wep.key3</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.wep.key4</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z
<i>wlan.wpa.psk</i>	R60.15.8Z	R62.15.8Z R63.15.8Z	SP994P SP999F SP1027F SP1056E SP1082F	R65.15.8Z

* On these printers with ZebraNet™ 10/100 Internal Print Server.

** On these printers with ZebraNet™ 10/100 External Print Server, firmware version 1.1.5



Features

This appendix covers features added to major firmware releases.

Firmware x.17



Identifies features that are available in printers with firmware version Vx.17.x or later.

The following is an overview of the new and enhanced features available in firmware version V53.17.x, V56.17.xZ, V60.17.x, V61.17.xZ, and later.

Mirror

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 can initiate Mirror. For details see the ZPL Manual.

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.
- Supported printers and print servers:

Printer	Print Servers
Xi4™ -series	• Internal, Integrated 10/100 IPV4 wired • Internal Wireless Plus
ZM400™	• Internal, Integrated 10/100 IPV4 wired • Internal Wireless Plus
ZM600™	• Internal, Integrated 10/100 IPV4 wired • Internal Wireless Plus
S4M™	• Internal Wireless Plus
Print Servers connected to the parallel port, either externally or internally, are not supported for Mirror use.	

Wireless Markup Language (WML)

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.

Firmware x.16



Identifies features that are available in printers with firmware version V60.16.x, V53.16.x, or later.

The following is an overview of the new and enhanced features available in firmware version V53.16.x and V60.16.x, and later; it is ZBI 2.0-Ready. These include:

- ZBI 2.0 can be enabled by placing a ZBI 2.0 key on the printer via the ZBI Key Manager & Downloader utility. Printers can be ordered from the factory with the ZBI 2.0 option enabled. Printers can be ZBI-Enabled in the field by purchasing and using a ZBI Key Kit. Printers can be ZBI-Enabled in the field by purchasing a key at www.zebrasoftware.com.
- Printers which have been ZBI-Enabled will not display a “Z” at the end of the firmware version string. For example, a printer which has been ZBI-Enabled will display the firmware version as “V53.16.1”, while a printer which has not been ZBI-Enabled will display the firmware version as “V53.16.1Z”
- The printer configuration label will display the ZBI status for the printer:
 - ZBI-Enabled

```
|| ENABLED..... ZBI
|| 2.0..... ZBI VERSION
|| READY..... ZBI STATUS ||
```

- Not ZBI-Enabled

```
|| DISABLED..... ZBI
|| 2.0..... ZBI VERSION ||
```

- The printer configuration label will list which ZBI program is running.

```
|| ENABLED..... ZBI
|| 2.0..... ZBI VERSION
|| CHOICES.BAS..... ZBI STATUS ||
```

- ZBI 2.0 is backwards compatible with previous versions of ZBI. Any code which was specifically written to handle a maximum string length of 255 characters will need to be reviewed to ensure it works with unlimited string lengths.

- Other ZBI 2.0 Features:
 - List ZBI programs from the front panel
 - Start/Stop ZBI programs from the front panel
 - Button presses on front panel can be registered as events in ZBI programs.
 - Applicator port control
 - Support for longer strings, the length of the string dependent on available memory.
 - Support for NULL values in Strings and String operations
 - SNMP & Set/Get/Do (SGD) Control of ZBI
 - On-printer Debugging (via ZBI-Developer)
 - ZBI 2.0 Program Encryption
 - ZBI 2.0 programs can be made hidden and/or persistent
- New ZBI 2.0 Commands:

ACCEPT	INTTOHEX\$
ADDBREAK	READ
AUXPORT_GETPIN	REGISTEREVENTS
AUXPORT_RELEASEPIN	SERVERSOCKET
AUXPORT_SETPIN	SPLIT
AUXPORT_STEALPIN	SPLITCOUNT
CLIENTSOCKET	SUB
CLOSE ALL	TCPX
DELBREAK	TRIGGEREVENT
HANDLEEVENT	UNREGISTEREVENTS
HEXTPOINT	WRITE

Firmware x.15



Identifies features that are available in printers with firmware version V60.15.x, V50.15.x, or later.

The following is an overview of the new and enhanced features available in firmware version V60.15.x, V50.15.x, or later. These include:

- Additional wireless securities
- New and updated ZPL commands (including wireless commands)
- Extended SNMP features, providing SNMP based control over commonly-needed printer settings.
- Support for Set / Get / Do (SGD) commands

Wireless Securities

Firmware V60.15.x and V50.15.x offer support for a wider range of wireless securities. These are the supported wireless securities:

- WEP-40-BIT
- WEP-128-BIT
- EAP-TLS
- EAP-TTLS
- EAP-FAST
- PEAP
- LEAP
- WPA-PSK
- WPA EAP-TLS
- WPA EAP-TTLS
- WPA EAP-FAST
- WPA EAP
- WPA LEAP
- KERBEROS*

* Use of any given wireless security is dependent on the RF card being used.

ZPL and SGD Commands

Table 46 identifies the new commands and updates to existing commands in firmware version V60.15.x, V50.15.x, or later. The bolded items are either new commands or new enhancements to existing commands.

Table 46 • x.14 Features in Detail

Command	Details
^KC	The ^KC command is a new command with firmware version V60.15.x, V50.15.x, or later. It allows wireless print servers to have their own client identifier (CID). When the print server is active it uses the CID on the printer. When both print servers are active they have the option to use the CID on the printer, or their own CID.
~DY	Firmware version V60.15.x, V50.15.x, or later supports .NRD (non readable file) files and .PAC files (protected access credential).
~WS	Firmware version V60.15.x, V50.15.x, or later added these new parameters: • wireless pulse—adds a pulse to the network traffic generated by the printer. To keep the printer online, a pulse is necessary with some network configurations. • wireless pulse interval—when the wireless pulse feature is enabled this sets the interval at which the wireless pulse is sent. • channel mask—for commonly used channel masks. • international mode—disables or enables International mode
^WX	The ^WX command configures the wireless security settings for printers using the wireless print server.

SNMP

Firmware V60.15.x, V50.15.x, or later feature support for an extended Management Information Base (MIB) file. The extended support now covers many of the printer settings, such as darkness, media type, and other printer specific settings. The MIB file can be downloaded from www.zebra.com for use with SNMP management systems.

Set / Get / Do Support

Printers with firmware V60.15.x and V50.15.x support a new set of commands known as Set / Get / Do (SGD) commands. These commands allow you to configure and get status on the wireless settings in the printer. For details on the supported commands, see [SGD Printer Commands on page 625](#).

XML-Enabled Printing

Printers with firmware V60.15.8Z, V53.15.8Z, or later support XML-Enabled printing. For details, go to <http://www.zebra.com/xml>. This new feature does not alter the ZPL capabilities of the printer.

Firmware x.14



Identifies features that are available in printers with firmware version V60.14.x, V50.14.x, or later.

[Table 47](#) identifies the new commands and enhancements to existing commands in firmware V60.14.x, V50.14.x, or later. The bolded items are either new commands or new enhancements to existing commands.

Table 47 • Features in Detail

Command	Details
^CI	The ^CI command selects the encoding and character set. To identify the encoding, include a ^CI in the ZPL script. These encodings are now supported with the ^CI command: These are new variables that were added to the desired character set or a parameter of the existing ^CI command: • Big 5 HKSCS (^CI26) with BIGHK . DAT. This supports the Hong Kong supplementary character set of the Big 5 character set and encoding. You can download the Big 5 HKSCS DAT table from www.zebra.com. • Code Page 1252 (^CI27). The ^CI27 command supports code page 1252. • GB 18030 (^CI26) with GB18030 . DAT. This supports the GB 18030 character set and encoding standard. • Unicode Support. These new values were added to the ^CI command: 28 = Unicode (UTF-8 encoding) 29 = Unicode (UTF-16 Big-Endian encoding) 30 = Unicode (UTF-16 Little-Endian encoding)
~DY	The ~DY command downloads fonts to the printer. Firmware version V60.14.x, V50.14.x, or later improves support for TrueType fonts and adds support for TrueType Extension fonts. When OpenType fonts can be downloaded as a TrueType font they are supported.
^FH	The ^FH command allows you to enter the hexadecimal value for any character directly into the ^FD statement. It has been updated to function with Unicode encodings.
^FL	The ^FL command provides the ability to link any TrueType font, including private character fonts to associated fonts, such that 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.
^FO, ^FT, ^FW	In firmware version earlier than V60.15.x, V50.15.x or later, the ^FO and ^FT commands were left-justified. In firmware version V60.14.x, V50.14.x, or later, a new parameter supports right- and auto-justification. Using the ^FW command allows you to set justification options.
^HT	Over a communication port, the ^HT command receives the complete list of the font links.

Table 47 • Features in Detail

Command	Details
^LF	The ^LF command prints out a list of the linked fonts.
^PA	The Advanced Text Layout Options command (^PA) is new in firmware version V60.14.x, V50.14.x, or later. Advanced text layout, such as bidirectional text layout, character shaping, and OpenType support, can be activated with the ^PA command. The glyph displayed when the character is missing from a font (default glyph) can be changed from a space character to the default glyph of a font, which is often a square box.
^SL	A new parameter was added to the ^SL command to support the Japanese language in the Real Time Clock feature set.
^TB	The text block (^TB) command is a new command in firmware version V60.14.x, V50.14.x, or later. The ^FB command allows you to print text into a defined block type format; it does not support complex text layout. In contrast, the ^TB command supports a text block with complex text.

Global Printing Solution

The Global Printing solution greatly simplifies international character label printing by supporting Unicode compatible fonts and encoding (UTF-8, UTF-16BE and UTF-16LE) directly in its printers.

These are new and enhanced features available in firmware version V60.14.x, v50.14.x, or later:

- Advanced text layout options
- Big 5 HKSCS
- GB 18030
- Unicode support (UTF-8 and UTF-16)
- Code Page 1252
- Field hexadecimal
- Field origin justification
- Font linking
- Improved font support
- Real-time clock – Japanese support
- Text block

For details on these new features, see [Table 46 on page 1022](#).

Zebra firmware V60.14.x, V50.14.x, or later continues to support locale-specific encoding formats. These formats include:

- Code Page 850
- Shift JIS
- GB
- Unified Hangul Code
- Big 5

What Is Unicode?

Unicode is a universal code page (which converts numeric codes into characters) that can be used with many computer platforms and software applications. While ASCII and other traditional code page encoding systems support 256 characters or less, Unicode can support almost 100,000 characters for all major languages. The languages include complex non-Western languages that can be difficult to print.

Unicode encoding and supporting fonts allow the printer to seamlessly output in any language. As a result, you do not have to select the language, font, code page, or configure or adjust the printer. Zebra's printer-resident capabilities provide fast label output and seamless multi-language printing.

You can create private libraries of logos and special characters and include them in labels. The ability to create custom characters is valuable for creating shipping labels for Asia, as many Asian fonts don't support characters for names, street names, and other proper nouns. Firmware V60.14.x, V50.14.x, or later offers features that support bidirectional printing or complex text layout, including:

- Arabic
- Hindi
- Thai
- Vietnamese
- Other languages



Notes •



Mirror

This section provides an overview of Mirror, details on how to use it, and configuration examples.

Contents

Mirror Overview	1028
Professional Services for Mirror Configuration.	1030
Requirements.	1030
How Mirror Works	1032
Creating ZPL Files for Use in the "<update-root>/files" Directory	1034
Configuration	1038
The Feedback.get File	1039
How to Set Up and Use Mirror	1042
Troubleshooting	1044

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 on page 1038](#).

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.

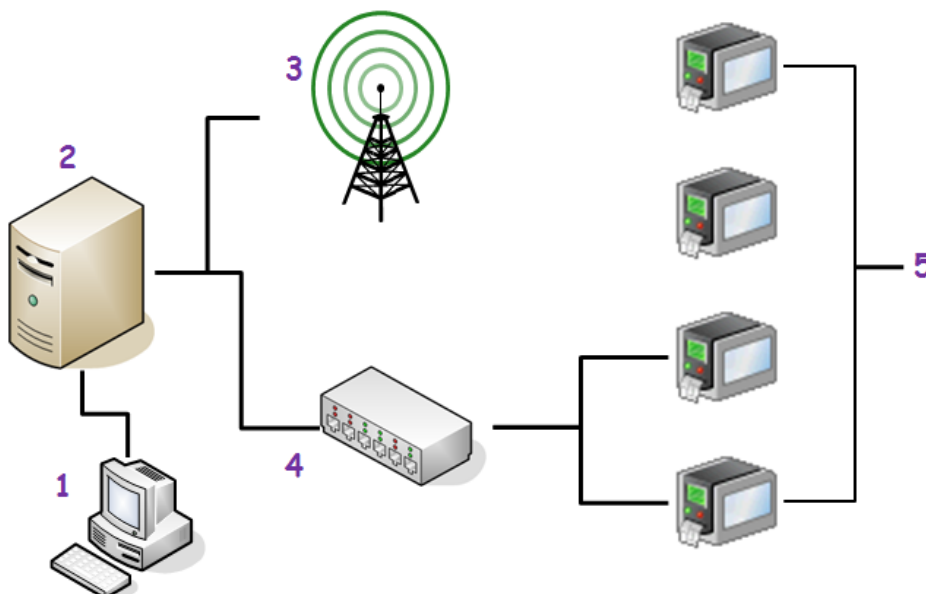
Through the "Feedback" feature, Administrators can easily log and monitor configuration updates on a printer-by-printer basis.

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

[Figure 13](#) provides an illustration of Mirroring.

Figure 13 • 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, please 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 [Table 48, *Printers and Print Servers that Support Mirror* on page 1031](#).
- 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 48 • Printers and Print Servers that Support Mirror

Printer	Firmware	Print Servers
Xi4™ series	V53.17.2Z (or later)	• Internal, Integrated 10/100 IPV4 wired • Internal Wireless Plus
XiIIIPlus™ series	V60.17.2Z (or later)	• Internal Wireless Plus • Wireless Plus
105SL™	V60.17.2Z (or later)	• Internal Wireless Plus • Wireless Plus
PAX4™	V60.17.2Z (or later)	• Internal Wireless Plus • Wireless Plus
ZM400™	V53.17.2Z (or later)	• Internal 10/100 wired • Internal Wireless Plus • Wireless Plus
ZM600™	V53.17.2Z (or later)	• Internal 10/100 wired • Internal Wireless Plus • Wireless Plus
S4M™	V53.17.2Z (or later)	• Internal Wireless Plus • Wireless Plus
G™ -series	V56.17.2Z V61.17.2Z (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 “V53.15.2Z”, the filename stored on the FTP server must be “V53.15.2Z.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 “V53.15.2Z.ZPL” and “v53.15.2z.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`, `! C1SDFCRC16` or `~DF`.



Important • 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
^F020,100^IME:ZEBRA.BMP^FS
^A@N,75,75,TT0003M_.TTF
^F020,400^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
^F020,100^IME:ZEBRA.BMP^FS
^A@N,75,75,TT0003M_.TTF
^F020,400^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 Prefix The default is the Caret (^)	HEX 1E
Delimiter Prefix The default is the comma (,)	HEX 1F
Control Prefix The default is the tilde (~)	HEX 10

For example, given the following format:

```
^XA
^F020,100^IME:ZEBRA.BMP^FS
^A@N,75,75,TT0003M_.TTF
^F020,400^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 on page 1037](#) 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 on page 1037](#) 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
^F025,25^AD,36,20^FN1^FS
^F0165,25^AD,36,20^FN2^FS
^F025,75^AB,22,14^FDBUILT BY^FS
^F025,125^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:

```
^F025,25^AD,36,20^FN1^FS
^F0165,25^AD,36,20^FN2^FS
^F025,75^AB,22,14^FDBUILT BY^FS
^F025,125^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

- 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
 - ~R0
- 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:

- 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 on page 820 • ip.mirror.error_retry on page 821 • ip.mirror.feedback.auto on page 822 • ip.mirror.feedback.freq on page 823 • ip.mirror.feedback.odometer on page 824 • ip.mirror.feedback.path on page 825 • ip.mirror.fetch on page 826 • ip.mirror.freq on page 827 • ip.mirror.freq_hours on page 828	• ip.mirror.last_error on page 829 • ip.mirror.last_time on page 830 • ip.mirror.password on page 831 • ip.mirror.path on page 832 • ip.mirror.server on page 833 • ip.mirror.success on page 834 • ip.mirror.success_time on page 835 • ip.mirror.username on page 836 • ip.mirror.version on page 837

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 on page 632](#).



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 "/" .



Example • 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.

```

1→ ! U1 SETVAR "ip.mirror.auto" "on"
2→ ! U1 SETVAR "ip.mirror.username" "printer"
3→ ! U1 SETVAR "ip.mirror.password" "printer"
4→ ! U1 SETVAR "ip.mirror.server" "10.14.5.133"
5→ ! U1 SETVAR "ip.mirror.path" "/all_printers/s4m/role1"
6→ ! U1 SETVAR "ip.mirror.feedback.auto" "on"
7→ ! U1 SETVAR "ip.mirror.feedback.path" "/all_feedback"
8→ ! U1 SETVAR "ip.mirror.feedback.freq" "0"

```

1	Configures the printer to perform a Mirror Update operation at power-up.
2	Configures the FTP Server "user name" for the printer to use
3	Configures the FTP Server "password" for the printer to use
4	Configures the FTP server address the printer should make a FTP connection to.
5	If the FTP connection is successful, the printer should attempt to perform an Update operation using this root directory.
6	Configures a printer to automatically perform a Mirror Feedback operation at start-up.
7	Configures the printer to upload the resulting Feedback file to the designated directory on the server.
8	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.



Example • 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.

```

1→ ! U1 SETVAR "ip.mirror.auto" "off"
2→ ! U1 SETVAR "ip.mirror.username" "printer"
3→ ! U1 SETVAR "ip.mirror.password" "printer"
4→ ! U1 SETVAR "ip.mirror.server" "10.14.5.133"
5→ ! U1 SETVAR "ip.mirror.path" "/all_printers/s4m/role1"
6→ ! U1 SETVAR "ip.mirror.feedback.auto" "on"
7→ ! U1 SETVAR "ip.mirror.feedback.path" "/all_feedback"
8→ ! U1 SETVAR "ip.mirror.feedback.freq" "60"

```

1	Configures the printer to not perform a Mirror Update operation at start-up
2	Configures the FTP Server "user name" for the printer to use
3	Configures the FTP Server "password" for the printer to use
4	Configures the FTP server address the printer should make a FTP connection to.
5	If the FTP connection is successful, the printer should attempt to perform an Update operation using this root directory.
6	Configures a printer to automatically perform a Mirror Feedback operation at start-up.
7	Configures the printer to upload the resulting Feedback file to the designated directory on the server.
8	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 49 lists possible printer responses, an explanation of each, and resolutions. Table 50 provides problem scenarios and solutions.



Important • A mirror path can have up to 50 characters.

Table 49 • 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.

Table 49 • Printer Response Troubleshooting (Continued)

Printer Response	Explanation	Resolution
"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.
"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 50 • 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 <code><mirror_path>/appl</code> matches the version of firmware contained in that file. Example • For firmware version V53.17.2Z, the name of the file in the <code><mirror_path>/appl</code> directory must be <code>V53.17.2Z.ZPL</code> 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 <code><mirror_path>/files</code> and <code><mirror_path>/commands</code> 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 <code><mirror_path>/files</code> and <code><mirror_path>/commands</code> 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 <code><mirror_path>/files</code> and <code><mirror_path>/commands</code> directories.
The printer is continually rebooting.	• Check the <code>/commands</code> directory. If a <code>file.delete</code> command resides, then you need to remove <code>file.delete</code> 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.

Contents

WML Overview	1048
WML Details	1048
Supported Printers	1049
Professional Services for WML Content Creation	1050
WML Tags	1050
Using WML	1050
WML Examples	1055
Troubleshooting Scenarios	1063

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 Printer Commands on page 625](#). For details on ZPL commands, see [ZPL Commands on page 39](#).

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 on page 1050](#).

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 printer's 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.



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.

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.

Printer	Firmware	Number of "lines" available	Menu Navigation Buttons	Keys to Access Standard Menu System
Xi4™ series	V53.17.2Z (or later)	5	Select + (PLUS) - (MINUS)	Hold down CANCEL & SETUP/EXIT during power-up
ZM400™	V53.17.2Z (or later)	5	Select + (PLUS) - (MINUS)	Hold down CANCEL & SETUP/EXIT during power-up
ZM600™	V53.17.2Z (or later)	5	NEXT/SAVE + (PLUS) - (MINUS)	Hold down CANCEL & SETUP/EXIT during power-up
G™ -series	V56.17.2Z (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, please contact your Zebra account representative.

WML Tags

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.

Using WML



Important • Using end tags is required to create well formed and functional WML scripts.

<code><wml> </wml></code>	indicates the beginning/end of the WML script
<code><display> </display></code>	indicates the beginning/end of the content to display on-screen
<code><card> </card></code>	indicates the beginning/end of a card
<code><p> </p></code>	indicates the beginning/end of a Paragraph
<code>
</code>	Line break
<code>Menu</code>	Hyperlink to another card
<code><timer value="xx"> </timer></code>	Controls display timer in 10 th of a second increments
<code>" ontimer="#main"</code>	Controls action to take at timer end
<code>alerts="on"</code>	Controls display of on-screen alerts
<code>\$(command.command)</code>	\$ executes a SGD “get” command
<code><do><setvar></do></code>	Controls execution of do and setvar commands

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.

Create a Sample index.wml File:

1. Open a text editor.
2. Type (or copy/paste) the following text:

```
<wml>
<display>
  <card>
    <p>Hello World!!</p>
  </card>
</display>
</wml>
```

3. Save this file with this name: `index.wml`.

Prepare the Printer to Receive WML Content via FTP:

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.

1. Set "ip.ftp.enable" to "on" and the "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"
```

2. 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

1. Go to a command prompt.
2. 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
```
3. Press Enter to connect to the printer.
4. Press Enter to log in to the printer.
5. At the FTP prompt, type "`put index.wml`" and press Enter. The `index.wml` file will be transferred to the printer's E: memory.
6. Type "`quit`" to disconnect from the printer and exit FTP.
7. 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 on page 1055](#).



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

1. 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:

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 [page 632](#).

1. To send the sample index.wml shown earlier, send the following commands to the printer:

```
! CISDFCRC16
0000
INDEX.WML
00000004E
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 on page 1055](#).



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:

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, please note that .nrd files are treated as confidential – they cannot be retrieved from the printer.

Using .nrd Files from WML Menus

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:

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.

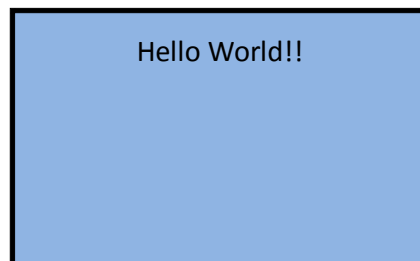
Example 1

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:



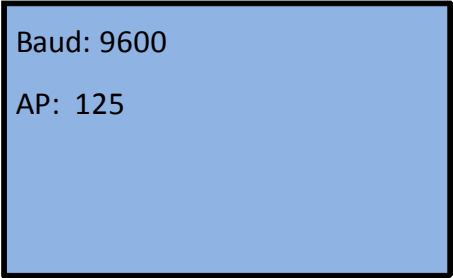
Example 2

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.

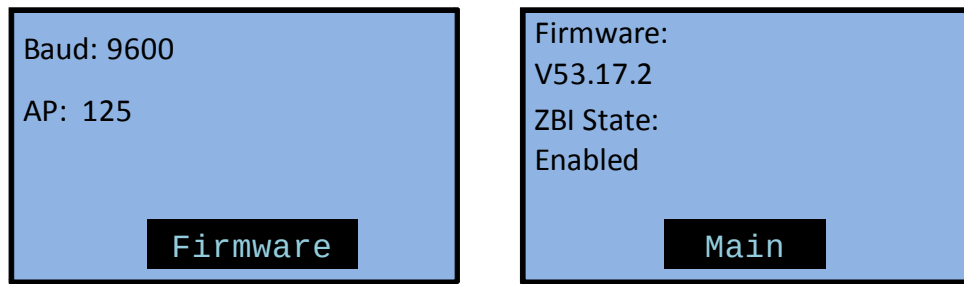
```

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></p>
19→    </card>
20→ </display>
21→ </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 10 th 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 10 th 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.

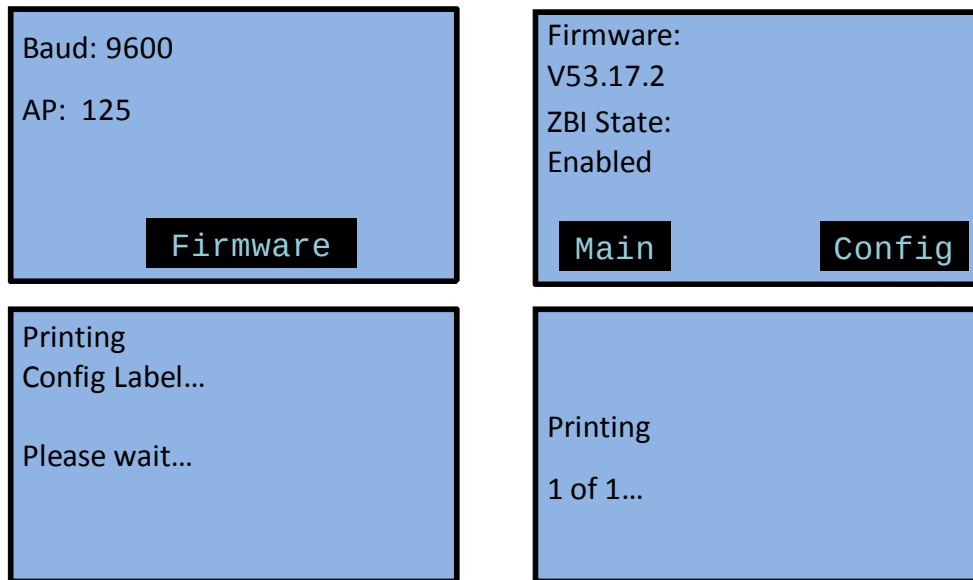
```

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→	<p>Main Config</p> • Defines two links, positioned next to each other - to the "main" and "config" WML cards
26→	<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.

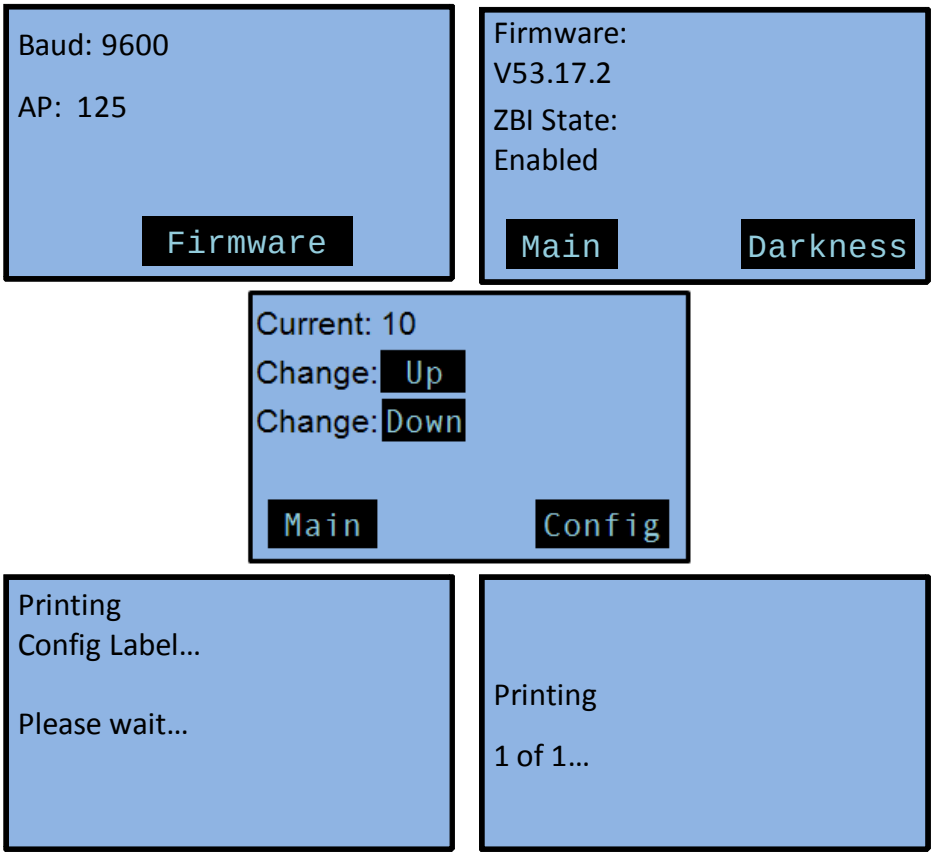
```

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>           <a href="#darkness">Darkness</a></p>
19→  </card>
20→  <card id="darkness" title="" ontimer="#main" alerts="on">
21→    <timer value="50"></timer>
22→    <p>Current: $(print.tone)</p><br/>
23→    <p>Change: </p><do type="accept" label="Up"><setvar name="print.tone"
24→      value="+1.0"/></do><br/>
25→    <p>Change: </p><do type="accept" label="Down"><setvar name="print.tone"
26→      value="-1.0"/></do><br/>
27→    <p> </p><br/>
28→    <p><a href="#main">Main</a>           <a href="#config">Config</a></p>
29→  <card id="config" title="" ontimer="#main" alerts="on">
30→    <timer value="50"></timer>
31→    <p>Printing </p><br/>
32→    <p> Config Label...</p><br/>
33→    <p></p><br/>
34→    <p>Please wait...</p><br/>
35→    <setvar name="file.run" value="e:config.nrd"/>
36→  </card>
37→ </display>
38→ </wml>

```

23→	<p>Change: </p> - Defines the fixed text "Change: " <pre><do type="accept" label="Up"><setvar name="print.tone" value="+1.0"/></do>
</pre> - 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: " <pre>do type="accept" label="Down"><setvar name="print.tone" value="-1.0"/></do>
</pre> - 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 your 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 <code>value=</code> parameter of the <code><do></code> 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 <code>CISDFCRC16</code> 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 <code><data></code> section of the command. See the full manual page on the <code>CISDFCRC16</code> 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").

Glossary



ASCII American Standard Code for Information Interchange. A 7-bit character set that includes Latin characters, as well as control and punctuation characters.

bidirectional text layout The arrangement of characters with mixed directionality on a single line. For example, English and Arabic.

big-endian In the encoding, the most significant byte is stored first.

byte order mark BOM; indicates big-endian (BE) or little-endian (LE).

character The smallest component of a writing system that has understanding, refers to the abstract meaning not the specific shape.

character code Another term for code point.

character set A collection of characters.

character shaping Characters assume different glyphic forms depending on the context. They can be used with a script-based language.

code point An integral reference number assigned to a character.

coded character set An ordered collection of characters that are assigned an integral reference number.

combining semantic cluster Consists of an atomic character, a combining character sequence consisting of a base character plus one or more nonspacing marks, enclosing marks, combining word joiner, such as an Indic virama, or a sequence of Hangul jamos equivalent to a Hangul syllable.

diacritic A mark that is added to a letter or other character to change its value or meaning.

encoding The mapping of a characters code points to a sequence of bytes used to represent the data.

font An electronic collection of glyphs used for the visual representation of characters.

GB18030 A standard required by the People's Republic of China for operating systems of non-handheld computers.

GB 2312 A Chinese character set and encoding.

glyph The visual representation of a character, such as a shape or image. A default glyph character is typically used when the requested glyph is not available in the font.

grapheme cluster Consists of an atomic character, a combining character sequence consisting of a base character plus one or more nonspacing marks or enclosing marks, or a sequence of Hangul jamos equivalent to a Hangul syllable.

ISO 10646 An international standard that defines the Universal Character Set (UCS) and a character encoding. The UCS matches Unicode.

ISO 8859-1 An encoding standard for representing Western European languages using the Latin Alphabet.

language A system of sounds and scripts used to represent and communicate concepts, ideas, meanings, and thoughts.

ligature A glyph representing a combination of two or more characters.

little-endian In the encoding, the least significant byte is stored first.

open type A file format for scalable font files that extends the existing TrueType font file format used by Microsoft Windows and Apple Macintosh operating systems. OpenType tables support advanced layout features.

presentation form A pre-combined character, ligature or variant glyph that has a separate code point; used for compatibility.

script A collection of symbols used to represent textual information in one or more writing systems.

Shift-JIS A shifted encoding of the Japanese character encoding standard, JIS X 0208, heavily deployed in PCs.

True type An outline font format that is scalable without quality loss.

Unicode The universal character set and encoding maintained by the Unicode Consortium.

UTF-8 The Unicode encoding that assigns each character code point to a sequence of one to four bytes.

UTF-16 A Unicode encoding form that represents Unicode code point values in the BMP with 16-bit code units and Unicode code point values in the supplementary planes with pairs of 16-bit code units.

UTF-16BE A Unicode encoding scheme based on UTF-16 that serializes the bytes in each code unit in big-endian order.

UTF-16LE A Unicode encoding scheme based on UTF-16 that serializes the bytes in each code unit in little-endian order.



Notes • _____

Index



Symbols

- ^KC, set client identifier (CID) 426
- ^MW 307
- ^NB, network boot 427
- ^NN, set SNMP parameters 428
- ^NP, set primary/secondary device 429
- ^NT, set SMTP 430
- ^NW, set web authentication timeout value 431
- ^WA, set antenna parameters 432
- ^WE, set wireless encryption values 433
- ^WI, change wireless network settings 435
- ^WL, set LEAP parameters 437
- ^WP, set wireless password 440
- ^WR, set transmit rate 441
- ^WS, set wireless radio card values 443
- ! (EXCLAMATION MARK) ZBI command 464
- ~WR, reset wireless radio card and print server 442

A

- abort download graphic 171
- ACCEPT ZBI command 508
- access password
 - set via ^RQ 401
 - set via ^RZ 417
- access point
 - associated 857
 - MAC address 858
- ADDBREAK ZBI command 479
- adhoc auto mode 855
- adhoc channel 856
- advanced counter
 - reset 331
- AFI byte

- read command (^RA) 390
- write command (^WF) 420
- alphanumeric default font
 - change 148
- antenna parameter setting via ZPL 432
- antenna port
 - select using ZPL 416
- appl.bootblock 630
- appl.name 631
- appl.option_board_version 629
- applicator functions (ZBI)
 - AUXPORT_GETPIN 559
 - AUXPORT_RELEASEPIN 560
 - AUXPORT_SETPIN 558
 - AUXPORT_STEALPIN 556
 - description 555
- applicator reprint 328
- ARP
 - setting using external_wired.ip.arp_interval 721
 - setting using internal_wired.ip.arp_interval 742
- ARP interval or cache time out 721, 742, 793, 865
- array functions
 - description 588
- array functions (ZBI)
 - COLUMNSIZE 594
 - DELROW 592
 - FIND 595
 - INSERTROW 591
 - REDIM 589
 - ROWSIZE 593
- authentication
 - timeout for printer home page 431
- AUTONUM ZBI command 466
- auxiliary port

- set 258, 261
- AUXPORT_GETPIN ZBI command 559
- AUXPORT_RELEASEPIN ZBI command 560
- AUXPORT_SETPIN ZBI command 558
- AUXPORT_STEALPIN ZBI command 556

B

- backfeed sequence
 - change 270, 271, 340
- bar code field default 142
- base types & expressions (ZBI)
 - LET 486
- base types and expressions
 - variable names 482
- base types and expressions (ZBI) 482
 - arrays 483
 - assignment 484
 - boolean expressions 489
 - combined boolean expressions 491
 - constants 483
 - numeric expressions 486
 - string concatenation 488
 - sub-strings 488
 - variable declarations 483
 - variable names 483
- battery
 - set condition 253
- battery status 213
- bitmap font
 - download 163
- bitmapped fonts 948
- bluetooth.address 773
- bluetooth.afh_map 774
- bluetooth.afh_map_curr 775
- bluetooth.afh_mode 776
- bluetooth.authentication 777
- bluetooth.bluetooth_pin 778
- bluetooth.date 779
- bluetooth.discoverable 780
- bluetooth.enable 781
- bluetooth.friendly_name 782
- bluetooth.local_name 783
- bluetooth.radio_auto_baud 784
- bluetooth.radio_version 785
- bluetooth.short_address 786
- bluetooth.version 787
- bootp
 - enable or disable 794
- box 204
- BREAK ZBI command 478

C

- cache on 155
- calibration
 - RFID tag using ZPL (^HR) 386
- calibration, setting dynamic media 377
- cancel all 247
- cancel format 267
- card.inserted 789
- card.mac_addr 788, 895
- caret
 - change 146
- change alphanumeric default font 148
- change backfeed sequence 270, 271, 340
- change caret 146
- change delimiter 147
- change international font 149
- change memory letter designation 153
- change tilde 158
- channel mask
 - setting via ^WS 443
- CHR\$ ZBI command 563
- circle 206
- CISDFCRC16
 - download files 632
- client identifier 727, 749, 801, 872
- client identifier (CID) 426
- CLIENTSOCKET ZBI command 507
- CLOSE ZBI command 503
- CLRERR ZBI command 554
- CODABLOCK 82
 - considerations for ^FD character set 85
 - considerations for the ^BY 84
- code 11 57
- code 128
 - subsets 88
 - subsets A and C 91
 - subsets a, b, and c 86
- code 39 61
- code 49 65
 - automatic mode 68
 - field data character set 68
- code 93 78
 - full ASCII mode 79
- code validation 159
- COLUMNSIZE ZBI command 594
- comm.type 790
- comma separated values (CSV) ZBI commands
 - CSVLOAD 533
 - CSVSTORE 535
 - description 532
 - TXTLOAD 536
 - TXTSTORE 537

- command structure 628, 718, 772
- Command/Function NAME 456
- comment 203
- communications diagnostics 251
 - enable 251
- concatination 198
- configuration
 - update 274
- configuration
 - using ZPL commands 425, 929, 971
- configuration label
 - print 362
- contacts 33
- control and flow commands (ZBI)
 - description 492
 - DO loops 494
 - END 499
 - EXIT 498
 - FOR loops 495
 - GOTO/GOSUB 496
 - IF statements 493
 - SUB 497
- CSVLOAD ZBI command 533, 537
- CSVSTORE ZBI command 535
- CTRL-C ZBI command 472
- current partially input format
 - cancel 276
- currently connected printer
 - set transparent 316
- customer service 33
- cutter.clean_cutter 635

D

- darkness
 - set 333
- data log for RFID 385
- data matrix 137
- DATAREADY ZBI command 504
- date for real time clock
 - set 354
- DATE ZBI command 600
- DATES ZBI command 598
- DEBUG ZBI command 476
- define EPC data structure (^RB) 392
- define kiosk values 284
- define language 280
- define password 283
- define printer name 281
- DELBREAK ZBI command 480
- delete object 240
- delimiter
 - change 147
- DELROW ZBI command 592
- description information
 - display 238
- detect multiple RFID tags (^RN) 400
- device.download_connection_timeout 636
- device.friendly_name 637
- device.frontpanel.key_press 638
- device.frontpanel.line1 639
- device.frontpanel.line2 640
- device.frontpanel.xml 641
- device.jobs_print 642
- device.languages 643
- device.orientation 644
- device.pnp_option 645
- device.reset 646
- device.restore_defaults 647
- device.unique_id 648
- device.uptime 649
- device.user_p1 650
- device.user_p2 651
- device.xml.enable 652
- DHCP
 - CID prefix 725, 747
 - enable 724, 746
- diagnostics
 - disable 252
- diagonal line 207
- directory label
 - print 363
- disable diagnostics 252
- discharge mode
 - battery 278
- display description information 238
- display.text 653
- do
 - device.restore_defaults 647
 - file.dir 655
 - media.cut_now 665
 - media.present.eject 670
- do command 627, 717, 771
- DO loops ZBI command 494
- double signal mode
 - setting 406
- download bitmap font 163
- download encoding 165
- download files 632
- download format 167
- download graphic
 - abort 171
- download graphics 168, 175
- download Intellifont 172
- download true type font 173

download unbounded true type font 174
dynamic media calibration, setting 377

E

E.A.S. bit enable/disable command (^RE) 394
EAN-13 101
EAN-8 74
ECHO ZBI command 468
editing commands (ZBI)
 ! (EXCLAMATION MARK) 464
 AUTONUM ZBI command 466
 description 461
 ECHO ZBI command 468
 LIST ZBI command 465
 NEW 462
 REM 463
 RENUM ZBI command 467
Electronic Product Code (EPC)
 define EPC data structure (^RB) 392
 write EPC data via ^RQ 401
 write EPC data via ^RZ 417
ellipse 208
enable RFID motion (^RM) 399
encoding
 download 165
 select 334
encoding results 414
encryption
 set LEAP parameters 437
 set values via ZPL 433
end format 378
END ZBI command 499
erase download graphics 179
ESSID
 printer 862
 setting via ^WS 443
events (ZBI)
 available events 538
 description 538
 HANDLEEVENT 546
 REGISTEREVENT 543
 TRIGGEREVENT 548
 UNREGISTEREVENT 545
 ZBI Key Names 540
EXIT ZBI command 498
external wired gateway address
 change 728
external wired subnet mask address
 enable or disable 729
external_wired.check 719
external_wired.ip.addr 720
external_wired.ip.arp_interval 721

external_wired.ip.default_addr.enable 722
external_wired.ip.dhcp.cid_all 723, 745
external_wired.ip.dhcp.cid_enable 724
external_wired.ip.dhcp.cid_prefix 725
external_wired.ip.dhcp.cid_suffix 726
external_wired.ip.dhcp.cid_type 727
external_wired.ip.gateway 728
external_wired.ip.netmask 729
external_wired.ip.port 730
external_wired.ip.timeout value 733, 765
external_wired.ip.timeout.value 733, 765
external_wired.ip.v6.addr 734
external_wired.ip.v6.gateway 735
external_wired.ip.v6.prefix_length 736
external_wired.mac_addr 737
external_wired.mac_raw 738
EXTRACT\$ ZBI command 574

F

feedback
 suppress 374
field
 field reverse 196
 orientation 201
 parameter 195
 separator 197
 typeset 198
 variable 200
field block 179, 180
field data 184
field hexadecimal indicator 185
field number 193
field orientation 201
field origin 194
field parameter 195
field reverse print 196
field separator 197
field typeset 198
field variable 200
file system commands (ZBI)
 DELETE ZBI command 531
 description 525
 DIR ZBI command 530
 LOAD ZBI command 529
 STORE ZBI command 528
file.delete 654
file.dir 655
file.run 657
file.type 656
files
 download 632
FIND ZBI command 595

- firmware
 - supported tag types for each version 407
 - ZPL command compatibility 383
- flash memory
 - initialize 248
- font identifier 161
- font linking 187
- font name
 - to call font 53
- fonts
 - font matrices 954
 - standard printer fonts 948
- FOR loops
 - ZBI command 495
- format
 - cancel 267
 - download 167
 - end 378
 - pause 267
 - recall 375
 - set 373
- FTP protocol setting 814
- function rules 456

G

- gateway address
 - change 816
- Gen 2
 - printers and firmware that support Gen 2 407
 - setting password or locking tag 417
 - specify tag type using ZPL 404
 - write EPC data to tag via ^RQ 401
 - write EPC data to tag via ^RZ 417
- getvar
 - appl.bootblock 630
 - appl.name 631
 - appl.option_board_version 629
 - arp_interval 793
 - bluetooth.address 773
 - bluetooth.afh_map 774
 - bluetooth.afh_map_curr 775
 - bluetooth.afh_mode 776
 - bluetooth.authentication 777
 - bluetooth.bluetooth_pin 778
 - bluetooth.date 779
 - bluetooth.discoverable 780
 - bluetooth.enable 781
 - bluetooth.friendly_name 782
 - bluetooth.local_name 783
 - bluetooth.radio_auto_baud 784
 - bluetooth.radio_version 785
 - bluetooth.short_address 786
 - bluetooth.version 787
 - card.inserted 789
 - card.mac_addr 788, 895
 - comm.type 790
 - cutter.clean_cutter 635
 - device.download_connection_timeout 636
 - device.friendly_name 637
 - device.jobs_print 642
 - device.languages 643
 - device.orientation 644
 - device.pnp_option 645
 - device.uptime 649
 - device.user_p1 650
 - device.user_p2 651
 - device.xml.enable 652
 - display.text 653
 - external_wired..ip.arp_interval 721
 - external_wired.check 719
 - external_wired.ip.addr 720
 - external_wired.ip.default_addr.enable address printer 722
 - external_wired.ip.dhcp.cid_all 723, 745
 - external_wired.ip.dhcp.cid_enable 724
 - external_wired.ip.dhcp.cid_prefix 725
 - external_wired.ip.dhcp.cid_suffix 726
 - external_wired.ip.dhcp.cid_type 727
 - external_wired.ip.gateway 728
 - external_wired.ip.netmask 729
 - external_wired.ip.port 730
 - external_wired.ip.timeout.value 733, 765
 - external_wired.ip.v6.addr 734
 - external_wired.ip.v6.gateway 735
 - external_wired.ip.v6.prefix_length 736
 - external_wired.mac_addr 737
 - external_wired.mac_raw 738
 - file.dir 655
 - file.type 656
 - head.latch 658
 - iip.dhcp.lease.last_attempt 803, 873
 - interface.network.active.gateway 659
 - interface.network.active.ip_addr 660
 - interface.network.active.mac_addr 661
 - interface.network.active.mac_raw 662
 - interface.network.active.netmask 663
 - interface.network.active.protocol 664
 - internal_wired..ip.arp_interval 742
 - internal_wired.auto_switchover 739
 - internal_wired.ip.addr 741
 - internal_wired.ip.default_addr.enable address printer 743
 - internal_wired.ip.dhcp.cache_ip 744
 - internal_wired.ip.dhcp.cid_enable 746

internal_wired.ip.dhcp.cid_prefix 747
internal_wired.ip.dhcp.cid_suffix 748
internal_wired.ip.dhcp.cid_type 749
internal_wired.ip.dhcp.lease.last_attempt 750
internal_wired.ip.dhcp.lease.length 751
internal_wired.ip.dhcp.lease.server 752
internal_wired.ip.dhcp.lease.time_left 753
internal_wired.ip.dhcp.option12 754
internal_wired.ip.dhcp.option12_format 755
internal_wired.ip.dhcp.option12_value 756
internal_wired.ip.dhcp.requests_per_session 758
internal_wired.ip.dhcp.request_timeout 757
internal_wired.ip.dhcp.session_interval 759
internal_wired.ip.gateway 760
internal_wired.ip.netmask 761
internal_wired.ip.port 762, 885
internal_wired.ip.protocol 763
internal_wired.ip.timeout.enable 732, 764
internal_wired.mac_addr 766
internal_wired.mac_raw 767
ip.active_network 791
ip.addr 792
ip.bootp.enable 794
ip.dhcp.cache_ip 795
ip.dhcp.cid_all 796
ip.dhcp.cid_enable 797
ip.dhcp.cid_prefix 798
ip.dhcp.cid_suffix 799
ip.dhcp.cid_type 801
ip.dhcp.enable 802
ip.dhcp.lease.length 800
ip.dhcp.lease.server 804
ip.dhcp.lease.time_left 805
ip.dhcp.option12 806
ip.dhcp.option12_format 807
ip.dhcp.option12_value 808
ip.dhcp.requests_per_session 810
ip.dhcp.request_timeout 809
ip.dhcp.session_interval 811
ip.dns.domain 812
ip.dns.servers 813
ip.ftp.enable 814
ip.ftp.execute_file 815
ip.gateway 816
ip.http.enable 817
ip.lpd.enable 818
ip.mac_raw 819
ip.mirror.auto 820
ip.mirror.error_retry 821
ip.mirror.feedback.auto 822
ip.mirror.feedback.freq 823
ip.mirror.feedback.odometer 824
ip.mirror.feedback.path 825
ip.mirror.fetch 826
ip.mirror.freq 827
ip.mirror.freq_hours 828
ip.mirror.last_error 829
ip.mirror.last_time 830
ip.mirror.password 831
ip.mirror.server 833
ip.mirror.success 834
ip.mirror.success_time 835
ip.mirror.username 836
ip.mirror.version 837
ip.netmask 838
ip.pop3.enable 839
ip.pop3.password 840
ip.pop3.poll 841
ip.pop3.server_addr 842
ip.pop3.username 843
ip.port 844
ip.primary_network 845
ip.smtp.domain 846
ip.smtp.enable 847
ip.smtp.server_addr 848
ip.snmp.enable 851
ip.snmp.get_community_name 849
ip.snmp.set_community_name 850
ip.tcp.enable 680, 853
ip.telnet.enable 852
ip.udp.enable 854
media.media_low.external 667
media.media_low.warning 668
media.present.cut_amount 669
media.present.cut_margin 674
media.present.length_addition 671
media.present.loop_length 672
media.present.loop_length_max 673
media.present.present_timeout 675
media.present.present_type 676
media.printmode 677
media.speed 679
odometer.headclean 680
odometer.headnew 681
odometer.media_marker_count 683
odometer.media_marker_count1 683
odometer.media_marker_count2 684
odometer.retracts_count 685
odometer.total_print_length 686
print.tone 687
wlan.adhocautomode 855
wlan.adhocchannel 856
wlan.associated 857
wlan.bssid 858

- wlan.channel 859
 - wlan.channel_mask 860
 - wlan.current_tx.rate 861
 - wlan.essid 862
 - wlan.firmware_version 863
 - wlan.ip.addr 864
 - wlan.ip.arp_interval 865
 - wlan.ip.default_addr.enable address printer 866
 - wlan.ip.dhcp.cache_ip 867
 - wlan.ip.dhcp.cid_all 868
 - wlan.ip.dhcp.cid_enable 869
 - wlan.ip.dhcp.cid_prefix 870
 - wlan.ip.dhcp.cid_suffix 871
 - wlan.ip.dhcp.cid_type 872
 - wlan.ip.dhcp.lease.length 874
 - wlan.ip.dhcp.lease.server 875
 - wlan.ip.dhcp.lease.time_left 876
 - wlan.ip.dhcp.option12 877
 - wlan.ip.dhcp.option12_format 878
 - wlan.ip.dhcp.option12_value 879
 - wlan.ip.dhcp.requests_per_session 881
 - wlan.ip.dhcp.request_timeout 880
 - wlan.ip.dhcp.session_interval 882
 - wlan.ip.gateway 883
 - wlan.ip.netmask 884
 - wlan.ip.protocol 886
 - wlan.ip.timeout.enable 887
 - wlan.ip.timeout.value 888
 - wlan.keep_alive.enable 889
 - wlan.keep_alive.timeout 890
 - wlan.kerberos.kdc 891
 - wlan.kerberos.password 892
 - wlan.kerberos.realm 893
 - wlan.kerberos.username 894
 - wlan.mac_raw 896
 - wlan.operating_mode 897
 - wlan.password 898
 - wlan.preamble 899
 - wlan.private_key_password 900
 - wlan.roam.interchannel_delay 901
 - wlan.roam.interval 902
 - wlan.roam.signal 904
 - wlan.security 905
 - wlan.signal_noise 914
 - wlan.signal_quality 915
 - wlan.signal_strength 916
 - wlan.station_name 917
 - wlan.tx_power 918
 - wlan.tx_rate 919
 - wlan.username 920
 - wlan.wep.auth_type 921
 - wlan.wep.index 922
 - wlan.wep.key1 923
 - wlan.wep.key2 924
 - wlan.wep.key3 925
 - wlan.wep.key4 926
 - wlan.wep.key_format 928
 - wlan.wpa.psk 927
 - wlanroam.max_chan_scan_time 903
 - getvar command 626, 716, 770
 - GETVAR ZBI command 604
 - GOTO/GOSUB ZBI command 496
 - graphic
 - box 204
 - circle 206
 - diagonal line 207
 - ellipse 208
 - field 209
 - recall 376
 - symbol 211
 - graphic field 209
 - graphics
 - download 168, 175
 - erase download 179
 - upload 237
 - graphing sensor calibration 254
- ## H
- HANDLEEVENT ZBI command 546
 - head test
 - fatal 265
 - interval 273
 - non-fatal 266
 - head test fatal 265
 - head test interval 273
 - head test non-fatal 266
 - head.latch 658
 - HEXTOINT ZBI command 587
 - host
 - directory list 235
 - graphic 216
 - identification 218
 - RAM status 219
 - status return 228
 - host directory list 235
 - host graphic 216
 - host identification 218
 - host linked font list 232
 - host query 220, 365
 - host RAM status 219
 - host status return 228
 - host verification command (^HV) 234
 - ~HQ Host Query 365

HTTP protocol 817

I

IF statements (ZBI) 493

image

load 242

move 244

save 245

image load 242

image move 244

image save 245

INBYTE ZBI command 514

industrial 2 of 5 106

initialize Flash memory 248

input and output commands (ZBI)

ACCEPT 508

CLIENTSOCKET 507

CLOSE 503

DATAREADY 504

description 500

INBYTE 514

INPUT 510

OPEN 502

OUTBYTE 513

PRINT 512

READ 515

SEARCHTO\$ 517

SERVERCLOSE 506

SERVERSOCKET 505

WRITE 516

INPUT ZBI command 510

INSERTROW ZBI command 591

interface.network.active.gateway 659

interface.network.active.ip_addr 660

interface.network.active.mac_addr 661

interface.network.active.mac_raw 662

interface.network.active.netmask 663

interface.network.active.protocol 664

interleaved

2 of 5 59

internal wired gateway address

change 760

internal wired subnet mask address

enable or disable 761

internal wired timeout enable 732, 764

internal wired.ip.timeout enable 732, 764

internal.wired.mac_addr 766

internal_wired.auto_switchover 739

internal_wired.ip.addr 741

internal_wired.ip.arp_interval 742

internal_wired.ip.default_addr.enable 743

internal_wired.ip.dhcp.cache_ip 744

internal_wired.ip.dhcp.cid_enable 746

internal_wired.ip.dhcp.cid_prefix 747

internal_wired.ip.dhcp.cid_suffix 748

internal_wired.ip.dhcp.cid_type 749

internal_wired.ip.dhcp.lease.last_attempt 750

internal_wired.ip.dhcp.lease.length 751

internal_wired.ip.dhcp.lease.server 752

internal_wired.ip.dhcp.lease.time_left 753

internal_wired.ip.dhcp.option12 754

internal_wired.ip.dhcp.option12_format 755

internal_wired.ip.dhcp.option12_value 756

internal_wired.ip.dhcp.requests_per_session 758

internal_wired.ip.dhcp.request_timeout 757

internal_wired.ip.dhcp.session_interval 759

internal_wired.ip.gateway 760

internal_wired.ip.netmask 761

internal_wired.ip.port 762

internal_wired.ip.protocol 763

internal_wired.ip.timeout.enable 732, 764

internal_wired.mac_addr 766

internal_wired.mac_raw 767

international font

change 149

international mode

setting via ^WS 443

Introduction to Zebra Basic Interpreter (ZBI) 458

INTTOHEX\$ ZBI command 586

IP address

printer 720, 741, 792, 864

setting via ip.addr 845

ip.active_network 791

ip.addr 792

ip.arp_interval 793

ip.bootp.enable 794

ip.dhcp.cache_ip 795

ip.dhcp.cid_all 723, 745, 796

ip.dhcp.cid_enable 797

ip.dhcp.cid_prefix 798

ip.dhcp.cid_suffix 799

ip.dhcp.cid_type 801

ip.dhcp.enable 802

ip.dhcp.lease.last_attempt 803, 873

ip.dhcp.lease.length 800

ip.dhcp.lease.server 804

ip.dhcp.lease.time_left 805

ip.dhcp.option12 806

ip.dhcp.option12_format 807

ip.dhcp.option12_value 808

ip.dhcp.requests_per_session 810

ip.dhcp.request_timeout 809

ip.dhcp.session_interval 811

ip.dns.domain 812

- ip.dns.servers 813
- ip.ftp.enable 814
- ip.ftp.execute_file 815
- ip.gateway 728, 816
- ip.http.enable 817
- ip.lpd.enable 818
- ip.mac_raw 819
- ip.mirror.auto 820
- ip.mirror.error_retry 821
- ip.mirror.feedback.auto 822
- ip.mirror.feedback.freq 823
- ip.mirror.feedback.odometer 824
- ip.mirror.feedback.path 825
- ip.mirror.fetch 826
- ip.mirror.freq 827
- ip.mirror.freq_hours 828
- ip.mirror.last_error 829
- ip.mirror.last_time 830
- ip.mirror.password 831
- ip.mirror.server 833
- ip.mirror.success 834
- ip.mirror.success_time 835
- ip.mirror.username 836
- ip.mirror.version 837
- ip.netmask 838
- ip.pop3.enable 839
- ip.pop3.password 840
- ip.pop3.poll 841
- ip.pop3.server_addr 842
- ip.pop3.username 843
- ip.port 730, 762, 844, 885
- ip.primary_network 845
- ip.smtp.domain 846
- ip.smtp.enable 847
- ip.smtp.server_addr 848
- ip.snmp.get_community_name 849
- ip.snmp.server_addr 851
- ip.snmp.set_community_name 849, 850
- ip.tcp.enable 853
- ip.telnet.enable 852
- ip.udp.enable 854
- ISERRORd ZBI command 550

K

- Kerberos
 - password 892
 - realm 893
 - username 894
- key distribution center (KDC) 891
- key rotation
 - firmware version required 451, 910
- kill battery 278

- kill password 417
 - set via ^RQ 401
 - set via ^RZ 417
- kiosk values 284

L

- label
 - maximum length 298
 - reverse print 289
 - shift 290
 - top 291
- label home 286, 287
- label length 288
 - set 263
- language
 - define 280
- LCASE\$ ZBI command 562
- LEAP mode
 - setting using ZPL 437
- LEN ZBI command 578
- LET
 - ZBI command 486
- liability 2
- linked font
 - host list 232
- LIST ZBI command 465
- lock RFID tag
 - Gen 2 417
 - not Gen 2 395
- log file for RFID 385
- LOGMARS 112
- LPD protocol setting 818
- LTRIM\$ ZBI command 564

M

- ^MA Set Maintenance Alerts 292
- MAC address
 - access point 858
 - retrieve via external_wired.mac_addr 737
 - retrieve via internal_wired.mac_addr 766
- maintenance alerts, setting 292
- map clear 294
- math functions
 - description 579
- math functions (ZBI)
 - HEXTOINT 587
 - INTTOHEX\$ 586
 - MAX 581
 - MAXNUM 583
 - MIN 582
 - MOD 584

- STR\$ 580
- VAL 585
- MAX ZBI command 581
- maximum label length 298
- MAXNUM ZBI command 583
- media
 - darkness 295
 - feed 296
 - ordering 33
 - tracking 301
 - type 304
- media calibration, setting dynamic 377
- media darkness 295
- media sensor
 - set 352
- media sensor calibration 250
 - set 250
- media tracking 301
- media type 304
- media.cut_now 665
- media.darkness_mode 666
- media.media_low.external 667
- media.media_low.warning 668
- media.present.cut_amount 669
- media.present.cut_margin 674
- media.present.eject 670
- media.present.length_addition 671
- media.present.loop_length 672
- media.present.loop_length_max 673
- media.present.present_timeout 675
- media.present.present_type 676
- media.printmode 677
- media.speed 679
- memory letter designation
 - change 153
- ^MI Set Maintenance Information Message 297
- MIN ZBI command 582
- mirror image
 - printing 321
- MOD ZBI command 584
- mode protection 302
- modify head warning 307
- motion in RFID label 399
- MSI 114
- multiple field origin locations 190

N

- network
 - connect 309
 - ID number 312
- network boot command 427
- network configuration label
 - print 438
- network connect 309
- network ID number 312
- network operating mode 897
- network printers
 - set all transparent 313
- network settings
 - setting via ZPL 435
- NEW ZBI command 462
- number of retries for block (^RR) 403

O

- object delete 240
- odometer
 - ~WQ 365
- odometer related commands
 - ^JH 255
 - ^MA 292
 - ^MI 297
 - ~HQ 220
 - ~RO 331
- odometer.headclean 680
- odometer.headnew 681
- odometer.label_dot_length 682
- odometer.media_marker_count1 683
- odometer.media_marker_count2 684
- odometer.total_print_length 686
- odometer.headclean 680
- odometer.headnew 681
- odometer.label_dot_length 682
- odometer.media_marker_count 683, 684
- odometer.media_marker_count1 683
- odometer.media_marker_count2 684
- odometer.retracts_count 685
- odometer.total_print_length 686
- offset for real time clock
 - set 346
- ON ERROR ZBI command 555
- OPEN ZBI command 502
- option 61 727, 749, 801, 872
 - CID prefix 725, 747
 - enable 724, 746
 - external_wired.ip.dhcp.cid_enable 724
 - internal_wired.ip.dhcp.cid_enable 746
 - ip.dhcp.cid_enable 797
 - ip.dhcp.cid_prefix 725, 747, 798, 870
 - wlan.ip.dhcp.cid_enable 869
- optional memory
 - reset 249
- ordering ribbon and media 33
- OUTBYTE ZBI command 513

P

- password
 - define 283
 - set wireless password via ^WP 440
- password for RFID tag 417
- pause
 - programmable 324
- pause format 267
- PDF417 70
 - consideration for ^FD 73
- permanently lock Gen 2 tag data 417
- POP3
 - mailbox 839
 - mailbox password 840
 - poll interval 841
 - username 843
- POP3 server
 - IP address 842
- POS ZBI command 577
- POSTNET 144
- power on
 - reset 269
- preamble length
 - radio 899
- pre-shared key value 927
- primary network device
 - selection by ^NC 308
- primary/secondary device
 - setting via ZPL 429
- print
 - start 329
 - width 330
- print mode 299
 - selection via ZPL 299
- print orientation 323
- print quantity 325
- print rate 326
- print start 329
- print width 330
- PRINT ZBI command 512
- print.tone 687
- printer
 - sleep 379
- printer IP address 720, 741, 792, 864
- printer name
 - define 281
- printer sleep 379
- Printers, ZBI Keys, & ZBI Versions 458
- printhead resistance
 - set 351
- printing mirror image of label 321
- private key password 900

- programmable pause 324

Q

- QR code
 - normal mode 123
- quantity
 - print 325

R

- read power
 - change using ZPL 415
- read RFID tag
 - read or write RFID format (^RF) 395
 - read RFID tag (^RT) 411
- READ ZBI command 515
- real time clock
 - set language 341
 - set mode 341
- real time clock date format
 - select 279
- real time clock time format
 - select 279
- recall format 375
- recall graphic 376
- REDIM ZBI command 589
- REGISTEREVENT ZBI command 543
- REM ZBI command 463
- RENUM ZBI command 467
- REPEAT\$ ZBI command 565
- report RFID encoding results 414
- reprint
 - after error 277
 - applicator 328
- reset
 - power on 269
- reset advanced counter 331
- reset optional memory 249
- reset wireless radio card and print server 442
- RESTART ZBI command 473
- return data to host computer (^HV) 234
- return RFID data log to host 385
- RFID
 - calibrate RFID tag using ZPL (^HR) 386
 - change read power using ZPL 415
 - change write power using ZPL 415
 - detect multiple tags (^RN) 400
 - enable motion (^RM) 399
 - number of retries for block (^RR) 403
 - RFID setup command (^RS) 404
 - verify write operation (^WV) 424
- RFID mode

- selection via ZPL 299
- RFID programming position
 - setting via ZPL 405
- RFID tag ID command (^RI) 398
- ribbon
 - ordering 33
- ribbon tension
 - set 275
- ROWSIZE ZBI command 593
- RTRIM\$ ZBI command 566
- RUN ZBI command 471
- running and debugging commands (ZBI)
 - ADDBREAK ZBI command 479
 - BREAK ZBI command 478
 - CTRL-C ZBI command 472
 - DEBUG ZBI command 476
 - DELBREAK ZBI command 480
 - description 469
 - RESTART ZBI command 473
 - RUN ZBI command 471
 - STEP ZBI command 475
 - TRACE ZBI command 477
 - ZPL ZBI command 481

S

- sales 33
- scalable font 51, 948
 - download Intellifont 172
- SEARCHTO\$ ZBI command 517
- select encoding 334
- sensor calibration
 - graphing 254
- serial communications
 - set 332
- serialization data 343
- serialization field
 - standard ^FD string 335
- SERVERCLOSE ZBI command 506
- SERVERSOCKET ZBI command 505
- set all network printers transparent 313
- set auxiliary port 261
- set battery condition 253
- set darkness 333
- set dots
 - millimeter 264
- set dots per millimeter 264
- set dynamic media calibration 377
- set label length 263
- set maintenance alerts 292
- set maintenance information message 297
- set RFID tag or kill password (^RZ) 417
- set serial communications 332
- set units of measurements 305
- set up RFID information (^RS) 404
- Set/Get/Do interactions
 - description 602
- Set/Get/Do interactions (ZBI)
 - GETVAR 604
 - SETVAR 603
- SETERR ZBI command 553
- setvar
 - bluetooth.afh_map 774
 - bluetooth.afh_mode 776
 - bluetooth.authentication 777
 - bluetooth.bluetooth_pin 778
 - bluetooth.discoverable 780
 - bluetooth.enable 781
 - bluetooth.friendly_name 782
 - bluetooth.radio_auto_baud 784
 - comm.type 790
 - cutter.clean_cutter 635
 - device.download_connection_timeout 636
 - device.friendly_name 637
 - device.orientation 644
 - device.pnp_option 645
 - device.reset 646
 - device.uptime 649
 - device.user_p1 650
 - device.user_p2 651
 - device.xml.enable 652
 - external_wired.check 719
 - external_wired.ip.addr 720, 722
 - external_wired.ip.arp_interval 721
 - external_wired.ip.dhcp.cid_all 723, 745
 - external_wired.ip.dhcp.cid_enable 724
 - external_wired.ip.dhcp.cid_prefix 725
 - external_wired.ip.dhcp.cid_suffix 726
 - external_wired.ip.dhcp.cid_type 727
 - external_wired.ip.gateway 728
 - external_wired.ip.netmask 729
 - external_wired.ip.port 730
 - external_wired.ip.timeout value 733, 765
 - file.dir 655
 - file.run 657
 - file.type 656
 - head.latch 658
 - internal_wired.auto_switchover 739
 - internal_wired.ip.addr 741, 743
 - internal_wired.ip.arp_interval 742
 - internal_wired.ip.dhcp.cid_enable 746
 - internal_wired.ip.dhcp.cid_prefix 747
 - internal_wired.ip.dhcp.cid_suffix 748
 - internal_wired.ip.dhcp.cid_type 749
 - internal_wired.ip.dhcp.option12 754

- internal_wired.ip.dhcp.option12_format 755
- internal_wired.ip.dhcp.option12_value 756
- internal_wired.ip.dhcp.requests_per_session 758
- internal_wired.ip.dhcp.request_timeout 757
- internal_wired.ip.dhcp.session_interval 759
- internal_wired.ip.gateway 760
- internal_wired.ip.netmask 761
- internal_wired.ip.port 762, 885
- internal_wired.ip.protocol 763
- internal_wired.ip.timeout enable 732, 764
- ip.active_network 791
- ip.addr 792
- ip.arp_interval 793
- ip.bootp.enable 794
- ip.dhcp.cid_all 796
- ip.dhcp.cid_enable 797
- ip.dhcp.cid_prefix 798
- ip.dhcp.cid_suffix 799
- ip.dhcp.cid_type 801
- ip.dhcp.enable 802
- ip.dhcp.option12 806
- ip.dhcp.option12_format 807
- ip.dhcp.option12_value 808
- ip.dhcp.requests_per_session 810
- ip.dhcp.request_timeout 809
- ip.dhcp.session_interval 811
- ip.dns.domain 812
- ip.dns.servers 813
- ip.ftp.enable 814
- ip.ftp.execute_file 815
- ip.gateway 816
- ip.http.enable 817
- ip.lpd.enable 818
- ip.mirror.auto 820
- ip.mirror.error_retry 821
- ip.mirror.feedback.auto 822
- ip.mirror.feedback.freq 823
- ip.mirror.feedback.odometer 824
- ip.mirror.feedback.path 825
- ip.mirror.fetch 826
- ip.mirror.freq 827
- ip.mirror.freq_hours 828
- ip.mirror.password 831
- ip.mirror.server 833
- ip.mirror.success 834
- ip.mirror.success_time 835
- ip.mirror.username 836
- ip.netmask 838
- ip.pop3.enable 839
- ip.pop3.password 840
- ip.pop3.poll 841
- ip.pop3.server_addr 842
- ip.pop3.username 843
- ip.port 844
- ip.primary_network 845
- ip.smtp.domain 846
- ip.smtp.enable 847
- ip.smtp.server_addr 848
- ip.snmp.enable 851
- ip.snmp.get_community_name 849
- ip.snmp.set_community_name 850
- ip.tcp.enable 853
- ip.telnet.enable 852
- ip.udp.enable 854
- media.cut_now 665
- media.darkness_mode 666
- media.media_low.warning 668
- media.present.cut_amount 669
- media.present.cut_margin 674
- media.present.eject 670
- media.present.length_addition 671
- media.present.loop_length 672
- media.present.loop_length_max 673
- media.present.present_timeout 675
- media.present.present_type 676
- media.printmode 677
- media.speed 679
- odometer.headclean 680
- odometer.headnew 681
- odometer.media_marker_count 683
- odometer.media_marker_count1 683
- odometer.media_marker_count2 684
- odometer.retracts_count 685
- print.tone 687
- wlan.adhocautomode 855
- wlan.adhocchannel 856
- wlan.channel_mask 860
- wlan.essid 862
- wlan.ip.addr 864, 866
- wlan.ip.arp_interval 865
- wlan.ip.dhcp.cid_all 868
- wlan.ip.dhcp.cid_enable 869
- wlan.ip.dhcp.cid_prefix 870
- wlan.ip.dhcp.cid_suffix 871
- wlan.ip.dhcp.cid_type 872
- wlan.ip.dhcp.option12 877
- wlan.ip.dhcp.option12_format 878
- wlan.ip.dhcp.option12_value 879
- wlan.ip.dhcp.requests_per_session 881
- wlan.ip.dhcp.request_timeout 880
- wlan.ip.dhcp.session_interval 882
- wlan.ip.gateway 883
- wlan.ip.netmask 884
- wlan.ip.protocol 886

- wlan.ip.timeout enable 887
 - wlan.ip.timeout value 888
 - wlan.keep_alive.enable 889
 - wlan.keep_alive.timeout 890
 - wlan.kerberos.kdc 891
 - wlan.kerberos.password 892
 - wlan.kerberos.realm 893
 - wlan.kerberos.username 894
 - wlan.operating_mode 897
 - wlan.password 898
 - wlan.preamble 899
 - wlan.private_key_password 900
 - wlan.roam.interchannel_delay 901
 - wlan.roam.interval 902
 - wlan.roam.max_chan_scan_time 903
 - wlan.roam.signal 904
 - wlan.security 905
 - wlan.station_name 917
 - wlan.tx_power 918
 - wlan.tx_rate 919
 - wlan.username 920
 - wlan.wep.auth_type 921
 - wlan.wep.index 922
 - wlan.wep.key1 923
 - wlan.wep.key2 924
 - wlan.wep.key3 925
 - wlan.wep.key4 926
 - wlan.wep.key_format 928
 - wlan.wpa.psk 927
 - setvar command 626, 716, 770
 - SETVAR ZBI command 603
 - SGD command structure 628, 718, 772
 - SGTIN-64 standard programming example 393
 - single signal mode
 - setting 406
 - SLEEP ZBI command 552
 - slew
 - home position 319
 - slew given number
 - dot rows 318
 - slew to home position 319
 - SMTP IP address 848
 - SMTP parameters
 - setting via ZPL 430
 - SMTP protocol 846, 847
 - SNMP
 - get community 849
 - set community 850
 - SNMP parameters
 - setting via ZPL 428
 - SNMP protocol 851
 - SNMP query 849
 - specify number of retries for block (^RR) 403
 - SPLIT ZBI command 567
 - SPLITCOUNT ZBI command 572
 - standard printer fonts 948
 - start print 347
 - start ZBI 258
 - STEP ZBI command 475
 - STR\$ ZBI command 580
 - string functions (ZBI)
 - CHR\$ 563
 - description 561
 - EXTRACT\$ 574
 - LCASE\$ 562
 - LEN 578
 - LTRIM\$ 564
 - POS 577
 - REPEAT\$ 565
 - RTRIM\$ 566
 - SPLIT 567
 - SPLITCOUNT 572
 - UCASE\$ 573
 - SUB ZBI command 497
 - subnet mask
 - change 729, 761
 - subnet mask address
 - enable or disable 838
 - supported
 - printers 458
 - symbol 211, 215
 - systems (ZBI)
 - CLRERR 554
 - description 549
 - ISERRORd 550
 - ON ERROR 555
 - SETERR 553
 - SLEEP 552
 - TRIGGEREVENT 551
- ## T
- tag type
 - specify using ZPL 404
 - TCP communication 730, 762, 844, 885
 - TCP socket protocol 853
 - tear-off adjust position 358
 - technical support 33
 - TELNET protocol 852
 - terminate ZBI 268
 - text blocks 359
 - text properties
 - advanced 317
 - tilde
 - change 158

- time and date functions
 - description 597
- time and date functions (ZBI)
 - DATE 600
 - DATES\$ 598
 - TIME 601
 - TIMES\$ 599
- time for real time clock
 - set 354
- TIME ZBI command 601
- TIMES\$ ZBI command 599
- timeout value 733, 765
- timeout value for printer home page 431
- TRACE ZBI command 477
- transfer object 360
- transmit rate
 - setting via ^WR 441
- transponders
 - calibration using ZPL (^HR) 386
- TRIGGEREVENT ZBI command 548, 551
- true type font
 - download 173
- TXTLOAD ZBI command 536

U

- UCASE\$ ZBI command 573
- UDP communication 730, 762, 844, 885
- UDP socket protocol 854
- unbounded true type font
 - download 174
- units of measurement
 - set 305
- UNREGISTEREVENT ZBI command 545
- UPC/EAN extensions 130
- UPC-A 135
- UPC-E 76
- update configuration 274
- upload graphics 237
- UPS maxicode 98
 - considerations for ^FD 99
- use font name to call font 53

V

- VAL ZBI command 585
- verify RFID write operation (^WV) 424

W

- web authentication timeout value 431
- web pages
 - set timeout value 431
- web server setting 817

- WEP key format 928
- WEP mode
 - setting encryption index via ^WX 446
- width
 - print 330
- wired equivalent privacy 922
- wired print server
 - ^NB to set check for 427
- wireless password
 - setting via ^WP 440
- Wireless Print Server
 - ZPL commands 929, 971
- wireless print server
 - change network settings 435
- wireless pulse and interval
 - setting via ^WS 443
- wireless radio card
 - reset via ZPL 442
 - setting values via ^WS 443
- wireless timeout enable 887
- wlan gateway address
 - change 883
- wlan subnet mask address
 - enable or disable 884
- wlan.adhocautomode 855
- wlan.adhocchannel 856
- wlan.associated 857
- wlan.bssid 858
- wlan.channel 859
- wlan.channel_mask 860
- wlan.current_tx_rate 861
- wlan.essid 862
- wlan.firmware_version 863
- wlan.ip.addr 864
- wlan.ip.arp_interval 865
- wlan.ip.default_addr.enable 866
- wlan.ip.dhcp.cache_ip 867
- wlan.ip.dhcp.cid_all 868
- wlan.ip.dhcp.cid_enable 869
- wlan.ip.dhcp.cid_prefix 870
- wlan.ip.dhcp.cid_suffix 871
- wlan.ip.dhcp.cid_type 872
- wlan.ip.dhcp.lease.length 874
- wlan.ip.dhcp.lease.server 875
- wlan.ip.dhcp.lease.time_left 876
- wlan.ip.dhcp.option12 877
- wlan.ip.dhcp.option12_format 878
- wlan.ip.dhcp.option12_value 879
- wlan.ip.dhcp.requests_per_session 881
- wlan.ip.dhcp.request_timeout 880
- wlan.ip.dhcp.session_interval 882
- wlan.ip.gateway 883

- wlan.ip.netmask 884
- wlan.ip.protocol 886
- wlan.ip.timeout enable 887
- wlan.ip.timeout value 888
- wlan.ip.timeout.enable 887
- wlan.keep_alive.enable 889
- wlan.keep_alive.timeout 890
- wlan.kerberos.kdc 891
- wlan.kerberos.password 892
- wlan.kerberos.realm 893
- wlan.kerberos.username 894
- wlan.mac_raw 896
- wlan.operating_mode 897
- wlan.password 898
- wlan.preamble 899
- wlan.private_key_password 900
- wlan.roam.interchannel_delay 901
- wlan.roam.interval 902
- wlan.roam.max_chan_scan_time 903
- wlan.roam.signal 904
- wlan.security 905
- wlan.signal_noise 914
- wlan.signal_quality 915
- wlan.signal_strength 916
- wlan.station_name 917
- wlan.tx_power 918
- wlan.tx_rate 919
- wlan.username 920
- wlan.wep.auth_type 921
- wlan.wep.index 922
- wlan.wep.key1 923
- wlan.wep.key2 924
- wlan.wep.key3 925
- wlan.wep.key4 926
- wlan.wep.key_format 928
- wlan.wpa.psk 927
- write AFI byte command (^WF) 420
- write data to tag
 - via ^RQ 401
 - via ^RZ 417
- write power
 - change using ZPL 415
- write RFID format (^RF) 395
- write RFID tag (^WT) 422
- WRITE ZBI command 516
- Writing ZBI Programs 460

X

- ^XS Set Dynamic Calibration Defaults 377

Z

ZBI

- Command/Function Name description 456
- function rules 456
- start 258
- terminate 268

- ZBI character set 993

ZBI commands

- ! (EXCLAMATION MARK) 464
- ACCEPT 508
- ADDBREAK 479
- AUTONUM 466
- AUXPORT_GETPIN 559
- AUXPORT_RELEASEPIN 560
- AUXPORT_SETPIN 558
- AUXPORT_STEALPIN 556
- BREAK 478
- CHR\$ 563
- CLIENTSOCKET 507
- CLOSE 503
- CLRERR 554
- COLUMNSIZE 594
- CSVLOAD 533
- CSVSTORE 535
- CTRL-C 472
- DATAREADY 504
- DATE 600
- DATES\$ 598
- DEBUG 476
- DELBREAK 480
- DELETE 531
- DELROW 592
- DIR 530
- DO loops 494
- ECHO 468
- END 499
- EXIT 498
- EXTRACT\$ 574
- FIND 595
- FOR loops 495
- GETVAR 604
- GOTO/GOSUB 496
- HANDLEEVENT 546
- HEXTPOINT 587
- IF statements 493
- INBYTE 514
- INPUT 510
- INSERTROW 591
- INTTOHEX\$ 586
- ISERRORd 550
- LCASE\$ 562
- LEN 578

- LET 486
- LIST 465
- LOAD 529
- LTRIM\$ 564
- MAX 581
- MAXNUM 583
- MIN 582
- MOD 584
- NEW 462
- ON ERROR 555
- OPEN 502
- OUTBYTE 513
- POS 577
- PRINT 512
- READ 515
- REDIM 589
- REGISTEREVENT 543
- REM 463
- RENUM 467
- REPEAT\$ 565
- RESTART 473
- ROWSIZE 593
- RTRIM\$ 566
- RUN 471
- SEARCHTO\$ 517
- SERVERCLOSE 506
- SERVERSOCKET 505
- SETERR 553
- SETVAR 603
- SLEEP 552
- SPLIT 567
- SPLITCOUNT 572
- STEP 475
- STORE 528
- STR\$ 580
- SUB 497
- TIME 601
- TIMES\$ 599
- TRACE 477
- TRIGGEREVENT 548, 551
- TXTLOAD 536
- TXTSTORE 537
- UCASE\$ 573
- UNREGISTEREVENT 545
- VAL 585
- WRITE 516
- ZBI Key Names 540
- ZPL 481
- ZBI keys 458
- ZBI printers 458
- ZBI versions 458
 - 1.0 through 1.5 458
 - 2.0 and higher 459
- zbi.control.add_breakpoint 694
- zbi.control.break 695
- zbi.control.clear_breakpoints 696
- zbi.control.delete_breakpoint 697
- zbi.control.line_number 698
- zbi.control.restart 699
- zbi.control.run 700
- zbi.control.step 701
- zbi.control.terminate 702
- zbi.control.variable_name 703
- zbi.control.variable_value 704
- zbi.key 705
- zbi.last_error 706
- zbi.reseller_key 707
- zbi.revision 708
- zbi.running_program_name 709
- zbi.start_info.execute 710
- zbi.start_info.file_name 711
- zbi.start_info.memory_alloc 712
- zbi.state 713
- Zebra Programming Language (ZPL II)
 - ^HL or ~HL, Return RFID Data Log to Host (^HL or ~HL) 385
 - ^HR, Calibrate RFID Transponder 386
 - ^RA, Read AFI Byte 390
 - ^RB, Define EPC Data Structure 392
 - ^RE, Enable/Disable E.A.S. Bit 394
 - ^RF, Read or Write RFID Format 395
 - ^RI, Get RFID Tag ID 398
 - ^RM, Enable RFID Motion 399
 - ^RN, Detect Multiple RFID Tags 400
 - ^RQ, Quick Write EPC Data and Passwords 401
 - ^RR, Specify RFID Retries for a Block 403
 - ^RS, RFID Setup 404
 - ^RT, Read RFID Tag 411
 - ^RW, Set RFID Read and Write Power 415
 - ^RZ, Set RFID Tag Password and Lock Tag 417
 - ^WF, Write AFI Byte 420
 - ^WT, Write Tag 422
 - ^WV, Verify RFID Write Operation 424
 - ~RV, Report RFID Encoding Results 414
 - command compatibility with printers and firmware 383
- ZebraNet Alert
 - halt 349
 - set 355
- ZPL
 - set 357
- ZPL commands 425
 - ^B7 70
 - ~HQ 255, 365

- ^MA 292
- ^MI 297
- Wireless Print Server commands 929, 971
- ^XS 377
- ZPL ZBI command 481

**Zebra Technologies Corporation**

Zebra Technologies Corporation
475 Half Day Road, Suite 500
Lincolnshire, IL 60069 USA
T: +1 847 634 6700
Toll-free +1 866 230 9494
F: +1 847 913 8766

Zebra Technologies Europe Limited

Dukes Meadow
Millboard Road
Bourne End
Buckinghamshire, SL8 5XF, UK
T: +44 (0)1628 556000
F: +44 (0)1628 556001

Zebra Technologies Asia Pacific, LLC

120 Robinson Road
#06-01 Parakou Building
Singapore 068913
T: +65 6858 0722
F: +65 6885 0838

<http://www.zebra.com>