NETWORK CONNECT FOR AUTOMATION

Developer Guide

P1096306-002

Part Number: P1096306-002 Rev. A

Contents

About This Document

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

Contents

Who Should Use This Document

This Developer Guide is intended for use by any person who needs to learn more about using Ethernet/IP with the printer.

How This Document Is Organized

The Developer Guide is set up as follows:

Section	Description
Introduction on page 7	This guide provides information on the use of Network Connect for Automation (NC4A).
Designing and Exporting a Template in ZebraDesigner on page 25	This section describes how to use ZebraDesigner, our label design application, to create a label template, and then export it. We'll use an example template called "EIP" as an example.
Raw Parser Data on page 37	This section describes how to utilize the raw parser data section of the Printer Input Assembly.

1

Introduction

This guide provides information on the use of Network Connect for Automation (NC4A).

Contents

Setup

The following components are required for initial setup and a fully functional Zebra Industrial Ethernet solution.

- Link-OS Ethernet/IP capable printer (such as a ZT400 printer)
- Special firmware SP004629A
- A PC running Windows 7, or higher
- An Industrial Ethernet PLC (Programmable Logic Controller) that supports EtherNet/IP such as CompactLogix
- Ethernet switch or router (if not connecting directly to a PLC) and Ethernet cables
- Sample applications to help accelerate development.

Communications Profile

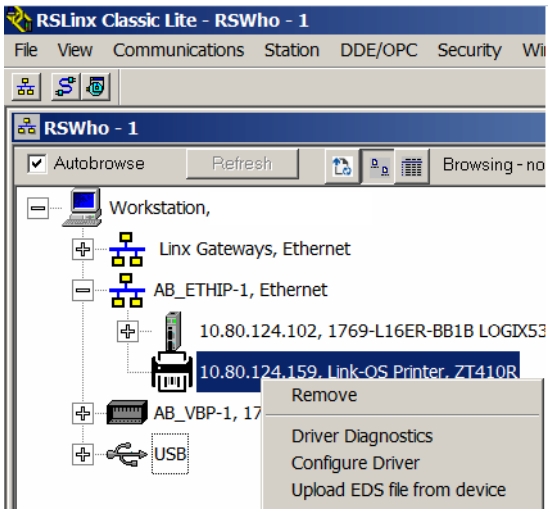
The EtherNet/IP interface on the printer supports CIP Adapter functionality. The device can receive, or be the target of, I/O connections from a CIP Scanner, but is not able to originate connections itself. The interface supports the Generic device profile. The Generic profile provides for all CIP objects that are required by the EtherNet/IP specification.

EDS File

The EtherNet/IP EDS file describes the Identity and I/O capabilities of the printer. The file is used by PLC configuration tools to configure the I/O connections and data tags used to communicate with the printer over the EtherNet/IP network.

The EDS file is keyed by Rockwell for EDS AOP usage (a custom Rockwell AOP will be available from Zebra or in the latest version of Studio 5000). See Figure 1. The latest EDS file can be acquired from the printer directly using RSLinx Classic – RSWho and uploading the EDS file from the device.

Figure 1



Supported Objects

The EtherNet/IP interface will support the following CIP objects:

- Identity
- Message Router
- TCP/IP Interface
- Ethernet Link

Identity Object

The Identity object provides the ability to get device information, product name, product code, revision, serial number, and the ability to reset the printer.

TCP/IP Interface Object

The TCP/IP Interface object provides only the ability to get TCP/IP configuration parameters, set is NOT supported at this time. The IP address configuration can only occur via existing configuration methods on the printer, and may NOT be set through the TCP/IP Interface object. This is known as being hardware configurable. All changes will require a device reset before any changes take effect. Address Conflict Detection is not supported.

Ethernet Link Object

The Ethernet Link object provides the ability to only get the MAC address. Getting the duplex is not supported.

I/O Assemblies

The EtherNet/IP interface includes two Assembly object instances that hold parameters and data used in printing forms, configuring the printer, and monitoring system status.

Printer Input Assembly

Instance: 112

Access: Set

Size: 496 bytes

See Table 1. The printer input assembly allows for ZPL formats to be recalled and printed as well as sending additional ZPL or other device language commands.

Table 1

Sequence Number	2 bytes
Format Number	2 bytes
XF Format Name (^XF)	24 bytes*
XF Field 1 (^FN1)	68 bytes*
XF Field 2 (^FN2)	20 bytes*
XF Field 3 (^FN3)	20 bytes*
XF Field 4 (^FN4)	20 bytes*
XF Field 5 (^XF5)	20 bytes*
Raw Parser Data	320 bytes*

*These ASCII NULL terminated strings contain 2 bytes of string length data in the beginning of the SINT array, which are ignored by the printer. When writing data as an array, start at index 2 instead of index 0.

The sequence number is used to specify that new data is available for the printer to send to the parser. This number must be incremented each time new data is present and ready to process on the printer. When the limit is reached for the sequence number, reset the value and roll it back to zero to begin incrementing again.

The format number should be set to ‘1’ at all times.

The format name should be used to specify the location of the ZPL format used for printing. The format of this data should be:

```
X:Y.Z
```

where X is the drive, Y is the file name, and Z is the extension

The ^FN field data sections are used to specify field data for the corresponding field numbers as referenced by the format name. If the field’s first character is a NULL, the field will be ignored. Field 1 is the only field that supports a larger amount of characters compared to Fields 2 through 5.

The raw parser data section has two modes. If the format name is not NULL, the raw parser data shall be inside the ^XA ^XZ of the recalled format. If the format name is NULL, the raw parser data shall NOT be inside any ^XA ^XZ, and will have to be specified in the raw parser data.

Printer Output Assembly

Instance: 100

Access: Get

Size: 100 bytes

See Table 2. The printer output assembly holds the current system status of the printer as well as any return data from additional parser commands.

Table 2

Previous Sequence Number	2 bytes
Previous Format Number	2 bytes
System Status	44 bytes
Parser Return Data	52 bytes*

*These ASCII NULL terminated strings contain 2 bytes of string length data in the beginning of the SINT array, which are ignored by the printer. When writing data as an array, start at index 2 instead of index 0.

The system status is the same format as the following Set Get Do command:

```
! U1 getvar "zpl.system_status"
```

with a result that looks like:

```
0,0,00000000,00000000,0,00000000,00000000
```

Each section is separated by commas for a total of 7 sections:

- Section 1 – 1 byte, denotes if the printer is paused (1 yes, 0 no)
- Section 2 – 1 byte, denotes if the printer has an error (1 yes, 0 no)
- Section 3 – 8 bytes, currently reserved

- Section 4 – 8 bytes, has the following bit values (those not specified are reserved):

```
MEDIA_OUT = 0x00000001
RIBBON_OUT = 0x00000002
HEAD_OPEN = 0x00000004
CUTTER_ERROR = 0x00000008
HEAD_OVERTEMP = 0x00000010
MOTOR_OVERTEMP = 0x00000020
HEAD_ELEMENT = 0x00000040
HEAD_DETECTION_PROBLEM = 0x00000080
THERMISTOR_OPEN = 0x00000200
PRINTER_PAUSED = 0x00010000
BASIC_RUNTIME_ERROR = 0x00100000
BASIC_FORCED = 0x00200000
RIBBON_AUTH_ERROR = 0x00400000
```

- Section 5 – 1 byte, denotes if the printer has a warning (1 yes, 0 not)
- Section 6 – 8 bytes, currently reserved
- Section 7 – 8 bytes, has the following bit values (those not specified are reserved):

```
PRINTHEAD_MAINT = 0x00000002
REPLACE_HEAD = 0x00000004
HEAD_UNDER_TEMP = 0x00001000
RIBBON_IN_DT_MODE = 0x00002000
BATTERY_LOW = 0x00004000
RFID_ERROR = 0x00008000
RIBBON_LOW = 0x00010000
```

I/O Connections

The EtherNet/IP interface supports a single I/O connection that is used to transfer printer format and status to and from the PLC.

Exclusive Owner Connection

Trigger and Transport: Class 1, Cyclic

RPI Range: 100 - 1000 ms

O -> T

Connection Point: 112

Size: 496 bytes

Format: Assembly instance 112

T -> O

Connection Point: 100

Size: 100 bytes

Format: Assembly instance 100

Configuring Rockwell ControlLogix Communication

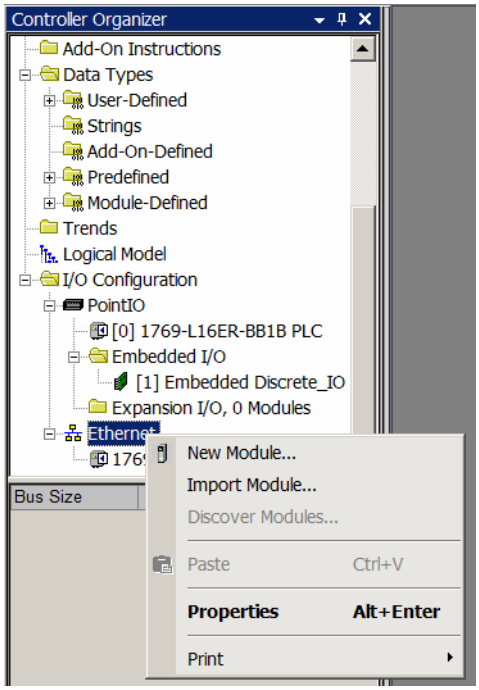
Add the Printer to the I/O Configuration

In order for the PLC to communicate with the printer, it must be added to the I/O configuration in the program.

To add the I/O configuration:

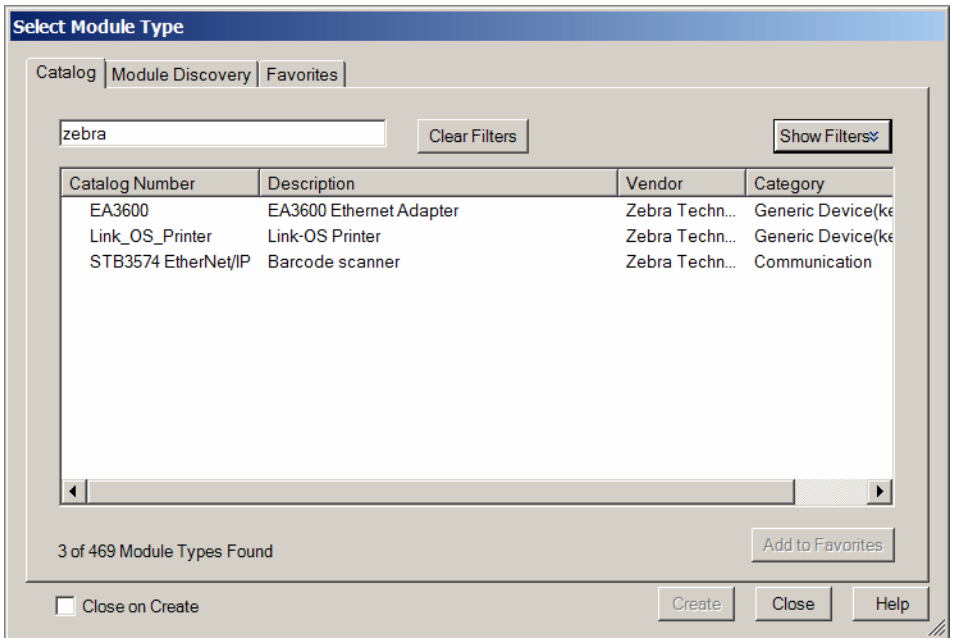
1. Expand the I/O Configuration tree in the Organizer pane to display the Ethernet network.
2. See Figure 2. Right-click on the Ethernet node in the tree and select New Module...

Figure 2



3. See Figure 3. The Select Module Type dialog will be displayed. Search for ‘Zebra’ to see only products from the Zebra Technologies vendor.
   The Link_OS_Printer displays in the device list.

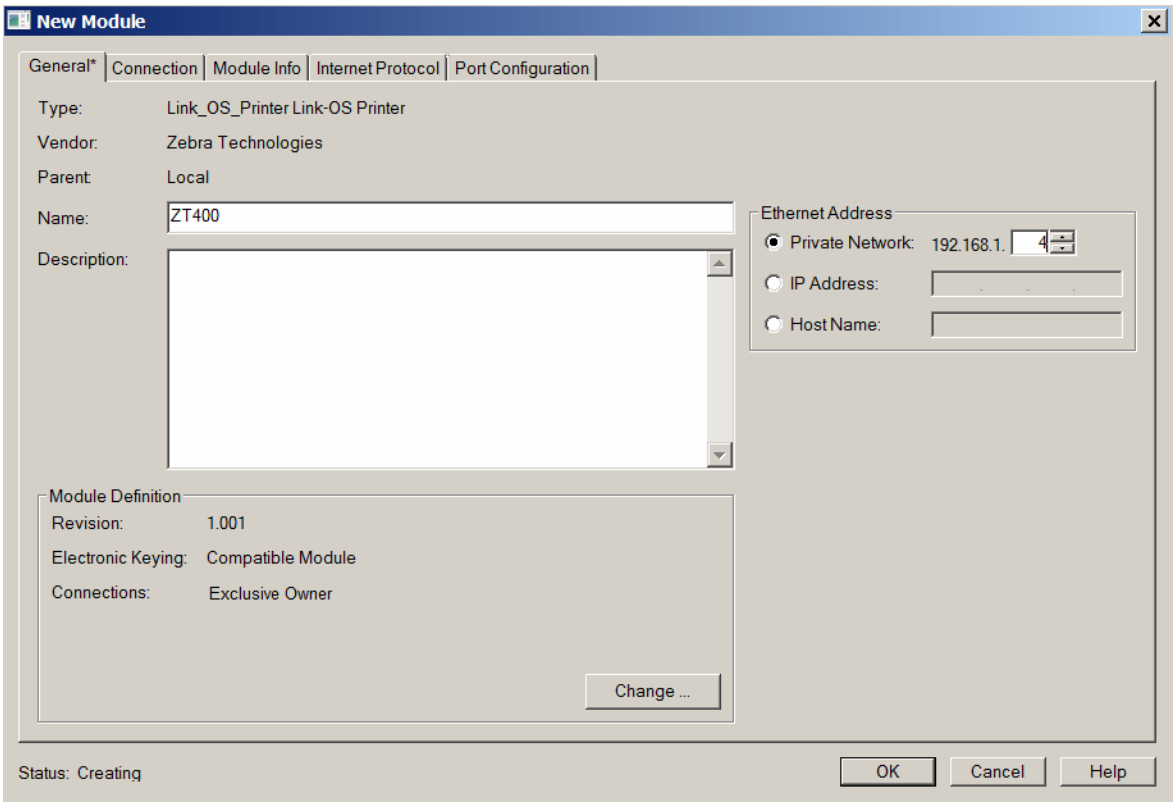
Figure 3



4. Select the Link_OS_Printer from the list and click Create.

5. See Figure 4. The New Module dialog will be displayed. Enter the desired name and IP address of the printer.

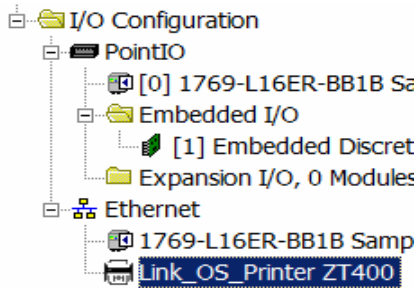
Figure 4



6. Press OK.

7. See Figure 5. The printer is added to the I/O configuration, and appears in the tree.

Figure 5



Create I/O Tags

All I/O connection parameters and I/O Tags are automatically configured when the module is added to the I/O Configuration. Looking at the controller tags you will notice the input in Figure 6:

Figure 6

⊟-ZT400:I	{...}	{...}		ZT:LinkOS:I:0
—ZT400:I.ConnectionFaulted	0		Decimal	BOOL
⊞-ZT400:I.SequenceNumber	6		Decimal	INT
⊞-ZT400:I.XF_FormatNumber	1		Decimal	INT
⊞-ZT400:I.SystemStatus	{...}	{...}		ZT:LinkOS_SysStat:I:0
⊞-ZT400:I.ReturnData	{...}	{...}	ASCII	SINT[52]

As well as the output in Figure 7:

Figure 7

⊟-ZT400:O	{...}	{...}		ZT:LinkOS:O:0
⊞-ZT400:O.SequenceNumber	6		Decimal	INT
⊞-ZT400:O.XF_FormatNumber	1		Decimal	INT
⊞-ZT400:O.XF_FormatName	{...}	{...}	ASCII	SINT[24]
⊞-ZT400:O.XF_Field1	{...}	{...}	ASCII	SINT[68]
⊞-ZT400:O.XF_Field2	{...}	{...}	ASCII	SINT[20]
⊞-ZT400:O.XF_Field3	{...}	{...}	ASCII	SINT[20]
⊞-ZT400:O.XF_Field4	{...}	{...}	ASCII	SINT[20]
⊞-ZT400:O.XF_Field5	{...}	{...}	ASCII	SINT[20]
⊞-ZT400:O.RawParserData	{...}	{...}	ASCII	SINT[320]

Both of which are fully described in I/O Assemblies on page 10. The only additional item is in the Input side under the name ConnectionFaulted, which is active when the PLC is no longer able to communicate with the printer (printer offline). Also, there are no configuration tags used for this printer.

Example

Note • Some values will vary, based on the resolution (Dots per Inch, or DPI) of your printer.

Example • The example shown here is for a 203 DPI printer.

1. Send a sample ZPL form to the printer which makes use of field number 1 and 2.

```
^XA
^DFE:EIP.ZPL
^FO10,10^XGZ:LOGO.PNG^FS
^FO125,10^A0,20,20^FDNetwork Connect^FS
^FO105,75^A0,20,20^FDFor Automation^FS
^FO50,100^A0,30,20^FN1^FS
^FO50,130^B3,,10^FN2^FS
^XZ
```

   The above ZPL creates a form called EIP.ZPL and saves it on the E: FLASH drive of the printer. It displays two text strings, and then has text for field number 1, and a Code-39 barcode for field number 2.

2. Create a new Studio 5000 project, and add the printer to the I/O configuration as described earlier (or feel free to use ZT400-Sample.ACD).

Note • All of the following strings input into arrays must begin at index 2 and not 0 as the first two bytes are ignored.

3. See Figure 8. In the Controller Tags Output Connection for the printer, change the XF_Format_Name to “E:EIP.ZPL”.

Figure 8

ZT400:O	{...}	{...}		ZT:LinkOS:O:0
ZT400:O.SequenceNumber	0		Decimal	INT
ZT400:O.XF_FormatNumber	1		Decimal	INT
ZT400:O.XF_FormatName	{...}	{...}	ASCII	SINT[24]
ZT400:O.XF_FormatName[0]	'$00'		ASCII	SINT
ZT400:O.XF_FormatName[1]	'$00'		ASCII	SINT
ZT400:O.XF_FormatName[2]	'E'		ASCII	SINT
ZT400:O.XF_FormatName[3]	':'		ASCII	SINT
ZT400:O.XF_FormatName[4]	'E'		ASCII	SINT
ZT400:O.XF_FormatName[5]	'I'		ASCII	SINT
ZT400:O.XF_FormatName[6]	'P'		ASCII	SINT
ZT400:O.XF_FormatName[7]	'.'		ASCII	SINT
ZT400:O.XF_FormatName[8]	'Z'		ASCII	SINT
ZT400:O.XF_FormatName[9]	'P'		ASCII	SINT
ZT400:O.XF_FormatName[10]	'L'		ASCII	SINT
ZT400:O.XF_FormatName[11]	'$00'		ASCII	SINT

4. See Figure 9. Change the XF_Field_1 to “Example”.

Figure 9

ZT400:O.XF_Field1	{...}	{...}	ASCII	SINT[68]
ZT400:O.XF_Field1[0]	'$00'		ASCII	SINT
ZT400:O.XF_Field1[1]	'$00'		ASCII	SINT
ZT400:O.XF_Field1[2]	'E'		ASCII	SINT
ZT400:O.XF_Field1[3]	'x'		ASCII	SINT
ZT400:O.XF_Field1[4]	'a'		ASCII	SINT
ZT400:O.XF_Field1[5]	'm'		ASCII	SINT
ZT400:O.XF_Field1[6]	'p'		ASCII	SINT
ZT400:O.XF_Field1[7]	'l'		ASCII	SINT
ZT400:O.XF_Field1[8]	'e'		ASCII	SINT
ZT400:O.XF_Field1[9]	'$00'		ASCII	SINT

5. See Figure 10. Change the XF_Field_2 to “12345”.

Figure 10

ZT400:O.XF_Field2	{...}	{...}	ASCII	SINT[20]
ZT400:O.XF_Field2[0]	'$00'		ASCII	SINT
ZT400:O.XF_Field2[1]	'$00'		ASCII	SINT
ZT400:O.XF_Field2[2]	'1'		ASCII	SINT
ZT400:O.XF_Field2[3]	'2'		ASCII	SINT
ZT400:O.XF_Field2[4]	'3'		ASCII	SINT
ZT400:O.XF_Field2[5]	'4'		ASCII	SINT
ZT400:O.XF_Field2[6]	'5'		ASCII	SINT
ZT400:O.XF_Field2[7]	'$00'		ASCII	SINT

6. See Figure 11-Figure 13. Set the remaining fields (XF_Field_3,4,5) to '$00' NULL for the first SINT.

Figure 11

ZT400:O.XF_Field3	{...}	{...}	ASCII	SINT[20]
ZT400:O.XF_Field3[0]	'$00'		ASCII	SINT
ZT400:O.XF_Field3[1]	'$00'		ASCII	SINT
ZT400:O.XF_Field3[2]	'$00'		ASCII	SINT

Figure 12

ZT400:O.XF_Field4	{...}	{...}	ASCII	SINT[20]
ZT400:O.XF_Field4[0]	'$00'		ASCII	SINT
ZT400:O.XF_Field4[1]	'$00'		ASCII	SINT
ZT400:O.XF_Field4[2]	'$00'		ASCII	SINT

Figure 13

ZT400:O.XF_Field5	{...}	{...}	ASCII	SINT[20]
ZT400:O.XF_Field5[0]	'$00'		ASCII	SINT
ZT400:O.XF_Field5[1]	'$00'		ASCII	SINT
ZT400:O.XF_Field5[2]	'$00'		ASCII	SINT

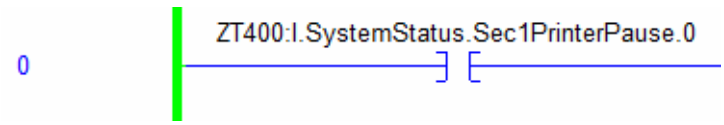
7. See Figure 14. Add a simple ZPL command to get return data back from the parser by putting "~HI" in the Raw_Parser_Data section.

Figure 14

ZT400:O.RawParserData	{...}	{...}	ASCII	SINT[320]
ZT400:O.RawParserData[0]	'$00'		ASCII	SINT
ZT400:O.RawParserData[1]	'$00'		ASCII	SINT
ZT400:O.RawParserData[2]	'~'		ASCII	SINT
ZT400:O.RawParserData[3]	'H'		ASCII	SINT
ZT400:O.RawParserData[4]	'I'		ASCII	SINT
ZT400:O.RawParserData[5]	'$00'		ASCII	SINT

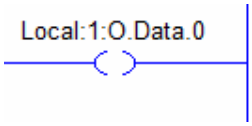
8. See Figure 15. In the MainRoutine, add an XIC (Examine On) ladder element to rung 0 and link it to the I.System_Status_0.0.

Figure 15



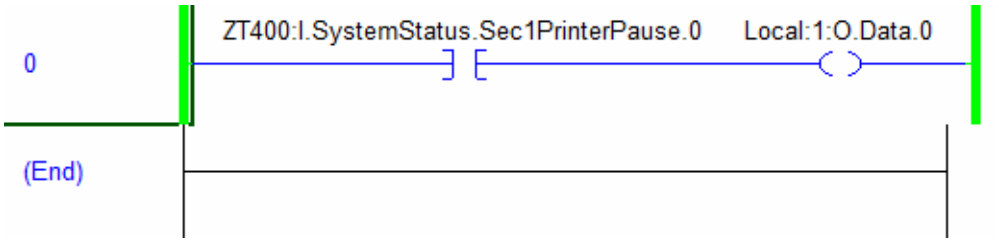
9. See Figure 16. Now, add an OTE (Output Energize) ladder element to rung 0, and link it to one of the PLC's LED lights.

Figure 16



10. The entire rung should look like Figure 17.

Figure 17



11. Save the project, download it to the PLC, and switch into run mode.
12. See Figure 18. In the Controller tags, increment the Sequence_Number.

Figure 18

ZT400:O	{...}	{...}		ZT:LinkOS:O:0
ZT400:O.SequenceNumber	1		Decimal	INT
ZT400:O.XF_FormatNumber	1		Decimal	INT

13. Now that the sequence number has been updated, the data will be sent to the printer. In the steps above, this would be equivalent to sending the following ZPL.

```
^XA
^XFE:EIP.ZPL
^FN1^FDExample^FS
^FN2^FD12345^FS
~HI
^XZ
```

This prints out the label shown in Figure 19.

Figure 19

14. Press PAUSE on the printer and verify the LED lights up on the PLC. Press PAUSE again to verify the LED on the PLC turns off.

15. See Figure 20. Verify that the return data from the ZPL command ~HI is in the Return_Data controller tags section (remember that the first two bytes of data are reserved and ignored).

Figure 20

ZT400:I	{...}	{...}		ZT:LinkOS:I:0
ZT400:I.ConnectionFaulted	0		Decimal	BOOL
ZT400:I.SequenceNumber	1		Decimal	INT
ZT400:I.XF_FormatNumber	1		Decimal	INT
ZT400:I.SystemStatus	{...}	{...}		ZT:LinkOS_SysStat:I:0
ZT400:I.ReturnData	{...}	{...}	ASCII	SINT[52]
ZT400:I.ReturnData[0]	'$19'		ASCII	SINT
ZT400:I.ReturnData[1]	'$00'		ASCII	SINT
ZT400:I.ReturnData[2]	'$02'		ASCII	SINT
ZT400:I.ReturnData[3]	'Z'		ASCII	SINT
ZT400:I.ReturnData[4]	'T'		ASCII	SINT
ZT400:I.ReturnData[5]	'4'		ASCII	SINT
ZT400:I.ReturnData[6]	'1'		ASCII	SINT
ZT400:I.ReturnData[7]	'0'		ASCII	SINT

Notes •

2

Designing and Exporting a Template in ZebraDesigner

This section describes how to use ZebraDesigner, our label design application, to create a label template, and then export it. We'll use an example template called "EIP" as an example.

Contents

Download and Install ZebraDesigner

This procedure describes how to download and install the ZebraDesigner software.

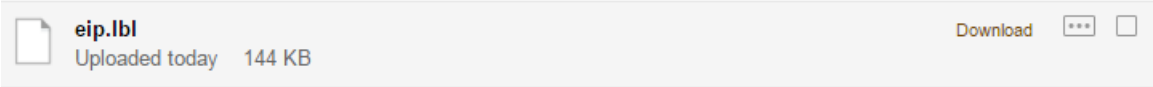
1. The ZebraDesigner software can be downloaded from our website. To open the download page, visit http://www.zebra.com/zebradesigner.
2. Click the download link. You will be directed to the End User License Agreement.
3. To download the software, you must review and agree to the terms of the End User License Agreement.
   If you agree, click the ACCEPT AND BEGIN DOWNLOAD NOW button.
4. On your computer, browse and locate the downloaded file and double-click it to start the installation wizard.
   Follow the on screen instructions to complete the ZebraDesigner installation.

Download the Sample Template

This procedure describes how to acquire the EIP Label template that you will modify in the ZebraDesigner software.

The EIP sample label was designed using the ZebraDesigner software and has been made available for download from a Cloud storage site.

1. To access the Zebra Cloud storage on box.com, go to https://zebra.app.box.com/v/eip. You should see two files – **eip.lbl** and **eip.zpl**.
2. Click the Download link next to the eip.lbl file.



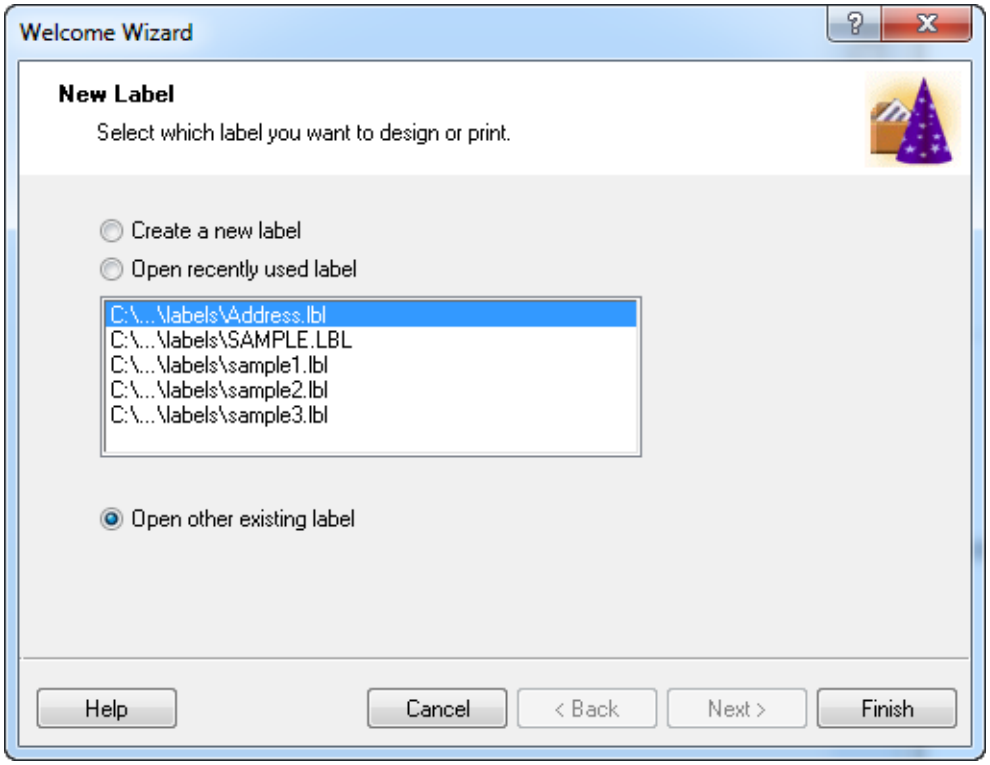
3. Note the location of the downloaded file as this is the template file that will be modified in ZebraDesigner in the next section.

Modify the Template File in ZebraDesigner

This procedure describes how to open the label file in ZebraDesigner and make a simple modification to it. It will then be transferred to your printer for use with your PLC-based program.

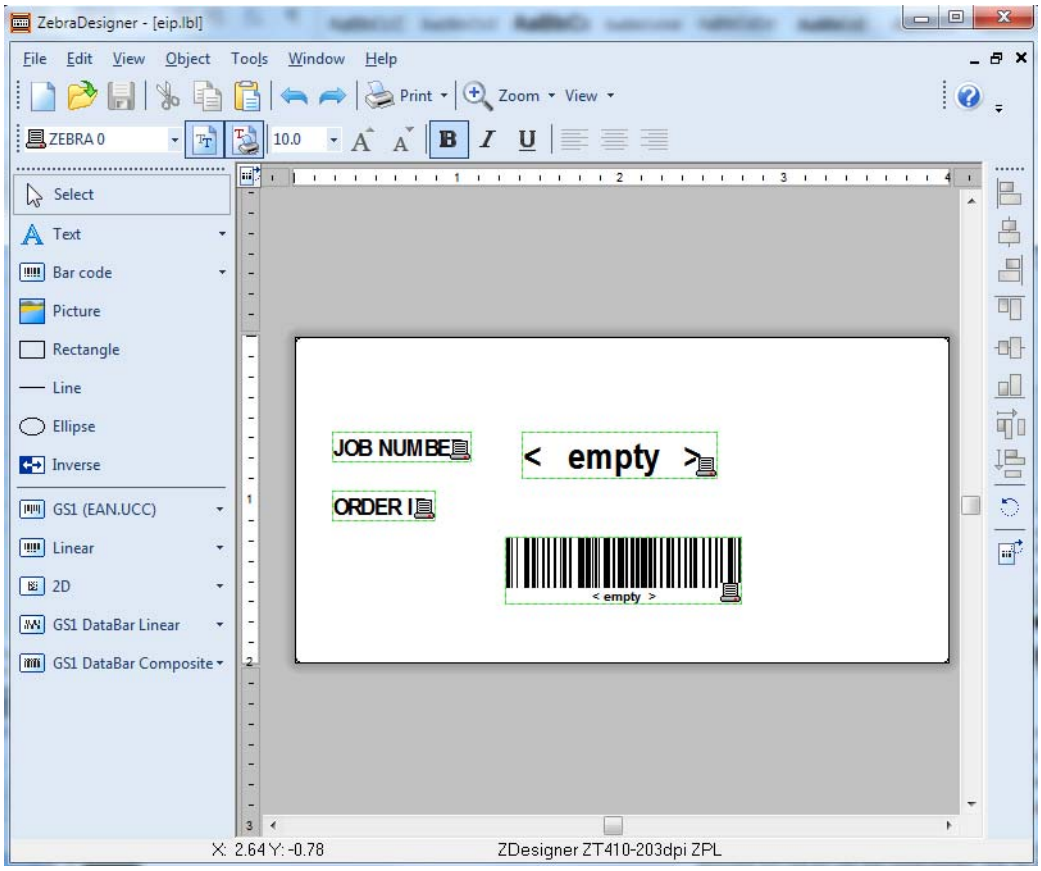
1. First, it's necessary to install the Windows printer driver that was used in the design of the template.
   a. From the Start menu browse to the Zebra Technologies program group and then launch Printer install from the ZebraDesigner 2 folder.
   b. If required, click yes on the User Account Control dialog.
      The printer installation wizard will be displayed.
2. Click Next.
3. Click Install Printer.
4. From the list of printers, select ZDesigner ZT410-203dpi ZPL, and click Next.

5. In the available ports list select FILE: and click Next.
6. Uncheck all boxes on the Additional Install Options dialog and click Finish.
   The driver installation will complete and the installation wizard will close automatically.
7. From the Start menu, browse to the Zebra Technologies program group and launch ZebraDesigner 2 from the ZebraDesigner 2 folder.
   The Welcome Wizard will be displayed.
8. Select the Open other existing label option and click Finish.
   The Open dialog will be displayed.

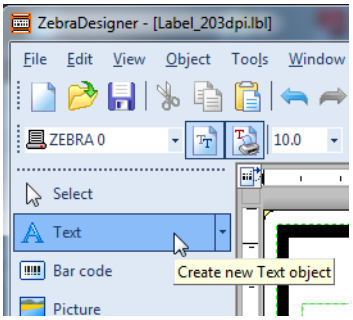


9. Browse to the location of the template file that was downloaded in the previous section. Select the eip.lbl file and click Open.

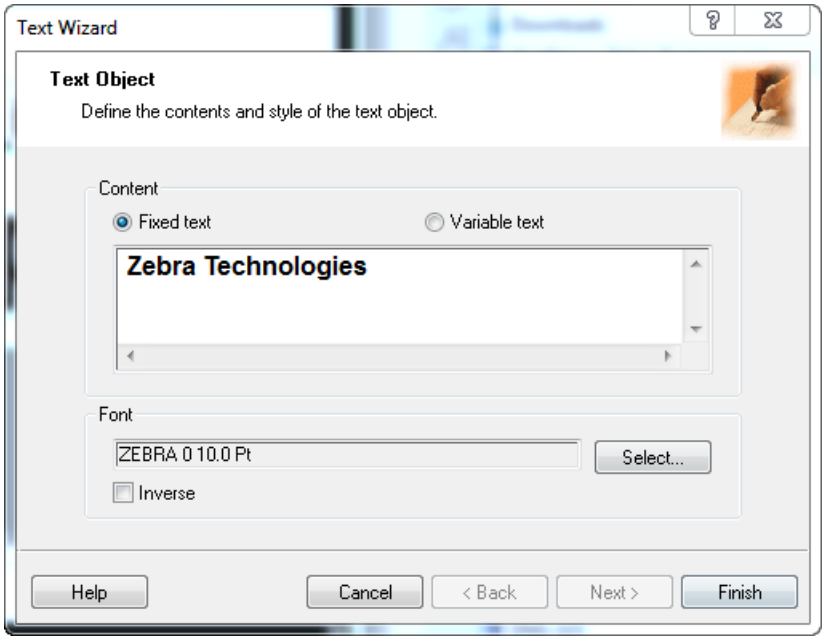
10. The template file should be displayed in ZebraDesigner. You will be adding a fixed text “company name” field to the template. If you want to, you can use your mouse to draw a box around the existing fields on the label, and then drag them downward to create space for the new field.



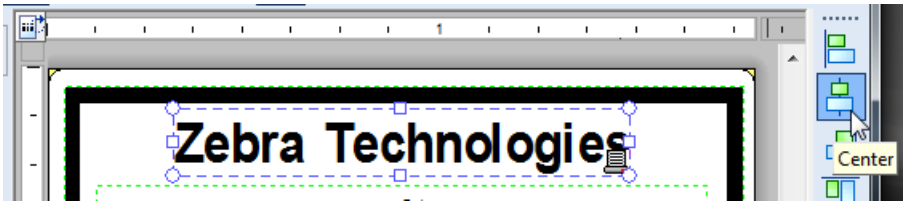
11. Click the Create new Text object option, and then click in the space created at the top of the template.
    The Text Wizard will be displayed.

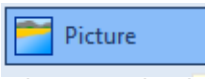


12. Enter your company name or some other text in the Content area and click Finish.



13. Adjust the position of the text field as necessary. To center the field relative to the label, press and hold the CTRL key and click the Center button on the alignment toolbar.



14. Click on the Picture icon 

on the left hand tool bar and then on the label design area to add an image to the label.

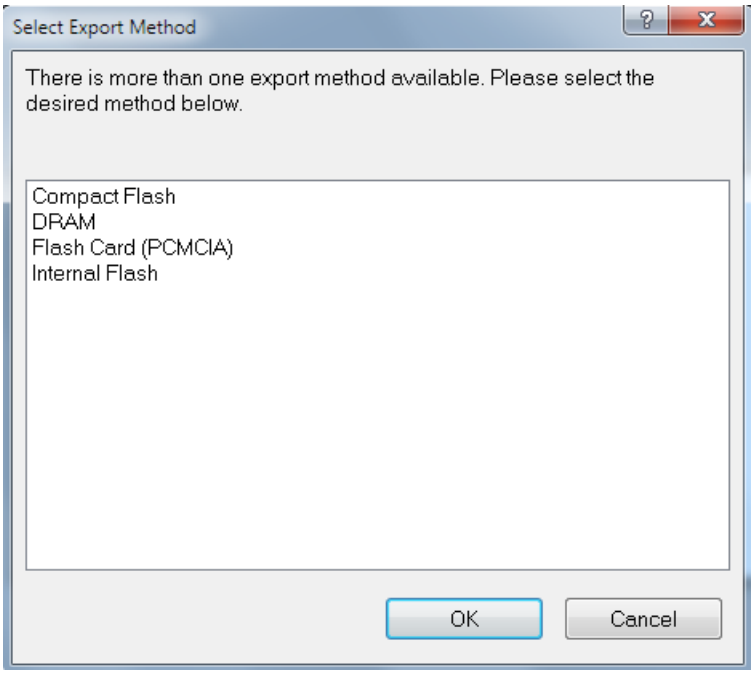
15. Navigate the location of your image file and select the image – we recommend using a .PNG file.

    An example file is included on the previously mentioned Box site.

16. You can grab the edge of the image and drag to resize the image.

17. Click Save to save the changes to the label.

18. Select the Export to Printer option, and then click Internal Flash – and then OK.



This will send the template to your printer, storing it in the E memory location.

19. For your reference, using the eip.lbl file as described and edited here will result in the following ZPL commands being sent to the printer.

Note • Some values will vary, based on the resolution (Dots per Inch, or DPI) of your printer.

Example • The example shown here is for a 203 DPI printer.

```
^XA
^DFE:eip.ZPL^FS
~TA000~JSN^LT0^MNW^MTT^PON^PMN^LH0,0^JMA^PR6,6^MD15^LRN^
CI0
^MMT
^PW812
^LL0406
^LS0
^FO672,0^GFA,02048,02048,00016,:Z64:
eJztlLFOwzAQhm3SqBUSTZHite1LIAUUxalEdldypPYt+gQ0iIGtvEIf
o0gMlRjY6NIVKYiNCTZLVAlXhvjOEwsL4O3r53N/
2+cw9j9+fhw4zFPKXkkw5YrMLj1DWPEJqTbUM03rWeV4Qf1scLXCrPwR
8VX7heTTfJxilpcfBVme5aQ+KOc7zL4SGsf3jKwRc65CPN1bm3iIy1MV
9rB/M3GE65UKyQI7Eyc4gJ6EIQ4gV3EicX6oFzg/1B8WDbb8CVn/
wAPfofnxBK8A38X5od5HfhnFSYDz98DbCdx0wLfL5oct2/u04Qu29/
aKBXjh2ysYFAn4quGjFNI92RNoL2H3tfVcwe5z6/
kuKFmNTnALf448m0P/
StRiAvob5WPBGsIX+ASQZPsbHBJ22hdu0GHtsHRYOBysKeNwXwGWlN33
yUqHXT9kf2jkIhspe0S8lpv3yn7CuD7Jxjk6wn7UPd1cN56FvdZxhvoz
An9/M2u4J7Ix9kZunuuFfeIqPJ/
qlv0ElFBfDZr+4ymsr+0nhr+eLW4fkzubT2TTzL5Qr4L8D6v+d7f/
q8Yn1KBy+w==:1C22
^FT403,165^A0N,45,45^FB0,1,0,C^FH\^FN12"EnterJobID"^FS
^BY2,3,61^FT263,311^BCN,,Y,N,N,A
^FN11"EnterOrderID"^FS
^FT48,147^A0N,28,28^FH\^FDJOB NUMBER:^FS
^FT48,221^A0N,28,28^FH\^FDORDER ID:^FS
^FT221,74^A0N,48,48^FH\^FDZebra Techologies^FS
^XZ
```

The next step is to create your PLC-based program so that it can recall the template and populate it with data at print time. See the section of this document that details Recalling and Printing with a Template on page 32.

Recalling and Printing with a Template

Using the template previously created by ZebraDesigner, we will recall and update the template format fields with custom data and print it from the PLC.

1. Create a new Studio 5000 project and add the printer to the I/O configuration as described earlier (or feel free to use ZT400-Sample.ACD).
2. See Figure 21. In the Controller Tags Output Connection for the printer, change the XF_Format_Name to "`E:EIP.ZPL`".

Figure 21

ZT400:O	{...}	{...}		ZT:LinkOS:O:0
ZT400:O.SequenceNumber	0		Decimal	INT
ZT400:O.XF_FormatNumber	1		Decimal	INT
ZT400:O.XF_FormatName	{...}	{...}	ASCII	SINT[24]
ZT400:O.XF_FormatName[0]	'$00'		ASCII	SINT
ZT400:O.XF_FormatName[1]	'$00'		ASCII	SINT
ZT400:O.XF_FormatName[2]	'E'		ASCII	SINT
ZT400:O.XF_FormatName[3]	':'		ASCII	SINT
ZT400:O.XF_FormatName[4]	'E'		ASCII	SINT
ZT400:O.XF_FormatName[5]	'I'		ASCII	SINT
ZT400:O.XF_FormatName[6]	'P'		ASCII	SINT
ZT400:O.XF_FormatName[7]	'.'		ASCII	SINT
ZT400:O.XF_FormatName[8]	'Z'		ASCII	SINT
ZT400:O.XF_FormatName[9]	'P'		ASCII	SINT
ZT400:O.XF_FormatName[10]	'L'		ASCII	SINT
ZT400:O.XF_FormatName[11]	'$00'		ASCII	SINT

3. See Figure 22. Set the XF fields (`XF_Field_1,2,3,4,5`) to '`$00`' NULL for the first SINT. The fields used in ZebraDesigner start with 11 and 12, so we cannot use the ones built-in for support in the data model. (Only field 3 is shown below.)

Figure 22

ZT400:O.XF_Field3	{...}	{...}	ASCII	SINT[20]
ZT400:O.XF_Field3[0]	'$00'		ASCII	SINT
ZT400:O.XF_Field3[1]	'$00'		ASCII	SINT
ZT400:O.XF_Field3[2]	'$00'		ASCII	SINT

4. See Figure 23. Set the Raw Parser Data section to fill in field number 11 with data (`^FN11^FD12345^FS`).

Figure 23

ZT400:O.RawParserData	{...}	{...}	ASCII	SINT[320]
ZT400:O.RawParserData[0]	'$00'		ASCII	SINT
ZT400:O.RawParserData[1]	'$00'		ASCII	SINT
ZT400:O.RawParserData[2]	'^'		ASCII	SINT
ZT400:O.RawParserData[3]	'F'		ASCII	SINT
ZT400:O.RawParserData[4]	'N'		ASCII	SINT
ZT400:O.RawParserData[5]	'1'		ASCII	SINT
ZT400:O.RawParserData[6]	'1'		ASCII	SINT
ZT400:O.RawParserData[7]	'^'		ASCII	SINT
ZT400:O.RawParserData[8]	'F'		ASCII	SINT
ZT400:O.RawParserData[9]	'D'		ASCII	SINT
ZT400:O.RawParserData[10]	'1'		ASCII	SINT
ZT400:O.RawParserData[11]	'2'		ASCII	SINT
ZT400:O.RawParserData[12]	'3'		ASCII	SINT
ZT400:O.RawParserData[13]	'4'		ASCII	SINT
ZT400:O.RawParserData[14]	'5'		ASCII	SINT
ZT400:O.RawParserData[15]	'^'		ASCII	SINT
ZT400:O.RawParserData[16]	'F'		ASCII	SINT
ZT400:O.RawParserData[17]	'S'		ASCII	SINT

5. See Figure 24. Set the remaining Raw Parser Data section to fill in field number 12 with data (^FN12^FD5^FS).

Figure 24

⊞ ZT400:O.RawParserData[17]	'S'		ASCII	SINT
⊞ ZT400:O.RawParserData[18]	'^'		ASCII	SINT
⊞ ZT400:O.RawParserData[19]	'F'		ASCII	SINT
⊞ ZT400:O.RawParserData[20]	'N'		ASCII	SINT
⊞ ZT400:O.RawParserData[21]	'1'		ASCII	SINT
⊞ ZT400:O.RawParserData[22]	'2'		ASCII	SINT
⊞ ZT400:O.RawParserData[23]	'^'		ASCII	SINT
⊞ ZT400:O.RawParserData[24]	'F'		ASCII	SINT
⊞ ZT400:O.RawParserData[25]	'D'		ASCII	SINT
⊞ ZT400:O.RawParserData[26]	'5'		ASCII	SINT
⊞ ZT400:O.RawParserData[27]	'^'		ASCII	SINT
⊞ ZT400:O.RawParserData[28]	'F'		ASCII	SINT
⊞ ZT400:O.RawParserData[29]	'S'		ASCII	SINT
⊞ ZT400:O.RawParserData[30]	'$00'		ASCII	SINT

6. Save the project, download it to the PLC, and switch into run mode.
7. See Figure 25. In the Controller tags, increment the Sequence_Number.

Figure 25

⊟ ZT400:O	{...}	{...}		ZT:LinkOS:O:0
⊞ ZT400:O.SequenceNumber	1		Decimal	INT
⊞ ZT400:O.XF_FormatNumber	1		Decimal	INT

8. The effective ZPL this will produce internally on the printer looks like this:

```
^XA
^XFE:EIP.ZPL
^FN11^FD12345^FS
^FN12^FD5^FS
^XZ
```

9. See Figure 26. This should print out a label that looks like this:

Figure 26

Create Your Own Solution

Using the techniques described in the above sections of this document along with information from the Developers Guide, create your own solution by following the steps below.

1. Create your own template using ZebraDesigner based on tasks performed in the Modify the Template File in ZebraDesigner on page 26.
   a. To access the Software Tutorials tab, go to http://www.zebra.com/zebradesigner
   b. Click on ZebraDesigner.
   c. Click on the Software Tutorials tab.
2. Transfer your template to your printer.
3. Create your PLC-based program to recall and populate the variable fields on your label.

3

Raw Parser Data

This section describes how to utilize the raw parser data section of the Printer Input Assembly.

Contents

Embedded Parser Commands

There are two ways of sending parser commands: one is embedded within a ZPL format, and another is entirely on its own. The first method of embedding it inside a format was described in the Example on page 18. The ~HI ZPL command (Host Information) was sent along with the ZPL format. The return data from that command was then populated in the Printer Output Assembly.

The second method occurs when the format name is not specified (NULL as the first byte of XF_Format_Name).

Figure 27

⊟ ZT400:O.XF_FormatName	{...}	{...}	ASCII	SINT[24]
⊞ ZT400:O.XF_FormatName[0]	'$00'		ASCII	SINT
⊞ ZT400:O.XF_FormatName[1]	'$00'		ASCII	SINT
⊞ ZT400:O.XF_FormatName[2]	'$00'		ASCII	SINT

In this situation, the printer will only process what is in the Raw Parser Data section and ignore all the other fields.

For example, if you input the data shown in Figure 28 in the raw parser data section

```
(! U1 getvar "appl.name"):
```

Figure 28

ZT400:O.RawParserData	{...}	{...}	ASCII	SINT[320]
ZT400:O.RawParserData[0]	'$00'		ASCII	SINT
ZT400:O.RawParserData[1]	'$00'		ASCII	SINT
ZT400:O.RawParserData[2]	'!'		ASCII	SINT
ZT400:O.RawParserData[3]	' '		ASCII	SINT
ZT400:O.RawParserData[4]	'U'		ASCII	SINT
ZT400:O.RawParserData[5]	'1'		ASCII	SINT
ZT400:O.RawParserData[6]	' '		ASCII	SINT
ZT400:O.RawParserData[7]	'g'		ASCII	SINT
ZT400:O.RawParserData[8]	'e'		ASCII	SINT
ZT400:O.RawParserData[9]	't'		ASCII	SINT
ZT400:O.RawParserData[10]	'v'		ASCII	SINT
ZT400:O.RawParserData[11]	'a'		ASCII	SINT
ZT400:O.RawParserData[12]	'r'		ASCII	SINT
ZT400:O.RawParserData[13]	' '		ASCII	SINT
ZT400:O.RawParserData[14]	'"'		ASCII	SINT
ZT400:O.RawParserData[15]	'a'		ASCII	SINT
ZT400:O.RawParserData[16]	'p'		ASCII	SINT
ZT400:O.RawParserData[17]	'p'		ASCII	SINT
ZT400:O.RawParserData[18]	'l'		ASCII	SINT
ZT400:O.RawParserData[19]	'.'		ASCII	SINT
ZT400:O.RawParserData[20]	'n'		ASCII	SINT
ZT400:O.RawParserData[21]	'a'		ASCII	SINT
ZT400:O.RawParserData[22]	'm'		ASCII	SINT
ZT400:O.RawParserData[23]	'e'		ASCII	SINT
ZT400:O.RawParserData[24]	'"'		ASCII	SINT
ZT400:O.RawParserData[25]	' '		ASCII	SINT
ZT400:O.RawParserData[26]	'$00'		ASCII	SINT

After incrementing the sequence number, you would receive the output shown in Figure 29 (SP75-004629A):

Figure 29

ZT400:I.ReturnData	{...}	{...}	ASCII	SINT[52]
ZT400:I.ReturnData[0]	'$19'		ASCII	SINT
ZT400:I.ReturnData[1]	'$00'		ASCII	SINT
ZT400:I.ReturnData[2]	'"'		ASCII	SINT
ZT400:I.ReturnData[3]	'S'		ASCII	SINT
ZT400:I.ReturnData[4]	'P'		ASCII	SINT
ZT400:I.ReturnData[5]	'7'		ASCII	SINT
ZT400:I.ReturnData[6]	'5'		ASCII	SINT
ZT400:I.ReturnData[7]	'-'		ASCII	SINT
ZT400:I.ReturnData[8]	'0'		ASCII	SINT
ZT400:I.ReturnData[9]	'0'		ASCII	SINT
ZT400:I.ReturnData[10]	'4'		ASCII	SINT
ZT400:I.ReturnData[11]	'6'		ASCII	SINT
ZT400:I.ReturnData[12]	'2'		ASCII	SINT
ZT400:I.ReturnData[13]	'9'		ASCII	SINT
ZT400:I.ReturnData[14]	'A'		ASCII	SINT

Configuration Commands Inside Format

Some customers may want to change the label length, print quantity, or any other configuration of the label or printer within a format. This can be achieved in much the same way as described previously when printing a format. You only need to add the desired ZPL configuration commands to the raw parser data section and they will be sent within the format.

For example, to change the label length to 500 dot rows (`^LL500`) and print two labels (`^PQ2`), you can add the following shown in Figure 30 to the raw parser data section:

Figure 30

ZT400:O.RawParserData	{...}	{...}	ASCII	SINT[320]
ZT400:O.RawParserData[0]	'$00'		ASCII	SINT
ZT400:O.RawParserData[1]	'$00'		ASCII	SINT
ZT400:O.RawParserData[2]	'^'		ASCII	SINT
ZT400:O.RawParserData[3]	'L'		ASCII	SINT
ZT400:O.RawParserData[4]	'L'		ASCII	SINT
ZT400:O.RawParserData[5]	'5'		ASCII	SINT
ZT400:O.RawParserData[6]	'0'		ASCII	SINT
ZT400:O.RawParserData[7]	'0'		ASCII	SINT
ZT400:O.RawParserData[8]	'^'		ASCII	SINT
ZT400:O.RawParserData[9]	'P'		ASCII	SINT
ZT400:O.RawParserData[10]	'Q'		ASCII	SINT
ZT400:O.RawParserData[11]	'2'		ASCII	SINT
ZT400:O.RawParserData[12]	'$00'		ASCII	SINT

Using More Than Five Field Numbers

Currently the data model for the printer input assembly supports up to five format fields to be specified. If more fields are desired in a particular ZPL form, you can add the commands to support them inside the raw parser data section.

Say, for example, you would want to support a form with 7 total fields, you could use the following ZPL data in the raw parser data section. First, specify the field number (^FN), then the field data (^FD), and the field separator (^FS). The sixth field might look like Figure 31:

Figure 31

ZT400:O.RawParserData	{...}	{...}	ASCII	SINT[320]
ZT400:O.RawParserData[0]	'$00'		ASCII	SINT
ZT400:O.RawParserData[1]	'$00'		ASCII	SINT
ZT400:O.RawParserData[2]	'^'		ASCII	SINT
ZT400:O.RawParserData[3]	'F'		ASCII	SINT
ZT400:O.RawParserData[4]	'N'		ASCII	SINT
ZT400:O.RawParserData[5]	'6'		ASCII	SINT
ZT400:O.RawParserData[6]	'^'		ASCII	SINT
ZT400:O.RawParserData[7]	'F'		ASCII	SINT
ZT400:O.RawParserData[8]	'D'		ASCII	SINT
ZT400:O.RawParserData[9]	'a'		ASCII	SINT
ZT400:O.RawParserData[10]	'b'		ASCII	SINT
ZT400:O.RawParserData[11]	'c'		ASCII	SINT
ZT400:O.RawParserData[12]	'^'		ASCII	SINT
ZT400:O.RawParserData[13]	'F'		ASCII	SINT
ZT400:O.RawParserData[14]	'S'		ASCII	SINT

The seventh field would immediately follow the sixth and end with a NULL terminator as shown in Figure 32:

Figure 32

ZT400:O.RawParserData[14]	'S'		ASCII	SINT
ZT400:O.RawParserData[15]	'^'		ASCII	SINT
ZT400:O.RawParserData[16]	'F'		ASCII	SINT
ZT400:O.RawParserData[17]	'N'		ASCII	SINT
ZT400:O.RawParserData[18]	'7'		ASCII	SINT
ZT400:O.RawParserData[19]	'^'		ASCII	SINT
ZT400:O.RawParserData[20]	'F'		ASCII	SINT
ZT400:O.RawParserData[21]	'D'		ASCII	SINT
ZT400:O.RawParserData[22]	'x'		ASCII	SINT
ZT400:O.RawParserData[23]	'y'		ASCII	SINT
ZT400:O.RawParserData[24]	'z'		ASCII	SINT
ZT400:O.RawParserData[25]	'^'		ASCII	SINT
ZT400:O.RawParserData[26]	'F'		ASCII	SINT
ZT400:O.RawParserData[27]	'S'		ASCII	SINT
ZT400:O.RawParserData[28]	'$00'		ASCII	SINT

Sending Data More Than 500 Bytes

In some cases, you may need to send more data than what fits within the existing Printer Input Assembly size of 496 bytes. If this is the case, the raw parser data section must be used exclusively by breaking up the data into chunks that fit inside the raw parser data size of 320 bytes.

1. Ensure that the XF Format Name is set to `NULL`.
2. Put as much of the ZPL data into the raw parser data section that will fit:

   ```
   ^XA… up to 320 bytes
   ```

3. Increment the sequence number to send that data to the parser.
4. Put the next chunk of ZPL data into the raw parser data section, and increment the sequence number each time more data is sent to the parser until the end of the data.

   ```
   remaining data…^XZ
   ```

Notes •

Corporate Headquarters

Zebra Technologies Corporation
3 Overlook Point
Lincolnshire, IL 60069 USA
T: +1 847 634 6700
Toll-free +1 866 230 9494
F: +1 847 913 8766

http://www.zebra.com

P1096306-002