

ITX 3000/2000 & ITL 2000 Printer Series *Programmer's Guide*

*ITX-2000/3000 V1.20
ITL-2000 V1.00*



PRACTICAL AUTOMATION INC.

Table of Contents

- 1 IMAGE PRINTING OVERVIEW..... 6**
- 2 SIMPLE TEXT PROGRAMMING..... 7**
 - 2.1 Rows and Columns..... 7**
 - 2.1.1 Setting the Row and Column Location..... 7
 - 2.1.2 Row and Column Limits..... 7
 - 2.1.3 Print Image Offset Values..... 8
 - 2.2 Font Selection..... 8**
 - 2.2.1 Factory Installed Fonts..... 8
 - 2.2.2 User Installed Fonts..... 8
 - 2.3 Rotation..... 8**
 - 2.4 Scaling..... 8**
 - 2.5 Style – Bold and Italics..... 9**
 - 2.5.1 Bold..... 9
 - 2.5.2 Italics..... 9
- 3 PRINT FUNCTIONS..... 10**
 - 3.1 Simple Printing..... 10**
 - 3.2 Simple Example..... 10**
 - 3.3 Default Settings..... 10**
 - 3.4 Advanced Printing..... 11**
 - 3.4.1 Repeated Tickets..... 11
 - 3.4.2 Print and Hold Function..... 11
 - 3.4.3 Ticket Packeting..... 11
 - 3.4.4 Clear Buffer Command..... 11
 - 3.4.5 Cash Drawer Solenoid Driver..... 11
 - 3.4.6 Ignore Data Command..... 11
 - 3.4.7 Feed and Retract Ticket (Tearbar Only)..... 12
 - 3.5 Extra Print Commands..... 12**
- 4 LINES AND BOXES..... 13**
- 5 TICKET COUNT..... 13**
- 6 INVERTED CHARACTERS..... 13**
- 7 SHADING OPTIONS..... 14**
 - 7.1 Background Shading – Characters and Regions..... 14**
 - 7.2 Foreground Shading – Characters Only..... 14**

[7.3 Shading Densities and Patterns..... 14](#)

[8 BAR CODES..... 15](#)

[9 USER FLASH SPACE..... 16](#)

[9.1 Clearing the User Flash Space..... 16](#)

[10 GRAPHICS..... 17](#)

[10.1 “Dot-matrix Printer” Graphics.....17](#)

[10.2 PCX Graphics..... 17](#)

[10.3 Storing Graphics in Nonvolatile Memory..... 18](#)

[10.4 Cautions for Storing Logos.....18](#)

[10.5 Recalling Stored Logos..... 18](#)

[11 DOWNLOADING USER FONTS..... 19](#)

[11.1 Differences Between ITX and ETX..... 20](#)

[12 PRINT AREA..... 21](#)

[12.1 Overprinting Precautions..... 21](#)

[13 SAMPLE TICKET..... 22](#)

[13.1 Sample Ticket Command Sequence..... 23](#)

[14 COMMAND SUMMARY TABLE..... 24](#)

[14.1 Windows Driver Support Commands..... 29](#)

[14.2 Unsupported Commands..... 29](#)

[14.3 Illegal Commands..... 29](#)

[15 SERIAL PORT CONFIGURATION..... 30](#)

[15.1 Serial Port Flow Control Summary.....30](#)

[15.2 XON/XOFF Data Flow Control..... 30](#)

[15.3 Busy Hardware Flow Control..... 30](#)

[15.4 RS-232 Serial Cable Connections 31](#)

[16 PRINTER STATUS - ERROR CONDITION FLOW CONTROL EXCEPTIONS32](#)

16.1 Data Flow Control: 32

16.2 Receive Data Buffer Overflow: 32

16.3 Error Condition Flow Control Exceptions: 32

16.4 Error States: 32

 16.4.1 Normal Operation 32

 16.4.2 Operator Recoverable Error..... 32

 16.4.3 Non-Recoverable Error..... 32

17 EEPROM HOST CONFIGURATION..... 33

17.1 Programming Overview.....33

17.2 Programming Specifics.....33

17.3 Programming Sequence.....34

17.4 Simplified Programming..... 34

17.5 Reading the Odometer.....34

17.6 Reading the Error Queue..... 35

18 PRINTING DIAGNOSTIC TICKETS..... 35

19 PRINTER TO HOST STATUS INFORMATION..... 36

19.1 Serial Port Status.....36

 19.1.1 Data Buffering Change.....36

 19.1.2 Unsolicited Serial Status.....36

 19.1.3 Serial Only - Status Commands.....37

 19.1.4 Other Serial Status Commands.....38

 19.1.5 Sending Serial Status Commands while Busy..... 38

19.2 Parallel Port Status..... 39

 19.2.1 Data Buffering Change.....39

 19.2.2 Unsolicited Status.....39

 19.2.3 Parallel IEEE 1284 Interface Standard.....39

 19.2.4 Parallel Interface - Compatibility Mode Status39

 19.2.5 Parallel Interface IEEE-1284 Reverse Channel Status.....39

19.3 Selection of Detailed Status Information.....41

 19.3.1 Status Data Request41

 19.3.2 Status Group Selection Commands.....41

 19.3.3 Status Information Data Organization.....42

19.4 Status Error Code List.....46

20 COMMAND DRIVEN FLASH MEMORY RE-PROGRAMMING PROCEDURES..... 48

20.1 Overview.....48

20.2 Flash Memory Reprogramming Mode Entry.....48

20.2.1 Manual Entry	48
20.2.2 Command Entry.....	48
20.3 Flash Read Only Memory Organization.....	48
20.4 Firmware Binary Data Files.....	48
20.5 Flash Memory Reprogramming Actions.....	49
20.6 Command Driven Process for Reprogramming the Flash Memory:.....	49
20.6.1 Command Driven Process Details:.....	51
21 ASCII CHARACTER SETS.....	53
21.1 Fonts 1, 2 and 5.....	54
21.2 Fonts 3 and 4.....	55
21.3 Fonts 6 and 7.....	56
21.4 Fonts 8, 9and 10.....	57
21.5 Fonts 11 and 12.....	58
21.6 Font 13 with LEGEND.....	59
21.7 Actual Printouts of Fonts 1-13.....	60

1 IMAGE PRINTING OVERVIEW

ITX and ITL: The ITX and ITL printers have almost identical processing of all FGL commands. The ITL is the low cost version of the ITX family of ticket printers. The ITL is only available at the 203dpi resolution. If there is any specific command that does not work exactly the same on all printers, it will be clearly noted. In general, any reference to ITX also refers to ITL.

The standard ITX printer, with FGL emulation firmware, is an Image Printer. This means that the printer decodes all of the host commands and creates a complete “image” of the ticket in memory before it is printed. This method allows the user to design many different tickets using a simple set of commands that can easily be edited. Using powerful commands containing just a few bytes, these commands are converted by the printer into barcodes, boxes, and fonts, all with different image rotations. When combined with the User Download tools, which allow custom fonts and graphics to be stored in the printer's Flash ROM, the printer's command efficiency is extraordinary.

The ITX has two “pages” of image memory. This allows the printer to create the image for the second ticket at the same time that the first ticket is being printed. The image of the page being printed is automatically cleared, during the printing process, unless the “print and hold” feature is used (see section 3.4.2). In most cases, a series of tickets can be printed without any noticeable pause.

By contrast, a “line printer” converts short command strings into text using internal fonts, one printed line at a time. Free-form character placement is not possible. Graphics options are also usually limited and slow.

Alternately, a “raster printer” requires that data be sent for every line of dots. This allows advanced systems (like Windows) and applications (word processors and graphics programs) to make sophisticated WYSIWYG graphical documents. One disadvantage is that an enormous amount of data must be sent from the host.

The ITX is very similar to the ETX, and can be used in similar applications. Refer to sections 11.1 and 14.2 for a summary of the programming differences. The ITX comes in two resolutions: 300 DPI and 203 DPI. The 203 DPI will print identical tickets as the ETX family. The 300 DPI printers use the same command set, but there are more dots per inch (DPI). If a ticket designed for 203 DPI is printed on a 300 DPI printer, the fonts and images will be 2/3 as wide and 2/3 as tall.

The ITX has a “Half Resolution” mode option, which can make the printer compatible with 100 DPI and 150 DPI printers. This is an EEPROM setting, which can be enabled as described in the User's Manual.

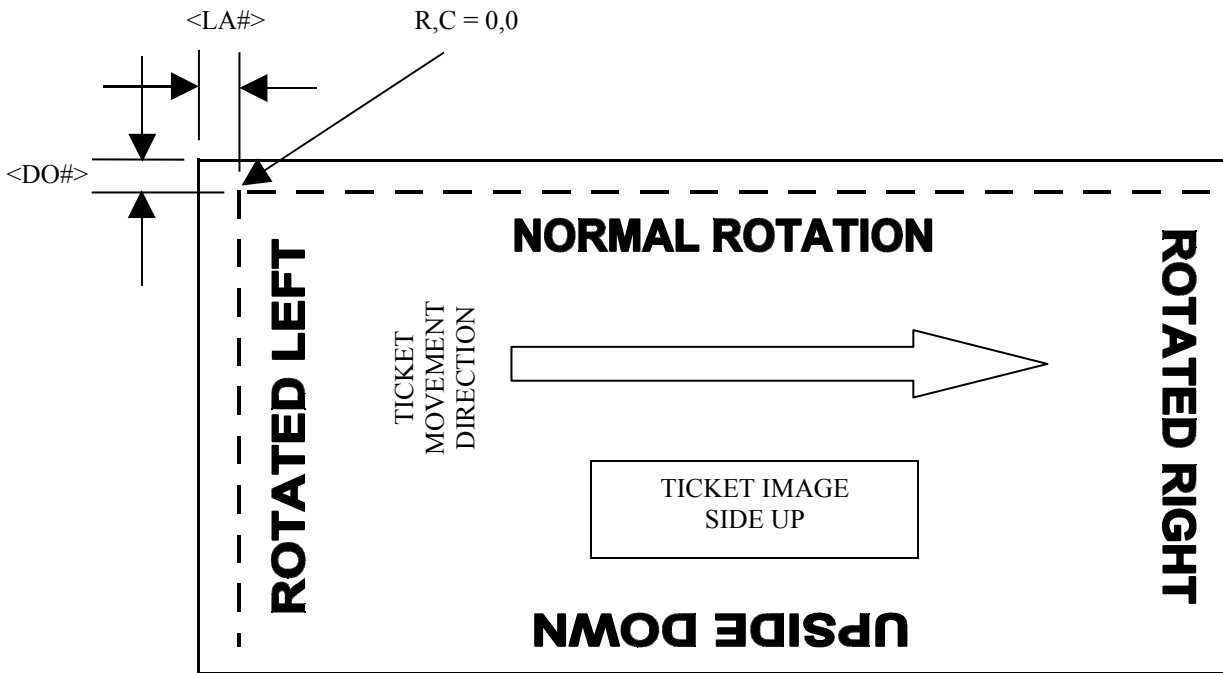
2 SIMPLE TEXT PROGRAMMING

The ITX command set is designed to provide the user with a wide range of choices while keeping the command strings simple and readable. Every command is enclosed in brackets: <RC100,200>. Any text not enclosed in the brackets is considered a printable string of characters. The general programming approach should be:

- 1) Set up parameters such as font selection, rotation, and scale.
- 2) Locate the starting point, which locates the upper left corner of the first letter (or image or barcode). The "upper left" is from the perspective of the letter. Each of the four rotations are printed with respect to that location.
- 3) Send the text to be printed.
- 4) Repeat for all text and graphics, including barcodes and lines.
- 5) Send the print command.

2.1 Rows and Columns.

Each row/column position is the address of a dot position in the print image memory and on the printed ticket. This row/column position system forms a coordinate system for the ticket as shown here:



2.1.1 Setting the Row and Column Location

The <RCr,c> command sets the starting location for the following text string or image. The 'r' value sets the starting Row, counting down from the top. The 'c' value sets the starting Column, counting from the left. A carriage-return (CR) will automatically advance the text so that it places the next string of text directly under the previous one, according to the font size, scale, and rotation. Do not insert a CR between the <RCr,c> command and the first text characters, as it will be interpreted as a new-line command and start the text one line lower than expected. The programmer must keep track of the ending location of each text string and image to make sure that they don't run off the ticket. For example, setting a starting location of 100,100 for either Left or Upside-Down rotations won't leave much room for printing.

2.1.2 Row and Column Limits

The row and column positions are numbered from 0 to their maximum usable limits are a function of the printed ticket length, printer width and stored print image offsets values. At 300 DPI, a 3.25" X 5.5" ticket has 960 rows and 1600 columns. At 203 DPI, the same ticket has 640 rows and about 1100 columns. The Row and Column Offsets will reduce the number of rows and columns available to print. There are no limit traps on the <RCr,c> command; the user must keep track of the location of the various elements on the ticket.

2.1.3 Print Image Offset Values.

These offsets effectively shift the entire print image on the physical ticket. Increasing the Row Offset will move the entire image down. Increasing the Column Offset will move the entire image to the right. On power-up, the Row and Column Offsets will be loaded with the values stored in the EEPROM (see section 2.1.3.3 below). The factory default offset is 16 dots in both directions.

2.1.3.1 Row Offset Command

The <DO#> command will set the number of dots that the image should be moved downward. This setting will be retained for as long as the printer is turned on. To make this offset permanent, use the <SOD> command described below.

2.1.3.2 Column Offset Command

The <LA#> command will set the number of dots that the image should be moved to the right. This setting will be retained for as long as the printer is turned on. To make this offset permanent, use the <SOD> command described below. This command is also called "Label Adjust" as it is used to offset the image when printing peel-off labels.

2.1.3.3 Save Offset Data Command.

The <SOD> command causes the Row and Column Offsets to be stored in the EEPROM. These values will be recalled every time the printer is turned on. The values, described above, can still be overwritten by the Offset Commands.

Note: <SOD> should only be used as a setup command. It *should not* be sent with each ticket. The EEPROM has a long but finite life (approximately 100,000 cycles).

2.1.3.4 Clear Offset Data Command.

The <COD> command resets the Row and Column Offsets to 16 dots each, and stores that value in the EEPROM. These values can still be overwritten by the Offset Commands described above.

Note: <COD> should only be used as a setup command. It *should not* be sent with each ticket. The EEPROM has a long but finite life (approximately 100,000 cycles).

2.2 Font Selection

2.2.1 Factory Installed Fonts

There are several factory installed fonts that can be selected using the <Fn> command, where 'n' is the font number (1-13). The Font Selection command resets the character box size to the default value. If you wish to change the character spacing, you must send the <BSh,w> command after the Font Selection command. In the rear of this manual all of the font images and character sets can be found.

2.2.2 User Installed Fonts

In addition, it is possible to download one or more custom fonts (see section 11). These fonts can be selected using the <fA> command, where 'A' can be 'A' through 'K', to select one of up to 11 User Fonts. You do not need to send character size or box size values, as they were defined during the download process. If you wish to change the character spacing, you must send the <BSh,w> command after the Font Selection command. The ETX-style command is still supported, using the command <FB12,20>. This indicates that the font is 12 dots wide and 20 dots high.

2.3 Rotation

Any font or image can be rotated in any of the four directions using the <NR>, <RR>, <RL>, and <RU> commands for Normal, Right, Left, and Upside-Down, respectively. Barcodes cannot be rotated this way (see section 8).

2.4 Scaling

Each font can be scaled up by an integer amount using the <HW_h,w> command, where 'h' determines the vertical (height) scaling, and 'w' determines the horizontal (width) scaling. The scaling is referenced to the un-rotated perspective (increasing Height will always make letters taller), and is adjusted for other rotations. The maximum scale factor is 32, and the H and W scales do not have to be the same. Fractional Scaling is also possible using the <SD_n> command. The font will be scaled down by n. To obtain a 150% scale size, send <HW3,3><SD2>, which is 3/2. Note that this command must be cleared <SD1>.

2.5 Style – *Bold and Italics*

There are 2 style options that can be added to any font. They can be used singly or in combination, and each has adjustable parameters. The style settings are disabled with every CR, and can be disabled with the <sC> command to permit in-line formatting.

2.5.1 **Bold**

The command <sB#> will enable the bolding of the font. The # value is the amount of bold offset to be used. If no value is passed, the default value of 2 dots will be used. The bold is accomplished by writing the character 4 times at 4 different starting points, using the bold offset value. The character and line spacing is not changed, so bold characters will appear closer together; use the <BSH,w> command to compensate for this if desired. If the offset value is greater than the nominal width of the strokes in the font, white gaps will appear between the successive placements, producing a confused character pattern.

2.5.2 **Italics**

The command <sI#> will enable the italicizing of the font. The # value controls the amount of tilting. The range of values is 1 to 9, where 1 is almost no tilting, and 9 is almost 45° of tilt. If no value is passed, the default value of 5 will be used. The upper-left corner of the font will be shifted to the relative-right by the italicizing process; the lower-left corner of the character will remain in the predicted location. Character and line spacing are not affected.

3 Print Functions

3.1 Simple Printing

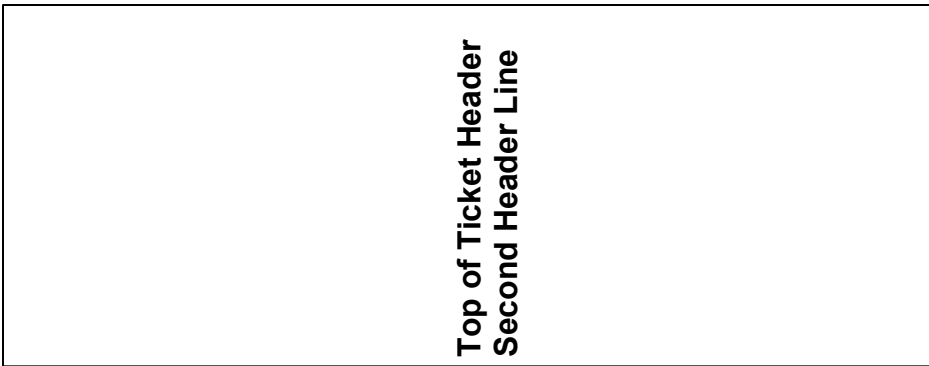
There are several ways that the printer can produce a ticket. The most typical way to get a single ticket is to use the <p> command. This will print one ticket and cut it. If you do not want the ticket to be cut, use the <q> command. These print commands will force the image memory to be cleared after the image is printed. To remain compatible with other systems, the ASCII Form Feed command (0CH) is the same as <p>.

3.2 Simple Example

The following command string illustrates the basic programming sequence:

<F3><HW2,3><RL>	{CR not needed, but OK}
<RC900,100>Top of Ticket Header	{CR locates start of next line}
Second Header Line	{CR not needed, but OK }
<p>	

and will produce a ticket that looks like this:



Note: to print the '<' character, send the character twice '<<'. This will indicate that it is not the beginning of a command string.

3.3 Default Settings.

Default settings for the printer are as follows:

- a. Font type – font 3 – (17 x 31 dots), character box = 20 x 33 dots
- b. Cursor starting position – row 0, column 0
- c. Height – normal (1)
- d. Width – normal (1)
- e. Rotation – no rotation (NR)
- f. Printing length as measured at power on
- g. Bar code size – vertical units (32 dots)
- h. Row Offset – 16 dots
- i. Column Offset – 16 dots
- j. Style – Normal (Bold and Italics off)

Default settings are used by the printer system if only text (no command sequences), are sent to the printer system. The printer system also returns to the default settings for all listed commands (except Row and Column Offset) after completing printing of a ticket (or series of tickets). Note: when using the Windows Language Monitor V2.xx Printer Driver, the Font selection, Rotation, and Height and Width scaling are not returned to the defaults listed above.

3.4 Advanced Printing

3.4.1 Repeated Tickets

If you want to make many copies of the same ticket, or you only need to change the ticket number, you can use the <RE#> command prior to the <p> or <q> command. This will keep the ticket image in memory and print it repeatedly as directed. If one or more Ticket Counts are part of the ticket image (see section 5 below), they will automatically be incremented after each ticket.

3.4.2 Print and Hold Function

If you only need to change a small part of the ticket image (say the bar code), you would use the <h> command (print and hold, with cut) or the <r> command (print and hold, don't cut). These commands do not clear the image memory, and changes can be made according to the commands. In general, the last ticket in the series should use either the <p> or <q> command to print and clear the Image Memory.

Note that whenever the <h> or <r> commands are used, there may be a short pause between tickets. Since the printer is re-using the same Image Memory page, the Image Memory cannot be updated until the printing is completed. However, this pause should not be noticeable unless a significant amount of the image is being changed between tickets.

3.4.3 Ticket Packeting

It is possible to have tickets printed in groups or "packets" of a certain length. For example, you can have 8 tickets printed in 2 sets of 4 by sending the command <M4> prior to the ticket data. There are certain command requirements to assist this process. To set the Packet Size, send <M#>, where # can be any number from 1 to 10. To mark the end of each page, the FormFeed command must be used (0x0C). If <p> or <q> is used, the packeting will be disabled. When all of the tickets have been sent, you must send <MX> to make sure that the last ticket(s) are cut. Thus if you send 7 tickets to be packeted in groups of 4, you will get one packet of 4 tickets and one with 3 tickets. This command will also work with the Repeat command <RE#>.

There are two variations on this command. If you send <M0> (zero), then none of the tickets will be cut. If you send <ML> then the printer will only cut after the last ticket has been printed, as indicated by the <MX> command.

A typical command sequence would look like this:

```
<M2> (Cut every 2nd ticket)
... FGL data...
0x0C (End of 1st page - no cut)
... FGL data...
0x0C (End of 2nd page - cut)
... FGL data...
0x0C (End of 3rd page - no cut)
<MX> (End of ticket data - cut remaining ticket)
```

3.4.4 Clear Buffer Command

The Image Memory is cleared after every ticket, except when the <h> or <r> commands are used. If the previous ticket image was not cleared, you can send the <CB> command, which will force clear the Image Memory. Caution, do not use this command if you do not need to; it takes a few seconds to clear the memory this way.

3.4.5 Cash Drawer Solenoid Driver

The User may send the Pulse Command (1CH) and the printer will generate a 100mS conduction pulse, with a maximum load of 2 Amps. If the printer is actively printing a ticket, the pulse will be postponed until the printer is done with all printing and cutting activities.

It is also possible to turn on the solenoid driver for an indefinite period using the <DRVon> command. The solenoid driver will turn ON after receiving this command when the printer is not printing or cutting. If the command is received when printing is in process, the command request will be delayed until printing is completed.

The command <DRVoff> will turn OFF the solenoid driver immediately. Also, the driver will be turned off automatically if a new print is started.

3.4.6 Ignore Data Command

Certain applications use <I> to tell the printer to ignore any data that follows. <n> disables this option. These commands should not be used in new applications.

3.4.7 Feed and Retract Ticket (Tearbar Only)

In certain applications, if a ticket is not taken by the customer, it is necessary to retract the ticket. The <RT> command will retract the ticket, and the <FT> command will feed the ticket forward.

3.5 Extra Print Commands

To prevent unintentional processing of the form feed characters (as a print command), the printer requires that at least one printable character be received before initiating print with the form feed control character (0CH). Some computer printer drivers attach an extra form feed to the end of the text. This requirement helps to suppress false print commands from being detected. This trap only works with the Form Feed Control character, not <p>, <q>, etc.

4 LINES AND BOXES

The appearance of a printed ticket can be enhanced by printing boxes and lines at various points on the ticket. There are 4 commands that enable line or box drawing. The first is <LTn>, where 'n' is the line thickness in dots. <HXn> and <VXn> will draw a horizontal or vertical line 'n' dots long, starting at the last <RC> location. The lines will expand down or to the right according to the set line thickness. The <BXr,c> command will draw a box 'r' rows high and 'c' columns wide, starting at the last <RC> location. The 4 lines will expand toward the middle of the box, according to the set line thickness. The rotation commands have no effect on these functions.

A diagonal line can be drawn using the <DXr,c> command. The r and c for this command are absolute locations, not relative as with the other box and line commands. The command sequence <RC100,100><DX250,150> will draw a diagonal line from the point 100,100 to the point 250,150. The line thickness is determined by the <LTn> command.

5 TICKET COUNT

Using the <PC> command, a ticket count will be placed according to the set font, scale, rotation, and starting location. The ticket count is always 7 characters, and uses leading spaces for numbers less than 1000000. The ticket count may be placed in up to 4 locations on the ticket. Use the <TC1234567> command to set the initial ticket count, and the <REn> command to print 'n' tickets with automatic incrementing. When the last ticket is printed, the location and other data for the ticket count is cleared, except for the ticket count itself which will continue to increment until overwritten by the next <TC> command.

6 INVERTED CHARACTERS.

Printing of inverted characters (negative images of the original) can be initiated by sending an <EI> (enable inversion) command to the printer system. All following data except bar codes will be inverted until a <DI> (disable inversion) command is sent (see section 14). The inverted mode must be used with caution; the entire ticket could be printed inverted if the <DI> command is forgotten. To improve readability, the printer adds a black border around the inverted characters; therefore, inverted character boxes are slightly larger than normal characters. This size difference must be taken into consideration when positioning characters below inverted characters. It is recommended that a new row/column command be sent for each line that follows an inverted character line; otherwise, lines may appear closer than expected.

7 SHADING OPTIONS

7.1 Background Shading – Characters and Regions

It is possible to create background shading in rectangular regions using the command <PAB>. There are two commands that determine the region to be shaded <ES> and <DS>. The sequence is <RC100,100><ES>Text<DS>. You can put any commands between the <ES> and the <DS>. <ES> sets the first corner of the region to be shaded, and <DS> sets the opposite corner. The previous example will shade all of the “Text”, with the region determined by the box size of the selected font. Or you could automatically shade a box: <RC100,100><ES><BX50,50><DS>. To shade a specific region: <RC100,100><ES><RC200,200><DS>.

7.2 Foreground Shading – Characters Only

It is possible to print font characters with a grey-scaled density using the command <PAF>. This command only works with text.

7.3 Shading Densities and Patterns

There are several shading patterns to choose from using the command <PA#> where # can be from 0 to 25. The complete command sequence should be: <PAB><PA4><RC100,100><ES>Text<DS>.

The gray patterns come in either coarse or fine densities:

Coarse	Fine	Patterns
0 = white	10 = white	20 = vertical lines
1 = very light pattern	11 = very light pattern	21 = horizontal lines
2 = light pattern	12 = light pattern	22 = forward diagonal lines
3 = medium-light pattern	13 = medium-light pattern	23 = backward diagonal lines
4 = medium pattern	14 = medium pattern	24 = square grid
5 = medium-dark pattern	15 = medium-dark pattern	25 = diagonal grid
6 = dark pattern	16 = dark pattern	
7 = unused (white)	17 = unused (white)	
8 = unused (white)	18 = unused (white)	
9 = black	19 = black	

8 BAR CODES

There are several bar-code encoding routines in this printer. Each bar code can be printed in all 4 rotations. The width of a narrow bar can be expanded from 1 dot up to 32 dots. The overall height of the symbol can be set in increments of 8 dots. The ratio of wide to narrow bars is fixed for all encoding except "Interleaved 2 of 5" and "Code 39", which can have either 2:1 or 3:1 ratios. There is an option to automatically print the text equivalent of the barcode beneath the symbol.

The preferred programming sequence is:

```
<RC100,100><X4><BI><UP8>J1234K56780L
```

where:

<RC> sets the starting location,

<Xn> sets the bars expansion,

<BI> enables the printed text equivalent,

<UPn> selects UPC barcode format, Picket-Fence orientation, and a symbol height of "n*8" dots.

The UPC format has a start, middle, and end delimiting character (J, K, L), and will automatically replace the last number with the check character. For all UPC and EAN formats, you must send a trailing '0' which will be replaced by the checksum character. Refer to section 14 on page 26 for all of the barcode formats and options. If the string is not formatted correctly, the bar code will not be printed.

Barcode	Cmd	Start/End	CS?	Example	Notes
UPC	<UP8>	J / K / L	Yes	<UP8>J12345K678900L	Last 0 = CS
EAN-8	<UP8>	J / K / L	Yes	<UP8>J4015K3470L	Last 0 = CS
EAN-13	<EP8>	J / K / L	Yes	<EP8>5J12345K678900L	Leading 5 = Number System. Last 0 = CS
Codabar	<CP8>	a,b,c,d or A,B,C,D	No	<CP8>a123456b upper or lower the same	Numbers and \$: / . + / -
Code 39	<NP8>	* for both	No	<NP8>*CODE39*	Upper-case Alpha and numbers
Interleaved 2 of 5	<FP8>	: for both	No	<FP8>:123456:	Numeric only
Code 128B	<OP8>	^ for both	Yes	<OP8>^Code 128^	94 ASCII chars, don't send CS
Code 128C (1)	<oP8>	^ for both	Yes	<oP8>^123456^	Numeric only, don't send CS

Note: CS= Check Sum.

(1): Code 128C has been updated to allow mixed alpha-numeric data, and odd-length strings.

Rotating the Barcode

The second letter determines the rotation. For compatibility with other printer manufacturers, using the lower case for the first letter will also work. Here is an example for UPC

<UP8> = Picket Fence (normal)

<UL8> = Ladder (rotated Right)

<Up8> = Reverse Picket Fence (Upside Down), or: <RU><uP8>

<U18> = Reverse Ladder (rotated Left), or: <RL><uL8>

Refer to section 14 on page 26 for all of the barcode formats and options.

9 USER FLASH SPACE

The standard printer has 512K bytes of User Flash ROM. Customers can order additional download ROM as a factory installed option. The User Flash Space is used to store both fonts and logos. If a user loads more data than will fit, the download will not be successful, and the font or logo in question will not be accessible. Since the printer can not know in advance how large a font or logo might be, it can not warn the user to prevent this. To assist users, the F0-Status ticket prints the amount of free space available. This value should be retrieved both before and after each download so that the user will know how much space is available. This value can also be retrieved using the <SA08> Status command (see section 16). Large volume customers may request their custom fonts or logos be pre-loaded at the factory.

9.1 *Clearing the User Flash Space*

To clear the entire User Flash Space, use the <CF> command. In order to make the User Flash Space as flexible as possible, without restricting the number or size of logos, it was necessary to treat the Flash ROM as one continuous block. As a result it is not possible to perform a partial erase. You must reload all of the fonts and logos desired. The previous ETX clear buffer command "ESCc" has been replaced with <CF>. The "ESCc" command should not be used.

10 GRAPHICS

Images and shapes can be constructed using the printer's graphics modes. These permit any dot to be turned on or off. Each graphics character consists of one byte of data, and it can be positioned using a row/column command. The first graphics character will be printed starting at the row and column specified in the row/column command, and successive graphics characters will be printed in successive locations.

Graphics can either be printed directly, or permanently stored in ROM for repeated use. Graphics can be sent in either a "dot-matrix printer" compatible format, or in PCX format. In addition, the "dot-matrix" format can be sent as binary data, or as ASCII-hex characters which will be converted to binary data.

10.1 "Dot-matrix Printer" Graphics

In this format, each 8-bit binary value represents 8 dots aligned vertically, with each value adding another "column" of 8 bits next to the previous. Each 8-bit tall "row" of graphics must be preceded by a new <RC> command. This graphics format may be rotated and scaled, but each new <RC> location must be adjusted by the programmer for the orientation and scaling.

The graphics command sequence is:

```
<RC100,100><G10>[first row, 10 bytes of data]
<RC108,100><G10>[second row, 10 bytes of data]
```

Some computers cannot send the full range of 8-bit data required for graphics. The printer also has been configured to receive dot data bytes as ASCII characters, as well as, straight decimal interpretations of each column. For example, a column in which every other dot is 1 would be represented by the byte 01010101. This is equivalent to a straight decimal value of 85, and a byte with the value of 85 would be sent to the printer system in normal graphics mode. This could be accomplished in Basic with a print chr\$(85) statement. In ASCII graphics mode, the byte would be split into two ASCII bytes which represent the hex value (55H) of the byte, and would be sent to the printer system as two bytes of ASCII 5s. In Basic, this could be done with a print "55" command. To select ASCII graphics mode, a small g is used in the graphics select command instead of a capital G. The ASCII graphics select command must specify the number of bytes to follow <g#>. Note that the number of bytes is twice as large, as two ASCII bytes are needed for each byte of graphic data. As in the example above:

```
<RC100,100><g20>[first row, 20 ASCII hex values, 10 bytes of graphic data]
<RC108,100><g20>[second row, 20 ASCII hex values, 10 bytes of graphic data]
```

10.2 PCX Graphics

The PCX file format is supported for graphics data input. Permitting PCX files to be sent directly to the printer greatly enhances its graphics capability. The PCX graphics file format is commonly used. As such, a vast array of "Clip Art" and drawing software exists for PCX files. The file structure has built in data compression. This feature is preserved by the printer system. The reduced file size minimizes the amount of data needed to be sent to the printer. The file is handled as a graphics block rather than a series of raster data segments. The printer system uses this attribute to permit image rotation (all four orientations) as well as scaling.

PCX is an industry standard graphics file format. When a PCX file is created for this printer it must be in the 2-color (B&W) single bit format. Any other format will cause unpredictable results. Prior to sending the PCX file, you must send two commands to set up the printer:

```
<pcx><Gnnn>[PCX File]
```

where:

```
<pcx> readies the printer to accept PCX format graphics,
<Gnnn> indicates that there will be 'nnn' bytes in the file,
[PCX File] is the actual file data, without brackets.
```

Example:

```
<RC100,100><NR><HW2,2><pcx><G1383>[pcx graphic data]<p>
```

will print the PCX graphic image scaled up by 2.

10.3 Storing Graphics in Nonvolatile Memory

By sending the ESC character (1BH), the programmer tells the printer to save all of the commands that follow in the User Flash ROM. A second ESC tells the printer that the download is complete. This set of data commands is considered one "logo" and can be recalled later when printing a ticket. This data is not written to the image memory. Each new download is tagged with a sequential logo number. Graphics and text can be combined to make a more complex logo, so long as the restrictions described below are followed.

PCX graphics have built in data compression that is preserved when stored as a logo. Thus a 1" x 1" 300 DPI PCX graphic image will require approximately 5 KBytes of ROM, assuming a compression efficiency of about 2.3X. By using the scaling function, you can create a 2" X 2" image from that 5Kbytes of data, covering 45KBytes of image area.

Note that prior to V1.20, PCX files were limited to 32Kbytes. This limit is now removed.

For example, after you have confirmed that a PCX graphic file prints correctly by sending the command sequence:

```
<pcx><Gnnn> data ... data <p>
```

you can load it into flash memory by using the following sequence:

```
ESC<pcx><Gnnn> data ... data ESC
```

Don't include the <p> command.

10.4 Cautions for Storing Logos

A stored "logo" can consist of any legal command sequence between the ESC commands, including text, barcodes, and graphics, with the following restrictions:

- A logo can not call another logo using <LD#> or <LO#>.
- A logo is assumed to be at Normal rotation and a scale of 1. Since a logo can be scaled and rotated when it is recalled, scale or rotation commands stored within a logo will produce unpredictable results.
- A logo can not contain any print commands, such as <p>. Nor is the repeat command <RE#> permitted.

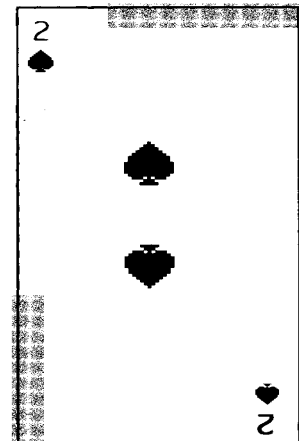
10.5 Recalling Stored Logos

By using the <LDn> command, the 'nth' saved graphics data set will be placed on the ticket according to the most recent rotation, and scale commands. You must use the <SPr,c> command to set the starting point for any logo, because there are <RC> commands embedded in the logo.

Similarly, the <LOn> command selects any logos pre-loaded at the factory. If you have the standard firmware, the 4 card suits (Spades, Clubs, etc) are pre-loaded into the 4 factory logo slots. They are 16X16 dots. The fifth factory logo is the PA Logo. If you try to select a logo that doesn't exist, the last User Logo will be selected. If no User Logos exist in Flash, one of the factory logos will be selected. This should help with debugging.

Example:

```
Using the factory logos, print a 2-of-Spades by the following sequence:
<F6><RR><RC75,900>2          {the lower right 2}
<RL><RC450,50>2             {the upper left 2}
<HW3,3><RL><SP460,110><L01>   {the upper left Spade}
<RR><SP65,840><L01>         {the lower right Spade }
<HW6,6><RL><SP300,350><L01>  {the upper middle Spade, larger}
<RR><SP210,650><L01>       {the lower middle Spade, larger}
<RC50,30><LT3><BX440,900>   {the box which outlines the card}
<p>                          {print it}
```



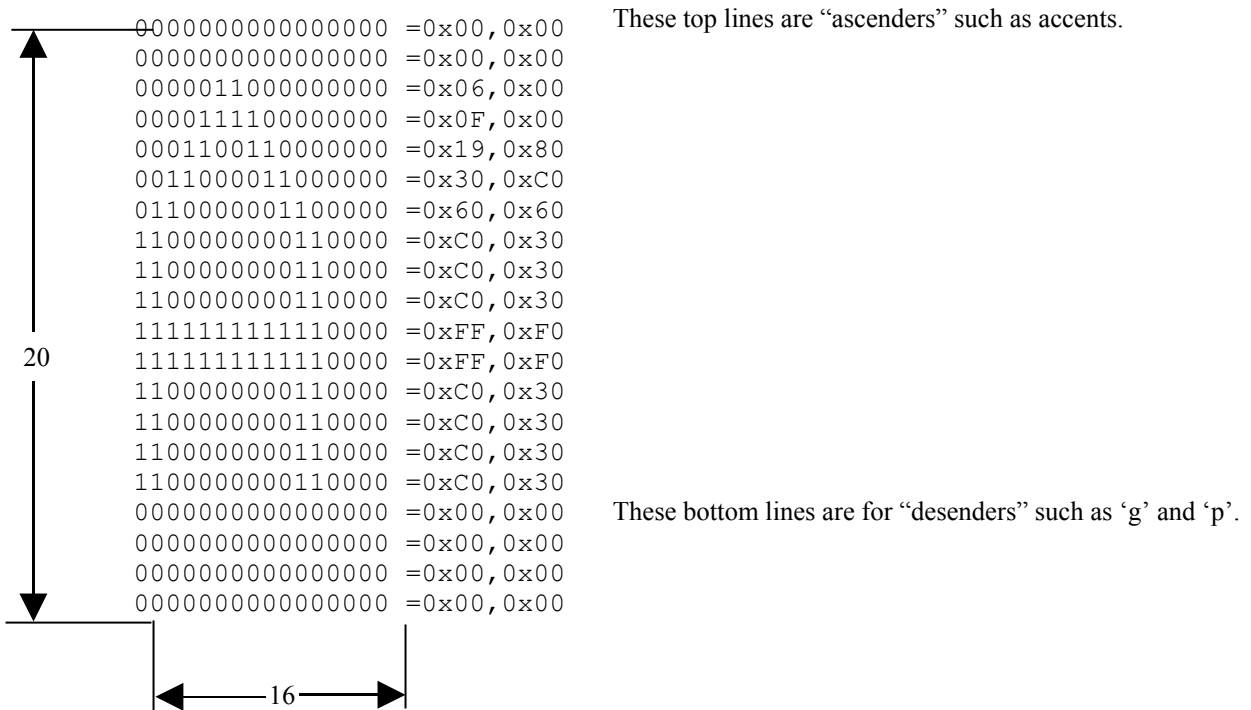
11 DOWNLOADING USER FONTS

It is possible to download custom fonts into the User Flash ROM. Users can create new font faces using various techniques, so long as the resulting data is binary raster, byte-formatted, and uncompressed. The range of characters is from 20H (SPACE) to FEH. Your font can have as many or as few characters as desired (up to 222), but there can not be any gaps. If you have missing characters, you must fill them with blanks (spaces).

The command <DF cw, ch, bw, bh> will tell the printer that the next set of data will be font raster images. The 'cw' and 'ch' parameters define the size of the character in dots. The dot data must be the same size as defined by the 'w' and 'h'. The 'bw' and 'bh' parameters define the size of the box in dots, thus defining the default spacing between characters. The box size must be greater than the character size.

For example <DF12,20,14,24> indicates that a font that is 12 dots wide and 20 dots high is to be downloaded. Each 12-bit wide row will be sent as 2 bytes, with the right hand bits ignored.

Here is a representation of a 12X20 character 'A':



Note: 16 bits wide is because bytes are sent in increments of 8.

In addition, the box size is declared to be 14 dots wide and 24 dots tall, thus setting the inter-character and inter-line spacing. It is not necessary to pad the character itself to create the desired spacing. This wastes memory space and processing time. Immediately following the <DF...> command, there must be an ESC to indicate the start of the font data. Since 12 dots requires 2 bytes to be stored, the printer expects 40 bytes to be sent for each character in the font. After receiving one character worth of bytes, the printer counts the character and tests for a trailing ESC.

<DF12,20,14,24> ESC {font data} ESC

Each new font is logged sequentially, and can be retrieved using the <fA> command (section 2.2.2). Up to 11 User Fonts can be downloaded, provided there is enough space in the User Flash ROM. If the <fA> command identifies a font that does not exist, then the last User Font will be selected. If no User Fonts exist, the Diagnostic font will be selected. These choices should assist with debugging.

11.1 Differences Between ITX and ETX

The ITX stores all of its fonts in 16-bit wide words, which permits faster imaging of the larger fonts typically used at 300 DPI but does not slow the system when using smaller fonts. The data is still sent in bytes; the printer converts and stores the data in words. The following items have changed for the ITX:

- The ETX requires that exactly 91 characters be sent. The ITX can accept more or fewer characters, up to a maximum of 222.
- The ETX is limited to 4 User Fonts. The ITX can accept up to 11 User Fonts, provided enough space exists in the User Flash ROM.
- The ETX command sequence of “ESC r #>{font data}ESC” is not valid on the ITX. Data sent using this command will be ignored. The ITX will accept the same font data file, but with a different command sequence “<DF c_w , c_h , b_w , b_h >ESC{font data}ESC”.

12 PRINT AREA


The printer automatically measures the ticket length, in dot column increments, at power up. To make this measurement the ticket is moved outward and then returned to its initial position. The measured distance between the registration marks, on two successive tickets, becomes the print length. From a programming prospective, data can be printed along the entire measured print length (and width). This, however, must be considered with the limitation that printing too near a perforation or the ticket edge is not recommended. The default print image offset values, discussed above, provide margins for this reason. These margins can be reduced (or expanded), by program commands. The measured ticket length cannot be overwritten by commands.

12.1 *Overprinting Precautions*

Special care is required when using character rotation and expansion commands to avoid overlapping of printed characters. Keep in mind that all characters are justified to the top left corner in their boxes, according to their rotation. For example, a rotated left character that starts at the bottom left corner of the ticket builds up and to the right on the ticket. Expanded characters build into adjacent rows and columns. When specifying such characters, make certain that the characters are not started in a row or column occupied by other characters or images, and that the characters will not build into spaces occupied by other characters. For example, if a 5X8 character is expanded by a factor of 2, it will occupy 16 dots vertically from its starting point instead of the normal 8; therefore, no character should be positioned less than 16 dot rows below the expanded character. After processing an expanded character, the printer will place the next character in the correct position to prevent overprinting automatically. The printer remembers only the parameters set up for the previous printed character. If a character sequence starting with normal characters and ending with expanded characters is printed across the ticket, a return will position the next character below the printed line as though the printed line started with an expanded character. If such positioning is incorrect, a new row/column command must be sent to position the next character at the desired location on the ticket.


13 SAMPLE TICKET

This ticket can be printed by pressing [F0] followed by [F1].

ITX-3003  HIGH SPEED TICKET PRINTER

Rotated Text and Logos

Graphics & Logos
Multiple Fonts
Box & Line Draw
User Downloads

8 Bar Codes

4015 3476

Practical Automation
45 Woodmont Road
Milford CT 06460
203-882-5640

INVERSE PRINTING

Ticket Count
#7654322

www.practicalautomation.com

13.1 Sample Ticket Command Sequence

This is the code for the internal "Splash Ticket". This command sequence was created with a simple text editor. The extra spaces caused by CRs do not affect the final image, and make this sequence more readable. This ticket was designed for a 3.25" X 5.5" ticket at 300 DPI.

```
<F12><HW2,2><RC0,330>ITX-3003
<F6><HW1,1><RC150,320>HIGH SPEED TICKET PRINTER
```

```
<LT5><RC250,250><BX300,640>
<F3><HW2,2><RC270,270>Graphics & Logos
Multiple Fonts
Box & Line Draw
User Downloads
```

```
<RC650,250><EI>INVERSE PRINTING<DI>
```

```
<F3><HW2,2><RC320,980>8 Bar Codes
<RC380,1000><UP18><X5><BI>J4015K3470L
```

```
<TC7654321>
<F3><HW2,2><RC680,1000>Ticket Count
#<PC>
```

```
<F3><HW1,1><RR><RC250,1600>Practical Automation
45 Woodmont Road
Milford CT 06460
203-882-5640
```

```
<F10><RU><HW1,1><RC900,1170>www.practicalautomation.com
<F11><RL><HW1,1><RC850,150>Rotated Text and Logos
```

```
<NR>
<SP50,150><HW2,2><LO1>
<SP50,200><HW2,2><LO2>
<SP100,150><HW2,2><LO3>
<SP100,200><HW2,2><LO4>
```

```
<SP25,1200><HW4,4><LO5>
```

```
<SP50,1500><HW2,2><NR><LO1>
<SP50,1582><HW2,2><RR><LO1>
<SP132,1500><HW2,2><RL><LO1>
<SP132,1582><HW2,2><RU><LO1>
```

```
<p>
```

14 COMMAND SUMMARY TABLE

Command	Format	Function	Comment
Cursor Control Commands			
Row/ Column	<RCy,x>	Sets row/column print position cursor to row y, column x	This should be sent after all selection commands and before any text or graphics,
Carriage Return	0DH	Sets the cursor to align the next text under the previous line.	Font Box Size sets the line spacing
Rotation	<NR>	No rotation of following characters (Default)	Remains active until new rotation command is received: default = <NR>.
	<RR>	Following characters rotated right (+90 deg.)	
	<RU>	Following characters rotated upside down (+180 deg.)	
	<RL>	Following characters rotated left(-90 deg.)	
Height/width	<HWx,y>	Multiplies character or graphics Height by x and Width by y (x:y =32max.)	Default = 1,1. Use <HW 1,1> to return to normal size after expansion
Font Selection and Control			
Font Face	<F1>	Selects LCD	(5x7 dots) Box 7x8
	<F2>	Selects LCD Bold	(8x16 dots) Box 10x18
	<F3>	Selects medium OCRB (Default)	(17x31 dots) Box 20x33
	<F4>	Selects small OCRA	(5x9 dots) Box 7x11
	<F5>	Same as Font 2	(8x16 dots) Box 10x18
	<F6>	Selects large OCRB	(30x52 dots) Box 34x56
	<F7>	Selects medium OCRA	(17x31 dots) Box 20x33
	<F8>	Selects Courier	(18x30 dots) Box 30x30
	<F9>	Selects small OCRB	(13x20 dots) Box 13x22
	<F10>	Selects Bold Prestige	(25x41 dots) Box 28x41
	<F11>	Selects Script	(25x49 dots) Box 26x49
	<F12>	Selects Orator	(46x79 dots) Box 47x91
	<F13>	Selects Courier	(20x40 dots) Box 20x42
User Font Face	<fA>	A = A to K, 11 User Fonts	Default spacing
User Font Face (ETX style)	<Fa W,H>	a = A to K, 11 User Fonts W = width, H = height	W & H should be same as declared during download
Box Size	<BSx,y>	Sets size of area in which printed characters sit to x dots wide, y dots high.	When Font is Selected, Box Size is set as above. Send <BSx,y> after Font Select.
Scale Down	<SDn>	When combined with <HWh,w>, permits fractional scaling by dividing the font size by this value. Ex 150% = <HW3,3><SD2>	Fonts only. Refer to Section 2.4 Does not clear automatically
Enable inversion	<EI>	Causes following characters to be printed in inverted (negative) mode	Be sure to send <DI> command when inversion is complete. (See section 6)
Disable inversion	<DI>	Terminates character inversion mode.	Default Mode (See sec. 6)

Command	Format	Function	Comment
Line and Box Drawing			
Draw box	<BXr,c>	Causes printer system to draw box r dots high and c dots wide starting at location specified by preceding row/column command	Refer to Section 4
Draw horizontal line	<HXc>	Causes printer system to draw horizontal line c columns wide starting at location specified by preceding row/column command	Refer to Section 4
Draw vertical line	<VXr>	Causes printer system to draw vertical line r rows high starting at location specified by preceding row/column command	Refer to Section 4
Draw diagonal line	<DXr,c>	Causes printer system to draw diagonal line at the location specified by preceding row/column command and ending at the location indicated by r and c.	Refer to Section 4
Line thickness	<LT#>	Sets Line Width to # dots	Reverts to 1 dot Refer to Section 4

Command	Format	Function	Comment
Graphics			
Graphics	<G#>byte 1, byte 2 ... byte#	Sets printer system to graphics mode and followed by graphics bytes to be printed	Refer to Section 10
	<G>byte 1, byte 2 ... byte 7	If no #, only 7 bytes	
	<g#>byte1a, byte1b, ...	Sets printer system to ASCII graphics mode and sends graphic byte-pairs to be printed.	
PCX File Lead-in	<pcx>	Precedes the PCX file data.	Refer to Section 10.2

Print Functions			
Print (Cut)	OCh or <p> Standard ASCII FormFeed Command	Causes printer to print and cut the ticket. The image memory is cleared.	Refer to Section 3.1
Print (No Cut)	1Dh or <q>	Causes printer to print the ticket, without cutting. The image memory is cleared.	Refer to Section 3.1
Print & Hold (Cut)	<h>	Causes printer to print and cut the ticket. The image memory is not cleared.	Refer to Section 3.4
Print & Hold (No Cut)	<r>	Causes printer to print the ticket without cutting. The image memory is not cleared.	Refer to Section 3.4
Ticket Packeting	<Mn> <ML> <MX>	Causes tickets to be cut in groups of 'n'.	Refer to Section 3.4

Ticket Count			
Load ticket count	<TC#####>	Pre-loads 7 digit ticket count into printer system	Must contain 7 digits; loaded count is number of current ticket
Print ticket count	<PC>	Causes printer system to print ticket count on ticket	Must be sent for first ticket. Max of 4 locations. Refer to Section 5
Repeat	<RE#>	Allows printing of many copies of the same ticket without re-transmitting	# valid from 1 to 60,000 Ticket Count is incremented for each ticket.

Command	Format	Function	Comment
Bar Code			
Bar Code Select	<AB#> see below	A=Format, B= Orientation, #=Height (divided by 8)	Refer to section 8
Codabar	<CP#> <Cp#> <CL#> <Cl#>	Codabar – Picket Fence Codabar – Upside Down Codabar – Ladder-Right Codabar – Ladder-Left	Or <RU><cP#> Or <RL><cL#>
Code 128B	<OP#> <Op#> <OL#> <Ol#>	Code 128B – Picket Fence Code 128B – Upside Down Code 128B – Ladder-Right Code 128B – Ladder-Left	Alpha Numeric
Code 128C	<oP#> <op#> <oL#> <ol#>	Code 128C – Picket Fence Code 128C – Upside Down Code 128C – Ladder-Right Code 128C – Ladder-Left	Compact Numeric Only
Code 39 (3 of 9) 2:1 Ratio	<NP#> <Np#> <NL#> <Nl#>	Code 39 – Picket Fence Code 39 – Upside Down Code 39 – Ladder-Right Code 39 – Ladder-Left	Bar Ratio is 2:1 Or <RU><nP#> Or <RL><nL#>
Code 39 (3 of 9) 3:1 Ratio	<NXP#> <NXp#> <NXL#> <NXl#>	Code 39 – Picket Fence Code 39 – Upside Down Code 39 – Ladder-Right Code 39 – Ladder-Left	Bar Ratio is 3:1 Or <RU><nP#> Or <RL><nL#>
EAN13	<EP#> <Ep#> <EL#> <El#>	EAN13 – Picket Fence EAN13 – Upside Down EAN13 – Ladder-Right EAN13 – Ladder-Left	Or <RU><eP#> Or <RL><eL#>
UPC and EAN8	<UP#> <Up#> <UL#> <Ul#>	UPC – Picket Fence UPC – Upside Down UPC – Ladder-Right UPC – Ladder-Left	Or <RU><uP#> Or <RL><uL#>
Interleaved 2 of 5 2:1 Ratio	<FP#> <Fp#> <FL#> <Fl#>	I:2of5 – Picket Fence I:2of5 – Upside Down I:2of5 – Ladder-Right I:2of5 – Ladder-Left	Bar Ratio is 2:1 Or <RU><fP#> Or <RL><fL#>
Interleaved 2 of 5 3:1 Ratio	<FXP#> <FXp#> <FXL#> <FXl#>	I:2of5 – Picket Fence I:2of5 – Upside Down I:2of5 – Ladder-Right I:2of5 – Ladder-Left	Bar Ratio is 3:1 Or <RU><fP#> Or <RL><fL#>
Bar Code Expanded	<X#>	Expands width of narrow bar from 1 dot to # dots	Default = 1 dot
Bar Code Interpretation	<BI>	Causes bar code interpretation line to be printed under bar code	Active only for bar code that immediately follows

Status			
Refer to Section 16	<S1>	Causes printer to send 1 byte Status message	Refer to section 19.1.3
	<S2>	Causes printer to send 7 digit ticket count and firmware level of printer	
	<S3>	Causes acknowledge status byte to be sent by printer only after last ticket of run has been printed	
	<S5>	Disables all status except Xon/Xoff	
	<Sz>	Causes printer to return single ASCII status byte	
	<S6>	ASCII Status (adds 0x30 to all)	
	<S7>	Flash Space Available	
	<S8>	Partial ASCII Status	
	<SS>	ATX style Short Status	Refer to section 19.3.2
	<SN>	ATX style Normal Status	
	<SE>	ATX Style Extended Status	
	<SC>	ATX style Complete Status	
	<SA#>	Returns the specific Status Field value, as selected by #.	
EEPROM Config			
	ESC IXN	How many fields?	Refer to section 17.
	ESC IXCmm	Current setting of field 'mm'	
	ESC IXOmm,nn	Description of field 'mm', option 'nn'	
	ESC IXSmm,nn	Select option 'nn' in field 'mm'	
	ESC IXFnnnn	Set Form Length: '0475' = 4.75"	
	ESC IXR	Reset EEPROM to Factory Defaults	
	ESC IXL	Return Odometer Values	
	ESC IXQ	Return Error Queue Values	
Diagnostic Tickets			
	ESC IXD00	Print System Status ticket	Refer to section 18.
	ESC IXD01	Print Splash Ticket (standard/short)	
	ESC IXD02	Print Long Splash Ticket (print standard if only one available)	
	ESC IXD03	Print Current Print Energy Test page	
	ESC IXD04	Print All Print Energy levels	
	ESC IXD05	Print Odometer and Error Queue	
	ESC IXD06	Advance and Cut	

Command	Format	Function	Comment
Flash Reprogramming Mode			
Enter Flash RP Mode	ESC * FR	Causes the printer to enter Flash Memory Reprogramming Mode	Refer to section 20
User Downloading and Retrieval			
Download Font	<DF cw, ch, bw, bh>ESC data... ... data ESC	Installs User Font in User Memory. Automatically counts the number of characters.	Defines character and box size. Different than ETX. Refer to Section 11.
Download Graphics	ESC <RC0,0><G#>data... <RC8,0><G#> data... ... data ESC	Installs User Graphics into User Memory	Assumes scale = 1 and no rotation. Refer to Section 10.3.
Download PCX Graphics	ESC <pcx><G#>data... ... data ESC	Installs User PCX Graphics into User Memory	Refer to Section 10.3.
Download Text	ESC <RC0,0><F#>text... ... text ESC	Installs User Text into User Memory as a User Logo	Refer to Section 10.3.
Select user Font	<FAw,h>	Select User Font A-K, and character box size w,h.	Refer to Section 2.2.2.
Set Logo Starting Point and place the Logo	<SPr,c>	Set the starting location of the selected logo.	Set scale and rotation prior to command. Refer to Section 10.5.
Select User Logo	<LD#>	Draws the User Stored Logo at the Starting Point	Refer to Section 10.5.
Select Factory Logo	<LO#>	Draws the Factory Stored Logo (1-4) at the Starting Point	Refer to Section 10.5.
Clear User Memory	<CF>	Clears all User Logos and Fonts	Refer to Section 9.1.
Miscellaneous			
Clear buffer	<CB>	Clears ticket buffer and sets all parameters to default conditions	NOTE: Not normally needed; the printer clears buffer automatically.
Clear Doc Count	<CD>	Clears the number of tickets printed.	Differs from Ticket Count, which is set by the user.
Dot offset	<DO#>	Changes row at which first point on ticket is printed by # dots	Dot offset set to default # of dots on power-up
Row Offset	<RO#>	Changes row at which first point on ticket is printed by # byte positions (8 dots)	Same as <DO#>, except in multiples of 8, legacy command.
Label adjust	<LA#>	Changes columns at which first point on ticket is printed by # dots	Label adjust set to 16 dots on power-up
Save offset data	<SOD>	Stores offset values in EEPROM	Refer to Section 2.1.3.3
Clear offset data	<COD>	Sets offsets to default values and stores these values in EEPROM	Refer to Section 2.1.3.4
Solenoid Driver Pulse	1CH	Pulses a 2A Max open drain cash drawer driver.	Refer to Section 3.4.5
Solenoid Driver ON	<DRVon>	Turns ON the solenoid driver if/after printer is not printing.	Driver will turn OFF if new print is started.
Solenoid Driver OFF	<DRVoff>	Turns OFF the solenoid driver.	
Feed Ticket	<FT> or <FT##>	Advances the ticket without cutting	If included, ## is the percentage of the ticket length to be advanced.
Retract Ticket	<RT>	Retract the ticket and align on mark	
Set Print Energy Level	<lve#>	Sets the print energy +/-5	

14.1 Windows Driver Support Commands

These commands are not to be used by the FGL language programmer. They are only used in support of the Windows Printer Driver, used with the FGL emulation firmware ITX 2000/3000 printers.

ETX Command	Function	Alternate Method or Explanation
<RX#>	Set Row Only (Windows)	Use <RC>
<RY#>	Set Column Only (Windows)	Use <RC>
<LM>	Landscape Mode (Windows)	Use Rotation Commands
<PM>	Portrait Mode (Windows)	Use Rotation Commands

14.2 Unsupported Commands

The following commands were added to the ETX/LTX family for various reasons. Some commands were added to work with the earlier Windows driver. These commands are not supported in the ITX. An alternate method is provided.

ETX Command	Function	Alternate Method or Explanation
<R+#>	Registration Adjust Forward	Align Sensors
<R-#>	Registration Adjust Backwards	Align Sensors
<px> and <hx>	Partial Cut	Cutter is Full Cut only
<CM>	Cut Mode (Windows)	Use Correct Print Commands <p> or <q> etc.
<t> and <n>	Transparent Mode	Alternate Serial Port not Available
<ME> and <MD>	Messaging Features	Alternate Serial Port not Available
<Sp#>	Parallel Status (ETX style)	Replaced by ATX style Status (see section 19.2.4)
ESCc	Clear User Flash	<CF> see section 3.4.4
ESC r7>	DL Font	<DF> see section 11

14.3 Illegal Commands.

In general, improperly formed commands will be ignored. In some cases, an improper numeric value will be evaluated as '0'. There are limits on the values for scaling and font and logo selections, but not for Row/Column values. In most cases, printing of preliminary tickets will be sufficient to test the layout of a ticket. However, programmers should keep track of calculated location values to make sure that the values do not go out of bounds.

15 Serial Port Configuration

In addition to being able to select the baud rate and parity settings (see User's Manual), there are two modes of "data flow control" available. When DIP switch #7 is set to ON, the flow control is done using XON and XOFF. When this switch is OFF, the flow control is done using the Busy hardware control signal.

Note that if you are using the Language Monitor driver, you must configure the Host PC Flow Control to "None".

15.1 Serial Port Flow Control Summary

Control Feature	Data Flow Control Selection	
	XON/XOFF	Busy Hardware
Interface Busy Signal	Always not-Busy	Flow Control
XON/XOFF Control Characters	Sent for Flow Control	Not Sent
ACK Control Character	Sent after tickets	Not Sent
Status – Normal Operation	Replies to Status Requests.	Replies to Status Requests
Status – Error Condition	Send Error Byte when detected, followed by XOFF. Reply to Status Requests.	Sets signal to Busy. Reply to Status Requests if Host ignores the Busy.

15.2 XON/XOFF Data Flow Control

XON (13H) and XOFF (11H) are ASCII defined control characters. XON tells the host that the printer is ready to receive data. XOFF tells the host that the printer is not ready to receive data. If the printer encounters an error, runs out of paper, or is almost out of room in its Receive Buffer, the printer will send an XOFF command. The printer keeps a small amount of reserve buffer space which allows it to accept some extra data that might be sent before the host processes the XOFF command. If the buffer overflows, all data is lost, and a non-recoverable error is flagged.

With XON/XOFF enabled, the printer will also send an ACK (06H) after every ticket (or after a group of tickets if <S3> is sent).

After the host has recognized the XOFF and stopped the transmission of printing data, the host may still send status commands and the printer will reply appropriately, according to the command. If normal printer operation has stopped (due to an error, out-of-paper, etc), status commands will continue to be processed, even if the Receive Buffer overflows.

The XON/XOFF mode must be used if the printer is connected to the host with a "null-modem" 3-wire cable. Note that if a 9-wire cable is used in XON/XOFF mode, the Busy control line (usually connected to the Host's CTS input) will not change; it will always remain in the not-Busy state.

15.3 Busy Hardware Flow Control

Busy is a hardware signal (pin 8) in the serial cable. It is controlled by the printer and read by the host, usually as the CTS input. When the printer sets the Busy line to the Busy state, the host is not supposed to send data to the printer. Usually this is handled by the host's Serial Port hardware. If the printer encounters an error, runs out of paper, or is almost out of room in its Receive Buffer, the printer will set Busy. The printer will not respond to Status commands.

The printer keeps a small amount of reserve buffer space, which allows it to accept some extra data bytes that might be sent before the host processes the Busy signal. If the buffer overflows, all data is lost, and a non-recoverable error is flagged.

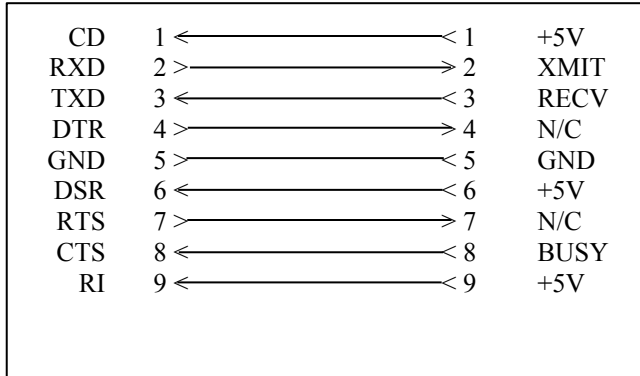
If Busy control is selected, the printer will still respond to any Status Commands. If there is an error and the printer sets the Busy, the host may be able to send status requests by ignoring or overriding the Busy control. Refer to Section 16.

The Busy flow control will only work with the Busy line connected and monitored by the host. It will not work with a null-modem 3-wire cable.

15.4 RS-232 Serial Cable Connections

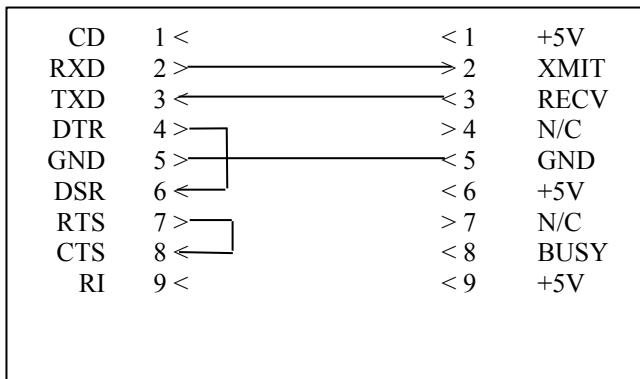
9-Pin Connector – 9-wires

PC ITX/ITL PRINTER



9-Pin Connector – Null-Modem (3-wires)

PC ITX/ITL PRINTER



16 Printer Status - Error Condition Flow Control Exceptions

16.1 Data Flow Control:

During the receipt of data, by the forward data channel, a flow control method is used to insure that the host computer does not send data faster than the printer can receive it. The method of flow control is a function of the type of data interface. The Parallel Data Interface has a hardware character-by-character synchronous Busy signal. For the Serial Data Interface the flow control can be either a hardware Busy signal or a reverse data channel XON/XOFF control character. The host computer must observe this data flow control to achieve reliable data transfer.

16.2 Receive Data Buffer Overflow:

If the printer's Receive Buffer becomes full the flow control is asserted to stop the input flow of data. If the host continues to send data (for more than a few characters after this flow control stop condition has been asserted - see section 19.1.5), a Receive Data Buffer overflow error will occur. This is a non-recoverable error. Any ticket in process will be completed and then all data is lost.

16.3 Error Condition Flow Control Exceptions:

For the Parallel Interface, exceptions to flow control procedures are not required to obtain status information after an error has occurred. This is true because an explicit forward data channel command is not required for this interface in order to read status from the printer (19.2.5.1).

The Serial Interface requires different treatment because it does require a forward data channel command to be sent and also because the forward data channel will be Busy during an Error condition. To overcome these apparent conflicts the host will need to distinguish a data flow Busy condition from an error Busy condition by using a time-out method. The printer will, under these error conditions, continue to process status commands sent by the host computer.

16.4 Error States:

The printer has three states relating to error conditions: Normal (no errors), Operator Recoverable and Non-Recoverable Errors. In each of these states, and for the Parallel or Serial Data interfaces, the methods for reading status information are different. These error states and the status method differences are discussed below.

16.4.1 Normal Operation

During the normal operation state the printer receives data, over the data interface, buffers that data and then processes it to produce the printed output or to reply with status data. During Normal Operation there are no special rules for reading status information. For the Serial Interface these status methods and command are outlined in section (19.1) and for the Parallel Interface in section (19.2).

16.4.2 Operator Recoverable Error

There are error conditions that are considered temporary or recoverable. Some of these are: out-of-paper, head lever up, printer manually deselected, etc. These errors are typically corrected by local operator intervention. Usually a simple action such as re-entering paper, closing the printhead lever or pressing the Front Panel's Select button will restore the printer to normal operation. If, however, the printer is deployed in a remote location this recoverable distinction becomes "gray" and these errors may also be viewed as a non-recoverable errors.

When these conditions occur the printer can no longer continue to receive data. It signals the host computer to stop sending data by asserting the flow control to halt the data flow. For the Parallel Interface, the printer's status information may be read as discussed in section (19.2.5.1). For the Serial Interface special rules need to be observed, refer to section (19.1.5).

16.4.3 Non-Recoverable Error

If the printer has encountered a more serious problem, for example a cutter or paper jam, this is considered a non-recoverable error. One of the primary characteristics of a non-recoverable error is that the printer will require a power cycle, after the error has been corrected, to restore normal operation. As noted, when these conditions occur the printer can no longer continue to receive data. It signals the host computer to stop sending data by asserting the flow control to halt the data flow. For the Parallel Interface, the printer's status information may be read as discussed in section (19.2.5.1). For the Serial Interface special rules need to be observed, refer to section (19.1.5).

17 EEPROM Host Configuration

Effective with version ITX V1.20 FGL firmware (all ITL), and ITX-G V1.10 firmware, it is possible to change the EEPROM settings from the Host. If you are using the UPPP printer driver, then you can change most of the settings from the Printer Properties panel as well.

Practical Automation provides a utility for changing the EEPROM settings. In addition, these programmatic commands can be added to your application. The specific controls and settings are described in the EEPROM section of the Printer's User's Guide.

The printer must be Online and Ready with tickets loaded for these features to work.

17.1 Programming Overview

Each command should be sent to the printer, and the corresponding response should be examined. The standard Status methods are used, as described in section 19.1 (Serial) or 19.2 (Parallel). The standard sequence is described here. Each printer has slightly different options, so it is important to follow all the steps.

- Determine the number of EEPROM Fields available
- Read the Name and Current Setting of each Field
- Determine the available Options for the Field you want to change
- Set the new Option for that Field

17.2 Programming Specifics

These are the Commands and Status responses for setting the EEPROM Configuration. Note that all variable length responses include the total string length as the second parameter (3rd and 4th characters). All values are in ASCII decimal except for the Odometer and Error Queue, which are in ASCII hex. All values are passed with at least 2-digits (ex #4 = 04).

Command	Response	Description
Read Number of EEPROM Fields		
ESC IXN	XNmm	'mm' = Number of EEPROM fields
Read Current Setting of Specific Field		
ESC IXCmm	XCxx,mm,nn,Title,pp,Option	'xx' = string length, 'mm' = Field, 'nn' = Number of Options, Title of Field, 'pp' = Selected Option value, Option Description
ESC IXC08	XC33,08,04,Low Paper,03,DISABLED	= Low Paper has 4 options (0-3), Disabled is selected
ESC IXC01	XC37,01,00,[Decrement],00,NotSettable	= Decrement Field has no options and can not be changed
Read Option for Field		
ESC IXOmm,nn	XOxx,mm,Title,nn,Option	'xx' = string length, 'mm' = Field, Title of Field, 'pp' = Selected Option value, Option Description
ESC IXO08,00	XO33,08,Low Paper,00,STATUS Only	Low Paper Option 0
ESC IXO08,01	XO26,08,Low Paper,01,BUSY	Low Paper Option 1
ESC IXO08,02	XO29,08,Low Paper,02,WARNING	Low Paper Option 2
ESC IXO08,03	XO30,08,Low Paper,03,DISABLED	Low Paper Option 3
Set New Options for Field		
ESC IXSmm,nn	XSxx,mm,Title,nn,Option	'xx' = string length, 'mm' = Field, Title of Field, 'pp' = Selected Option value, Option Description
ESC IXS08,01	XS26,08,Low Paper,01,BUSY	Change Low Paper Option to Busy
Other Commands		
ESC IXFnnnn	XFxx,nnnn	Set Form Length to 'nnnn' where 0475 = 4.75"
ESC IXR	No Response (default Status)	Reset EEPROM to Factory Defaults
ESC IXL	XLxx,mmmmmmmmNNNNNNNN...	Read Odometer Settings, see 17.5
ESC IXQ	XQxx,mm,nn,pp,...	Read Error Queue, see 17.6

17.3 Programming Sequence

First determine the number of fields available (ESC IXN). Then read the current settings:

```
For 0 to (Number of Fields - 1)
    Read current setting (ESC IXCmm)
    Save Title and Number of Options for each Field
Next
```

Determine which settings you wish to change. Read all the Options for that Field

```
For 0 to (Number of Options-1)
    Read the Option Text (ESC IXOmm,nn)
Next
```

Determine the new Option Number, and set the new Option.

```
ESC IXSmm,nn
Read back the confirmation
```

17.4 Simplified Programming

Once you determine the proper EEPROM settings using the methods above, you can create a command sequence using only the Set Option command (ESC IXSmm,nn). You do not need to retrieve Status between each command. Note that Status will always be sent when using a Serial port.

17.5 Reading the Odometer

The printer retains a record of the print activity in the EEPROM. It is possible to retrieve this information for maintenance or trouble-shooting purposes.

Send the following command to the printer:

```
ESC IXL
```

And retrieve the Status response:

```
XLxx,mmmmmmmmNNNNNNNN...
```

Where 'xx' is the total number of characters in the response string. The Odometer values are sent as 8 ASCII Hex characters, without comma separations.

Example:

```
XL45,0000947700A7938C000109180001605A000A6DD8
```

Breaks down into 5 8-digit fields:

```
00009477 = checksum for entire EEPROM, ignore
00A7938C = 10,982,284 seconds On Time (3050 hours)
00010918 = 67,864 seconds Print Time (18 hours)
0001605A = 90,202 Cut Cycles (Tickets printed)
000A6DD8 = 683,480 inches printed (7.5" ticket)
```

17.6 Reading the Error Queue

The printer retains a record of all errors encountered. It can store the 12 most recent errors. The Error Queue is sent in ASCII Hex pairs.

Send the following command to the printer:

ESC IXQ

And retrieve the Status response:

XQxx,mmNNooPPqqRRss...

Example:

XQ29,323500000000000000000000

Breaks down into 12 2-digit fields:

32 hex = 50 decimal: POR Registration Error

35 hex = 53 decimal: End of Print Registration Error

00 hex = 00 decimal: No other errors recorded

In this case, you might conclude that the printer ran out of paper, and the last ticket was not marked properly (error 53). Then the printer was reloaded improperly (error 50) as no mark was found during power up.

18 Printing Diagnostic Tickets

The following commands are used to force the printer to print a ticket. These can be useful when the switch buttons are not accessible due to the mounting of the printer inside an enclosure.

The printer must be Online and Ready with tickets loaded for these features to work. If desired, the standard Status response will be available.

Command	Description
ESC IXD00	Print System Status ticket
ESC IXD01	Print Splash Ticket (standard/short)
ESC IXD02	Print Long Splash Ticket (print standard if only one available)
ESC IXD03	Print Current Print Energy Test page
ESC IXD04	Print All Print Energy levels
ESC IXD05	Print Odometer and Error Queue
ESC IXD06	Advance and Cut

19 PRINTER to HOST STATUS INFORMATION

The ITX printer can have a Serial or a Parallel interface. It is possible to obtain Status information, from the printer, using either of these interfaces, however, the methods are slightly different. The ITX printer combines the status features and methods used in both the ETX/LTx and ATX/GTX printer families.

In addition, printer Status can be obtained using the DLL utilities provided with the Windows Driver. The DLL is the preferred method for obtaining status information from within an application. However, the following procedures are useful for non-Windows applications, or for existing applications that do not use a driver.

19.1 Serial Port Status

19.1.1 Data Buffering Change

In order to accommodate higher speeds, from the Serial Port, the ITX buffers the incoming data immediately, in interrupt service, without parsing any of the commands. In most instances this should not make any difference. It does, however, mean that detecting status or print commands will occur later, as the background process reads the commands out of the input data buffer.

This has two effects:

- 1- the Busy signal will not be set true in response to a print command and then cleared at the start of printing
- 2- the status command will be acted upon later, in time, as the command is read from the input data buffer.

The host application program may need to have small changes made, if these ETX/LTx legacy functional attributes had been used.

19.1.2 Unsolicited Serial Status

An advantage of the serial port is that it is a full duplex interface, with separate forward and reverse data channels. The printer uses this reverse data channel to send status, without prompting, back to the host computer. The printer must be in XON/XOFF flow control mode for this type of unsolicited status data to be sent (see the User's Manual for serial interface settings).

The printer will send an ACK (06H) character after each ticket is printed successfully. If desired, by sending the <S3> command, at the beginning of the group the printer can be told to send only one ACK for an entire group of tickets (19.1.3.4).

If an error is detected a status character will be sent, immediately followed by an XOFF (13H) control character.

The following are the unsolicited serial status characters:

<u>Status Data(Hex)</u>	<u>System Status</u>
06 (ACK)	Printer acknowledges that the last ticket (group) was printed successfully
11 (XON)	Printer is on-line with paper, ready to print
13 (XOFF)	Printer is NOT ready or is off-line.
0F	Low Paper Condition
10	Printer is out of tickets/paper
12	Power On Successful (previously Low Paper)
18	Jammed ticket (requires operator intervention)
19	Receive Buffer Overflow (16.2).
1A	Power On Not Successful
1C	Flash Download Error
1D	Cutter Error

To suppress the sending of this unsolicited data, send the <S5> command 19.1.3.5. Also, as noted above, unsolicited data will not be sent if the Busy flow control mode is selected. This same unsolicited status data can also be read, by command, see section 19.1.3.2.

19.1.3 Serial Only - Status Commands

The following commands will produce a response from the printer in both XON/XOFF and Busy control modes. Some commands (eg S3 and S5) don't produce any response.

19.1.3.1 Single Byte Status - ASCII

The <Sz> command returns a single ASCII character indicating the current status of the printer system. The <Sz> status is enabled in the Xon/Xoff mode only. The status data returned is as follows:

<u>Status Data(ASCII)</u>	<u>System Status</u>
0	Printer system on line and ready to print
1	Printer system out of tickets/paper
2	Jammed ticket (requires operator intervention)
3	Printer system off line (deselected)
4	Printer system failure
5	Low paper condition
6	Receive Buffer Overflow (16.2)

19.1.3.2 Single Byte Status - Binary

The <S1> status command returns a single character indicating the current state of the printer system. The <S1> status command is only enabled in Xon/Xoff mode. The status data that is returned, is the same (with the exception of the ACK character) as for the unsolicited status, see section 19.1.2.

19.1.3.2.1 Modifying Single Byte Binary Status

In order to prevent the host serial port handler from intercepting some of the status commands (specifically XON and XOFF), the <S6> command can be used to add 0x30 to every byte. For example, ACK would change from 0x06 to 0x36, which is the ASCII character '6'. This will help programs that do not want the host to handle the flow control automatically.

The <S8> command will add 0x30 to all characters **except** XON and XOFF. This will allow programs to read the status responses more easily, while leaving the flow control to the host port handler.

19.1.3.3 Ticket Count and Firmware Version

The <S2> status command returns an ASCII string that contains the current ticket count (7 digits), followed by the version number and date of the installed firmware. The total length of this string varies. This is a legacy ETX/LTx command.

19.1.3.4 Group Acknowledge

Normally, if the printer system is configured in the XON/XOFF mode, an ACK character (06H) is returned after each ticket is printed. If the <S3> command is sent, with the first ticket in the group, a single acknowledge is then returned after all the ticket in that group have been printed. The printer makes the determination that it is at the end of the group if:

- 1- printing is completed
- 2- and there is no new data in the Receive Buffer.

The <S3> command's effect is cleared after each group of tickets has been printed. It must be sent for each group.

19.1.3.5 Disable Unsolicited Status

The <S5> command is used to disable unsolicited status data 19.1.2 from being sent, with the exception of the XON and XOFF flow control characters. Once invoked, the unsolicited status data remains disabled until the printer's power is cycled.

19.1.3.6 User Flash Space Available

The <S7> command will return the number of bytes available in the User Flash. The data is returned in an 8 character ASCII number. This is similar to the parallel status command <SA7>.

19.1.4 Other Serial Status Commands

These status commands are common to both the serial and parallel interfaces: They are:

<u>Command</u>	<u>Description</u>
<SS>	Short Status Group
<SN>	Normal Status Group
<SE>	Extended Status Group
<SA##>	Address Field Status Group
<SC>	Complete Status Group

For a description of the status data, for each of these commands, refer to section (19.2.5.4). The functioning, of these commands, is slightly different for the serial and parallel interfaces. For the serial interface, these commands select the status data that will be returned and also initiate the sending of that data. For the parallel interface, these commands only select the data to be returned.

19.1.5 Sending Serial Status Commands while Busy

Data Flow Control: The serial interface, to support XON/XOFF flow control, is designed to accept a small amount of additional data, after the flow control (Busy signal or XOFF control character) was asserted. This extra buffer space is created by asserting the flow control before the buffer's full limit is reached. This extra space permits the host computer to be few characters late when responding to the flow control. It also permits status commands to be sent after an error has occurred. Under normal data conditions, the Busy/XOFF must be observed and data must not be sent if the flow control has been asserted to stop the data flow.

Detection of an Error: When the printer has detected an error it cannot continue to accept new data. It must assert the flow control (set Busy or send XOFF) to stop the data flow. The host computer may, however, still want to acquire status information to determine what error condition has occurred (type of error, out of paper, etc). To distinguish a normal flow control "Busy" condition, from an error condition "Busy" condition, the host computer will need to apply a timeout. If the Busy condition persists for an extended period (> 30 seconds), then the "Busy" is as a result of an error condition. Additionally, if the printer's Unsolicited Status (19.1.2) has NOT been disabled (19.1.3.5) then host will have additional information to determine the nature of the "Busy" condition. The Unsolicited Status data sent, by the printer, will permit the host computer to identify that an error has occurred, and, for most cases, what type of error (jam, out of paper, etc). Only for this unique condition, when the host computer has determined that the printer "Busy" is due to an error, can the normal flow control procedures be violated by sending a status request command after the "Busy" flow control has been asserted.

Status after Busy: When the host sends data after the "Busy", it is stored in the Receive Buffer until the first Status command is received, or until it cannot hold any more data (the buffer full condition) while continuing to look for a Status command. After the first Status command is detected, all other new data are scanned and discarded while looking for additional Status commands.

Status commands should be sent between tickets to create a safe place to break the data stream. This will allow the host to send an unlimited number of error state Status commands without overflowing the data buffer and creating a Non-Recoverable Data Error (16.2). When paper is loaded (or other error condition resolved), the printer should be ready to re-start printing. As part of the re-start, new data will be received and processed normally. The host should only continue to make Status requests until the replied response indicates that the printer is ready to continue printing. In addition, the printer will clear the flow control Busy condition, to indicate that it is ready to continue printing.

When receiving status commands, after an error Busy condition, these status commands are purged, when acted upon, to maintain this extra data buffer space needed to continue to receive them with. Refer to section for more information regarding status and the printer's operating state (16).

19.2 Parallel Port Status

19.2.1 Data Buffering Change

The Busy signal effects and status command parsing delays, noted above (section 19.1.1), apply to the parallel interface as well.

19.2.2 Unsolicited Status

Unlike the serial interface (section 19.1.2), the parallel interface is half duplex and does not support unsolicited status data.

19.2.3 Parallel IEEE 1284 Interface Standard

The IEEE-1284 Standard is titled: "IEEE Standard Signaling Method for a Bi-directional Parallel Peripheral Interface for Personal Computers". It defines the signaling protocol for the parallel interface. In a Windows operating system environment the IEEE 1284 signaling protocol is built into the Parallel Port's driver. For an operating systems that has not implemented the IEEE-1284 standard a new port driver will be required. With an IEEE 1284 port driver implemented, getting the status data, from the printer, is then a matter of "reading" the parallel port. The port driver then uses the reverse nibble mode communications to capture the status data.

19.2.4 Parallel Interface - Compatibility Mode Status

The Compatibility Mode status is commonly known as Centronics status. This is legacy status that has been part of the parallel interface since its inception. The parallel interface has several signals, which provide a top-level view of the printer's status. These are Busy; PError; Select, and nFault. Collectively these interface signals convey if the printer is ready for data, has ticket stock and does not have a detected fault. In many applications this level of status is sufficient. These status signals are on parallel port's hardware interface when the interface in operating in Forward Compatibility mode.

19.2.5 Parallel Interface IEEE-1284 Reverse Channel Status

19.2.5.1 Requesting Status Data

There is no forward data channel command that requests status information from the Parallel Port. The IEEE-1284 signaling actions, of "reading" the parallel port, triggers an interface level hardware request, to the printer, to send its status data.

19.2.5.2 Reverse Channel Status Data Types

There are two types of requests for status data:

- 1- is a request to have the IEEE-1284 ID String data returned. This information is used, for example, by the Windows "Plug-and-Play" function to identify what peripheral is attached to an interface. This string is not typically requested by a user application program.
- 2- is a request to have "normal" status data returned. This status information is the printer's Detailed Status information. This status information, available via this reverse data channel, includes the information contained in the Compatibility Mode signals as well as additional detailed printer status information. The application software can use this detailed status information for printer supervision, in unattended, remote site deployments.

The data organization for this status information is detailed below in section (19.3.3).

19.2.5.3 Reverse Channel - Detailed Status - Application Program Reading Protocol

To acquire this status information, the application program must use the appropriate API to "read" the parallel port. The API will return with a byte count and the data that it read. For programmer's reference, the byte count, associated with each Status Group Command, is outline below. The returned byte count along with the ID headers for the Status Group and each data Field, completely defines the status data string returned so that the application program can parse it. This parsing is required to distinguish between different data strings. Under certain circumstances, the status data string returned will be different than expected.

If the forward data channel is Busy, or becomes busy, during the sending of a Status Group Selection command, that status selection command will not be received. The status data returned will not be that which was intended by this selection command. The printer's status methods, however, insure that any status data, that is required, will always be available.

If the printer is Busy, at the time of a status data request (when the port was read by the host computer), the printer will always return the Complete Status Group data. This method insures that the application will always have access to all the status data it will need, even if the last forward data channel Status Group Selection command could not get through, due to the Busy on the forward data channel. The Complete Status Group contains all of the Status Groups.

Because of these interface collision circumstances the programmer must take this into account by using the appropriate real-time programming techniques to prevent the application from locking up. For example, if the port is Busy, the Write API, used to send the status data selection command, must be setup to return after a time-out interval with its failure to write error. The programmer must handle this, and other expected error conditions, for the port's "Read" and "Write" operations.

19.2.5.4 Reverse Channel Speed

The reverse data channel is "relatively" slow. A tenth of a second, or so, may be required to return the Short Status. Longer status strings will take proportionately longer. As such, it is important collect status wisely if working in a high printing throughput environment. The Short Status is the most time efficient way to return the top-level printer status information. Other status data should only be requested when needed. For information on the selection of the status data refer to section 17.

19.3 Selection of Detailed Status Information

19.3.1 Status Data Request

As noted above, there is no forward data channel command that requests status information from the Parallel Port (19.2.5.1). When the parallel port is "read" the printer will always respond with status data. This data will typically be the printer's Default Status Group data, unless a prior forward data channel Status Group Selection command had been received. The specific Default Status Group that will be returned is determined by a stored parameter in the printers EEPROM (refer to the User's Manual for EEPROM selections).

19.3.2 Status Group Selection Commands

These commands are:

<u>Command</u>	<u>Description</u>	<u>Section</u>
<SS>	Requests that the Short Status Group data be returned.	(19.3.3.2.2)
<SN>	Requests that Normal Status Group data be returned.	(19.3.3.2.4)
<SE>	Requests that Extended Status Group data be returned.	(19.3.3.2.5)
<SA##>	Requests that an Addressed (##) field's data be returned.	(19.3.3.2.7)
<SC>	Requests that Complete Status Group data be returned.	(19.3.3.2.6)

The above commands are composed of ASCII characters, including the ## for the Addressed Field status.

The host computer sends one of these Status Group Selection commands to select the Status Group data it would like to receive. This command will only remain in force until the next status data request has been processed (19.3.1). Also, noting the interface collision limitations, discussed above (19.2.5.3), there will be occasions where that data selected will not be returned and will need to be parsed from the Complete Status group data that was returned.

19.3.3 Status Information Data Organization

The data organization for the two types of reverse channel data (19.2.5.2) are detailed below.

19.3.3.1 IEEE 1284 ID String Format

String Length MSB; String Length LSB; String Byte #1...; String Byte #n

For example:

MFG:Practical Automation, Inc.;CMD:PAIL2;MDL:ITX-3002;CLS:PRINTER; FIRMWARE VERSION:V1.01 (09/15/00)

Which would be decoded as:

MFG:	Practical Automation, Inc.	; Manufacturer
CMD:	PAIL2	; Command Set, PA Image Language 2
MDL:	ITX3002	; Model
CLS:	PRINTER	; Printer
FIRMWARE VERSION:	V1.01 (09/15/00)	; Manufacturer's discretionary comments

19.3.3.2 Detailed Status Format

All detailed status information is returned as a string, with its format as follows:

- Each string is composed of a grouping of one or more data fields.
- The string's first byte is the status string's Group ID byte (ASCII).
- This Group ID is followed by one of more data fields. These data fields can be of fixed or variable length. Each data field has a two byte Field ID number (ASCII). For a variable length field this Field ID number is also followed by a two byte Field Length count (ASCII).
- For a fixed length field, the data follows the Field's ID number. For a variable length field the data follows the Field's Length count.

19.3.3.2.1 Status Field Format

There are 16 data fields. The first five (00-04) are fixed length and the remaining are variable length fields. The Status data is arranged into numbered fields. These fields can be requested individually or they can be requested in pre-defined groups, as described section 19.3.2.

<u>Status</u>	<u>Data</u>	<u>Field</u>	<u>Field</u>	<u>DataType</u>	<u>Section</u>
<u>Field</u>	<u>Bytes</u>	<u>Type</u>	<u>Name</u>		
00	2	Fixed	Bit Flags Status	Binary Bit Data	(19.3.3.2.3).
01	3	Fixed	Error Code	ASCII Number	(19.4).
02	8	Fixed	Document Count	ASCII Number	(19.3.3.2.8)
03	3	Fixed	PH Temperature	ASCII Number	(19.3.3.2.9)
04	2	Fixed	Form Length	Binary Number	(19.3.3.2.10)
05	'nn'	Variable (~16)	Firmware Version	ASCII String	(19.3.3.2.5)
06	'nn'	Variable (7)	Ticket Number	ASCII Number	(19.3.3.2.11)
07	'nn'	Variable (7)	User Flash Memory Space	ASCII Number	(19.3.3.2.12)
08	'nn'	Variable (~90)	Plug and Play string	ASCII text	(19.3.3.1)
09	'nn'	Variable (16)	Serial Number	ASCII Hex Number	(19.3.3.2.13)
10	Factory Use Only				
11	nn	Variable(1-5)	Advanced Features	ASCII Letters	(19.3.3.2.14)
12-14	TBD	Variable	<i>Future Use</i>	TBD	
15	'nn'	Variable (~15)	Flash Reprogramming	ASCII String	(20)

19.3.3.2.2 Short Status Group

The <SS> Status Group Selection command is used to select this status group. The Short Status Group only returns data field 00. Field 00 is a 16-bit binary data word (2-bytes), which contain the Bit Flag Status defined below.

The status data string format, for the Short Status Group, is:
 "S00bb"

where: 'S' is the string's Group ID (ASCII), '00' is the Field ID (ASCII) and 'bb' are the two status data bytes (Binary).

The Short Status Group is typical default Status for a parallel interface printer. This status data provides information on the printer's internal operating conditions, out of paper, low paper, and other error conditions. Typically using the "Short Status" is sufficient for most applications.

19.3.3.2.3 Short Status Group – Bit Status Format

This data is organized to provide a complete printer status profile in a compact format.

Field 00:			2nd Byte		
1st Byte					
Bit	Data Description	(Notes)	Bit	Data Description	(Notes)
7	System Not Ready		7	Document in Process	
6	System/Diagnostic Error	(a)	6	Interface Data Error	(a)
5	Manually De-Selected	(a)	5	Busy	(a)
4	Output Jam	(a)	4	PH Low/High Temp Wait	(a)
3	Paper/Registration Jam/Error	(a)	3	Reserved Future: Bit = X	
2	Cutter Jam/Error	(a)	2	Average Power Delay Wait	(a)
1	Head Lever Not Ready	(a)	1	Reserved Future: Bit = X	
0	Out Of Paper	(a)	0	Low Paper Condition	(b)

Each bit flag will be asserted to a logical "one" when the printer condition that it represents is asserted. For example the out of paper bit will be equal to a "one" when the printer has detected that there is no paper.

Notes: (a) The System Not Ready Flag is the logical OR of all "a" flags.

(b) The Low Paper Condition will only be included System Not Ready Flag if the printer low paper reaction is selected to go busy after low paper is detected (refer to the User's Manual).

19.3.3.2.4 Normal Status Group

The <SN> Status Group Selection command is used to select this status group. The Normal Status Group returns data fields 00 and 01. These are the Bits Status and the Error code.

The status data string format, for the Normal Status Group, is:
 "N00bb01eee"

where: 'N' is the string's Group ID (ASCII), '00' and '01' are the Field IDs (ASCII), 'bb' is the Bit Status (Binary), 'eee' is a 3-digit Error Code (ASCII) for the most recent error. Refer to the Error Table 19.4

19.3.3.2.5 Extended Status Group

The <SE> Status Group Selection command is used to select this status group. The Extended Status Group returns data fields 00, 01 and 05. These are the Bits Status, Error code and the Firmware Version.

The status data string format, for the Extended Status Group, is:
 "E00bb01eee05nnFWString"

where: 'E' is the string's Group ID (ASCII), '00', '01' and '05' are the Field IDs (ASCII), 'bb' is the Bit Status (Binary), 'eee' is a 3-digit (ASCII) Error Code for the most recent error, 'nn' is the string length (ASCII) for the FWString that follows (ASCII).

A typical Firmware Version String would be: "V1.01 12/25/00".

19.3.3.2.6 Complete Status Group

The <SC> Status Group Selection command is used to select this status group. The Complete Status Group returns data fields 00 through 04. These are the Bits Status, Error, Document Count, PH Temperature, Form Length and the Firmware Version.

The status data string format, for the Complete Status Group, is:

“C00bb01eee02ddddddd03ttC04ff05nnFWString”

where: ‘C’ is the string’s Group ID (ASCII), ‘00’, ‘01’, ‘02’, ‘03’ and ‘05’ are the Field IDs (ASCII), ‘bb’ is the Bit Status (Binary), ‘eee’ is a 3-digit (ASCII) Error Code for the most recent error, ‘ddddddd’ is the Document Count (ASCII), ‘tt’ is the Printhead temperature in deg-C (ASCII), ‘ff’ is the Ticket’s Form Length (Binary) and ‘nn’ is the string length (ASCII) for the **FWString** that follows (ASCII).

19.3.3.2.7 Addressed Status Group

The <SA##> Status Group Selection command is used to select this status group. The Addressed Status Group returns any selected data field, 00 through 15.

The status data string format, for the Addressed Status Group, is:

“A###mddddddd”

where: ‘A’ is the string’s Group ID (ASCII), ‘##’ is the Field ID (ASCII) that was addressed by the <SA##> command, ‘nn’ this is the data length (ASCII) only returned if a variable length field had been selected, ‘ddddddd’ is the selected field’s data.

19.3.3.2.8 Document Count

This data field contains an eight-digit ASCII number (MSB first) representing the documents (tickets) printed since the printer was powered up (it is reset to zero at power up). This document count can be used to schedule preventative maintenance service, etc. Also, this count’s zero value, after having been non-zero, is an indication that the printer has undergone a power cycle.

19.3.3.2.9 Printhead Temperature

This data field contains a two digit ASCII number (MSB first) representing the printhead’s internal temperature (in degrees C). The printer controller internally monitors this temperature. It is used to adjust and limit the energy applied to the printhead. This printhead temperature will rise, above its idle state ambient temperature (several degrees above the printer’s external ambient temperature), as a function of amount of black area printing and printing duty cycle rate. When it reaches a maximum limit (~65C) printing will be suspended to permit cooling. Printing will be resumed when this has cooled (~60C).

This status field can be used to monitor the printer system’s thermal exposure. Understanding how this temperature changes, as a function of the printer’s ambient environment and the printing load, is critical to its proper use.

19.3.3.2.10 Form Length

This data field contains a two digit Binary number (MSB first) representing the reported document length. This data will always be a non-zero number when the printer is operating with registration marked ticket stock. The zero value is reserved to designate a continuously variable forms length printer (a non-registration mark printer - *future*).

19.3.3.2.10.1 Reporting Document Length

The form length is reported in 1/100” increments. The printer will report the size measured during auto-sizing. If auto-sizing has been disabled, the printer will report the Stored Size. The reported size will be the actual print length, which is the measured (or stored) ticket size minus the top and bottom unprintable margins. For example, a standard 5.5” ticket may measure 5.51” (due to minor tolerance differences). After subtracting 0.080” from the top and bottom, the printer will report a Form Length of 5.35”. If the reported Form Length is 0.0” then either the printer has not been loaded with ticket stock to measure, or unmarked stock is being used (*continuous form length printer - future*).

19.3.3.2.11 Ticket Number

This data field contains a seven digit ASCII number (MSB first) representing the ticket number, set by the user command <TC#####> (5).

19.3.3.2.12 User Flash Memory Space

This data field contains a seven digit ASCII number (MSB first) representing the number of data words remaining in the user's Flash memory download data space.

19.3.3.2.13 Serial Number

This data field contains a unique serial number associated with the printer. This number is unique for every printer, and it has no information encoded; it is random. This is useful for systems where many printers are connected to a central server or monitoring system.

19.3.3.2.14 Advanced Features

This field contains single letters to indicate which of several advanced features are enabled. This encoding method was selected for flexibility and ease of parsing by the host programmer. The following letters have meaning at this time:

L = Low Paper Cable Detection is Active.

C = Low Paper Cable detected.

c = Low Paper Cable not connected (lower case).

H = Half Resolution enabled, the printer will behave as if it were one half the model's natural resolution.

E = ECP Enabled (not available on FGL printers).

For example, the Status reply might be "A1102Lc" which would indicate that the Low Paper Cable Detection is Enabled, and the cable is disconnected.

19.3.3.2.15 Flash Reprogramming

This data field contains ASCII string message that represents the Flash memory reprogramming mode status (20).

19.4 Status Error Code List

The list describes the source of the error that has occurred and the code that will be associated with it in the returned printer status information.

	<u>Status</u>	<u>Flash/Beep</u>	<u>Notes</u>
	Error Code	Sequence Length Error Code	
/* System temporary conditions */			
MANUALLY DESELECT CONDITION	1		-a-
TICKET NOT TAKEN CONDITION	2	[3]	1
THERMISTOR TOO COLD WAIT	3	[3]	3
THERMISTOR TOO HOT WAIT CONDITION	4	[3]	3
AVERAGE POWER WAIT CONDITION	5	[3]	4
/*Miscellaneous print time errors */			
DATA RECOVERY EVENT	9	[3]	5 -na-
PRINthead LEVER OPENED	10	[4]	6
PRINthead LEVER OPENED WHILE PRINTING	11	[4]	6
PAPER OUT WHILE PRINTING	12	[4]	7
/* POR errors */			
THERMISTOR_POR_ERROR	30	[5]	10
RAM TEST POR ERROR	31	[5]	2
/* Other Errors */			
FPGAIMAGE_RAM_ERROR	32	[5]	2 -d-
RX_OVERRUN_ERROR	33	[4]	8
A2D_CONFIG_ERROR	34	[5]	7
ESC_CMD_ERROR	35	[4]	8
/* Cutter errors */			
CUTTER INITIALIZATION SEEK EDGE TIMEOUT ERROR	40	[5]	13
CUTTER START NOT AT HOME ERROR	41	[4]	2
CUTTER NOT EXIT HOME TIMEOUT ERROR	42	[4]	2
CUTTER NOT ENTER HOME TIMEOUT ERROR	43	[4]	2
CUTTER NOT ENTER HOME – START OF PRINT	44	[4]	2
/* Registration mark detection errors */			
POR REGISTRATION ERROR	50	[4]	5
POR AUTO SIZE ERROR	51	[4]	5
POR AUTO SIZE BACK UP ERROR	52	[4]	5
END OF PRINT DID NOT FIND MARK REGISTRATION ERROR	53	[4]	5
END OF PRINT WHITE SEEK REGISTRATION ERROR	54	[4]	5
/* Paper output delivery errors */			
POR OUTPUT JAM ERROR	60	[4]	4
PRINT OUTPUT JAM ERROR	61	[4]	4
STEPPER PAPER MOVEMENT OUTPUT JAM ERROR	62	[4]	4
/* Transport Presenter Errors */			
TRANSPORT PRESENTER LOADING FAILURE	63	[4]	3
TRANSPORT PRESENTER INITIAL PRESENT JAM	64	[4]	3
TRANSPORT PRESENTER INITIAL PRESENT JAM - WAIT	64	[3]	2
TRANSPORT PRESENTER DELIVERY/DISPOSE JAM	65	[4]	3
TRANSPORT PRESENTER EJECT JAM	66	[4]	3
TRANSPORT PRESENTER TICKET NOT HELD ERROR	67	[4]	3
TRANSPORT PRESENTER BACKLOAD JAM	59	[4]	3

/* System power errors */			
SYSTEM POWER ERROR LOW SUPPLY VOLTAGE	70	[5]	9
SYSTEM POWER ERROR UNSTABLE SUPPLY VOLTAGE	71	[5]	9
SYSTEM POWER ERROR PH LEAKAGE VOLTAGE INCORRECT	72	[5]	9
SYSTEM POWER ERROR PH SUPPLY INCORRECT	73	[5]	9
/* Parameter EEPROM Errors */			
EEPROM PARAMETER DATA CHECKSUM ERROR	83	[5]	3
EEPROM ALTERNATE DATA SECTOR 0 CHECKSUM ERROR	84	[5]	3
EEPROM ALTERNATE DATA SECTOR 1 CHECKSUM ERROR	85	[5]	3
/* Flash Data Errors */			
FLASH_LOGO_ERROR	97	[5]	4
FLASH_FONT_ERROR	98	[5]	4
/* System Configuration POR errors */			
PH CONFIGURATION ERROR 0	100	[5]	14
PH CONFIGURATION ERROR 1	101	[5]	14
PH CONFIGURATION ERROR 3	102	[5]	14
PH CONFIGURATION ERROR 4	103	[5]	14
PH CONFIGURATION ERROR 5	104	[5]	14
PH CONFIGURATION ERROR 6	105	[5]	14
PH CONFIGURATION ERROR 7	106	[5]	14
SYSTEM CONFIGURATION ERROR 1	107	[5]	14
SYSTEM CONFIGURATION ERROR 2	108	[5]	14
SYSTEM CONFIGURATION ERROR 5	109	[5]	14
SYSTEM CONFIGURATION ERROR 6	110	[5]	14
SYSTEM CONFIGURATION ERROR TP WITH TEARBAR	111	[5]	14
/* MPU Exception vector errors */			
EXCEPTION VECTOR MISC	120	[5]	1
EXCEPTION VECTOR ERRORV	121	[5]	1
EXCEPTION VECTOR GERROR	122	[5]	1
EXCEPTION VECTOR ERROR 6 - 31	123	[5]	1
EXCEPTION VECTOR BUS_ERROR	124	[5]	1
EXCEPTION VECTOR ADDRESS ERROR	125	[5]	1
EXCEPTION VECTOR ILLEGAL INSTRUCTION	126	[5]	1
EXCEPTION VECTOR DIVIDE ERROR	127	[5]	1

-b-

Notes:

- a- The Manually Deselect condition along with, Out of Paper, and Low Paper are annunciated with flashing sequences that are described in the User's Manual, section 4.
- b- Unused error numbers (less than 127) may be assigned in the future.
- d- This error can only occur in an offline diagnostic test. Refer to the User's Manual for this Image RAM tested.
- na- Not applicable

20 Command Driven Flash Memory Re-Programming Procedures

Note: If you are using the latest Windows FGL Driver, you may be able to perform these functions automatically using either the User Printer Properties Panel (UPPP) or the DLL provided.

If you are not using the Windows Driver, then the following information is useful.

20.1 Overview

The printer's operating program, called "firmware," is stored in Flash memory. Occasionally, it is necessary to reprogram this Flash memory to add a new printer feature or correct an error.

The printer has a Flash Memory Reprogramming Mode that permits this update process to take place. The Flash Memory Reprogramming Mode uses that printer's data interface to receive a binary data file, from the host computer, to update the Flash memory. This Flash Memory Reprogramming Mode can be entered in two ways: manually, using the printer's switches, or command driven, by the host computer, over the data interface.

The data, used to reprogram the Flash memory, is stored in Firmware Binary Data Files.

Once the printer is in Flash Memory Reprogramming Mode, the Flash memory reprogramming process is essentially that of copying this Firmware Binary Data File, to the printer, over the data interface. All normal data interface procedures apply to this process.

20.2 Flash Memory Reprogramming Mode Entry

20.2.1 Manual Entry

To manually enter the Flash Memory Reprogramming Mode Flash, two of the printer's User Front Panel switches (F0 and F1) are pressed, and held, while turning on power to the printer. These switches need to be held until a long beep is heard (5 Sec.). At the end of this beep, the printer is in Flash Memory Reprogramming Mode and is ready to accept the reprogramming data over the printer's data interface.

This manual method, of reprogramming the Flash memory, is the most commonly used. A technical service person operates the host computer and the attached printer to control the reprogramming process. The User's Manual provides additional information on this method.

20.2.2 Command Entry

For the printer to be able to accept the commands, to enter Flash Memory Reprogramming Mode, it must be idle (not printing or processing data) but in all other ways ready for normal operation (have paper loaded and not be in error, etc). The command process (defined below) is a sequence of forward data channel commands, monitoring of the printer's reverse data channel status information, and timed action steps.

This command driven method, of reprogramming the Flash memory, is used for fully automatic reprogramming, where the process is completely under the control of the connected host computer.

20.3 Flash Read Only Memory Organization

The printer's Flash memory is organized into two logical sectors: a Boot-Sector and a Program Data Sector. The Boot-Sector contains the printer's operating program that supports the Flash Memory Reprogramming Mode. Even if the Program Data Sector has been erased, the printer will automatically power up and re-enter the Flash Memory Reprogramming Mode. This provides added reliability, for the flash reprogramming process, by always retaining this minimum level of printer function.

20.4 Firmware Binary Data Files

Each Firmware Binary Data file has a unique filename and a suffix (filename.BIN or filename.ALL). The file name identifies the target printer, for which the data is intended, as well as the firmware's version level. The file is internally organized with a preamble header followed by the flash reprogramming image binary data. The file's header information is used, by the printer, during the reprogramming process. This information is used to match the file to the target printer and also to provide a command for the type of programming action that will take place. The file suffix is used to externally denote the type of programming action that is associated with that file.

20.5 Flash Memory Reprogramming Actions

Protected Boot-Sector Flash Memory Reprogramming is the most common form of Flash memory reprogramming. This programming action only causes the Program Data Sector to be erased and then reprogrammed. The advantage to this technique is that the printer always retains its ability to be reprogrammed even if something caused the reprogramming process to fail (for example, due to a power interruption). The Flash Binary Data File that evokes this programming action has the "BIN" suffix (filename.BIN).

Full Flash Memory Reprogramming method is less frequently used but is necessary, in rare cases, where the data in the Boot-Sector, and the Program Data Sector, both need to be reprogrammed. This process functions exactly as the above, however, the Boot-Sector is also erased leaving the small potential for a power interruption to cause the loss of all printer function (where the Flash memory would need to be reprogrammed at the factory or a service depot). The Flash Binary Data File that evokes this programming action has the "ALL" suffix (filename.ALL).

20.6 Command Driven Process for Reprogramming the Flash Memory:

This method of reprogramming the Flash memory requires that the printer must be idle (not printing or processing data), however, in all other ways ready for normal operation (have paper loaded and not be in error, etc). The connected computer then uses a combination of commands (sent in the forward data channel to the printer), reading of printer's status data (via the reverse data channel) and executing a time driven sequence of steps to send the new firmware data to the printer. Finally, verification of successful reprogramming is done by reading the printer's new firmware version that is now embedded in its status data.

An overview of the command driven reprogramming algorithm is as follows:

(0): Host Computer: waits for all previously sent print jobs to be completed and for the printer to become idle.

(1): Host Computer: then requests the Complete Status data field from the printer (<SC>).

(2): Printer: The printer decodes this status command and replies with the requested data.

(3): Host Computer: verifies this status data to determine if:

-1- the printer is ready and

-2- it logs the current version of the firmware that is to be upgraded.

If the printer is found to be idle and ready, go to step (4) otherwise step to the (Error Exit).

(4): Host Computer: Sends the Enter Flash Memory Reprogramming Mode command (ESC*FR) to begin the reprogramming process. (ESC is the ASCII control character = 1BH).

(5): Printer: The printer decodes this Enter Flash Memory Reprogramming Mode Command and then waits to enter the Flash Memory Reprogramming Mode.

(6): Host Computer: Requests the Flash Mode Status field from the printer (<SA15>).

(7): Printer: The printer decodes this status command and replies with the requested Flash Mode Status data. The printer having been in a wait state, as a result of step (5), and now having received the request for the Flash Mode Status, returns the "FLASH READY" status data string, exits normal printer operation and enters the Flash Memory Reprogramming Mode.
Flash.

(8): Host Computer: verifies that the printer is ready to receive the firmware data file by examining the returned status data. The combination of the host requesting this specific status field (<SA15>) and the printer already being in a wait state, from the previously sent Enter Flash Memory Reprogramming Mode command (4), are the two trigger conditions, for the printer, to enter Flash Memory Reprogramming Mode. If the "FLASH READY" status data is returned then go to step (9) otherwise go to step (Error Exit).

(9): Host Computer: waits 15 seconds to permit the printer to complete its entry into the Flash Memory Reprogramming Mode and become ready to receive a Firmware Binary Data file, over the data interface.

(10): Host Computer: the host computer then sends the required Firmware Binary Data file (filename.BIN), to the printer (this takes approximately 30 seconds, to over 15 minutes, depending on the speed of the data interface used).

(11): Printer: *uploads all the Firmware Binary Data* and checks for errors.

If no errors are found the printer will execute a reset operation and be ready for normal printer operation.

If an error was detected the printer will *re-enter the start of the Flash Reprogramming Mode* and wait for the Firmware Data File to be re-sent.

(12): Host Computer: *then waits 15 seconds, after all data has been sent,* for the printer to execute a reset operation and become ready for normal printer operation.

(13): Host Computer: *must verify that state of the programming process.* This is done by observing the returned status data, from the printer:

for a Serial Data Interface Printer:

if the reprogramming process was successful the printer will send an unsolicited (no forward data channel status command is required) ACK character (06H) by the reverse serial data channel. Also, after the reset process has been completed it will send an X-ON character (11B) (if the interface is in XON/XOFF flow control mode).

if the reprogramming was unsuccessful an NACK character (15H) will be sent.

for a Parallel (or USB) Data Interface Printer:

if the reprogramming process was successful then reading of the printer's default status (no forward data channel command needs to be sent and sending one, at this stage of the process, is prohibited). Valid (default) status data being returned, after the printer has completed its reset sequence indicates that the reprogramming has succeeded.

if the reprogramming was unsuccessful then no status data will be returned.

(after the 1st attempt): If successful, go to step (14) otherwise steps (9) through (13) need to be repeated, one more time.

(after the 2nd attempt): If successful go to step (14) otherwise steps (9) through (13) need to be repeated, however, sending the Firmware Data File will the "ALL" suffix (filename.ALL).

(after the 3rd attempt): If successful, go to step (14) otherwise steps (9) through (13) need to be repeated, one more time.

(after the 4th attempt): If successful go to step (14), otherwise go to step (Exit Error).

(14): Host Computer: the printer's *Complete Status Data is requested* (by sending a forward data channel status request command) to verify the correct firmware version information is reflected in the printer's status data. If it is correct go to step (Exit) otherwise (Error Exit).

(Exit): Host Computer: normal printer operation is resumed.

(Error Exit): Host Computer: follows error recovery and notification procedures that are consistent with the system level application.

20.6.1 Command Driven Process Details:

The following are details, related to the re-programming process, are a supplement the information above:

Command Driven Process Steps:

(1): The status command to request complete status is detailed in the Programmer's Manual.

The ASCII character command format is:

<SC>

this command's C-formatted hexadecimal representation is:

0x3C,0x53,0x43,0x3E

(3): The returned status data is formatted as described in the Programmer's Manual. From this returned status data string, all required information can be parsed out by the application program. The printer can be determined to be ready (not in an error condition) and also have its present firmware version string read and logged by the host computer.

(4): The command to request that the printer enters into Flash Memory Reprogramming mode is detailed in the Programmer's Manual. The ASCII character command format is:

ESC*FR

this command's C-formatted hexadecimal representation is:

0x1B,0x2A,0x46,0x52

(5): The printer receives the above command ("ESC*FR"), from the host computer, and then waits for it to sent the request for the Flash Mode Status. If this status request command (" <SA15> ") does not come within 60 seconds the wait interval will expire and the printer will return to its previously idle, normal condition (not waiting to enter Flash Memory Reprogramming Mode).

(6): The command to request the printer's Flash Mode status is detailed in the Programmer's Manual.

The ASCII character command format is:

<SA15>

this command's C-formatted hexadecimal representation is: 0x3C,0x53,0x42,0x30,0x35,0x3E

(7): The printer receiving the above command (before reaching its timeout limit) will respond with the following ASCII status data string:

A1511FLASH READY

this status data string's C-formatted hexadecimal representation is:

0x53,0x31,0x35,0x31,0x31,0x46,0x4C,0x41,0x53,0x48,0x20,

0x52,0x45,0x41,0x44,0x59

This above Addressed Status data field is detailed in the Programmer's Manual.

(8): After the above status string "FLASH READY" has been received by the host computer the printer has already started its transition into Flash Memory Reprogramming Mode. If status data "FLASH NOT READY" was returned it is an indication that the printer did not read the command to enter Flash Memory Reprogramming mode. Error recovery is required.

(9): This 15 sec. timing interval is required to give the printer time to transition into the Flash Memory Reprogramming Mode and be ready to start to receive that firmware data file.

(10): Using normal interface data transmission techniques, the Firmware Binary Data File is sent to the printer. If a Serial data interface is being used:

-1- it must be set for 8 bit data (to support the 8 binary data of the flash firmware file)

-2- the Xon/Xoff flow control mode is recommended, however Busy can also be used.

-3- the fastest baud rate should be selected (up to 512K bytes of data can be sent during this update process although 400K bytes is typical).

(11): The firmware data file is received and a checksum is calculated for the flash data space just programmed. If this value is correct the process is considered successful and the printer executes a reset to begin normal printer operation. If this was not successful the Flash Memory Reprogramming Mode will return to its starting point and start looking for the data to be sent again. There three ways that the Flash Memory Reprogramming Mode can be exited:

- 1- success
- 2- receiving a special abort firmware data file (ABORT_FR.BIN)
- 3- power on reset.

Notes: If the abort file is sent it will only cause an exit from the Flash Memory Reprogramming Mode if no flash memory had been erased. Note: The power cycling of the printer or and external reset will leave the printer dysfunctional if a full flash erase was done before successfully reprogramming in the new flash data.

(12): This timeout interval lets the printer reset and become ready for normal operation after a successful reprogramming. If a Serial data interface is used, the ACK character, followed by the XON character, can be detected and cause the host's timing interval to end. Note: the XON character is only sent if the XON/XOFF flow control is selected.

(13): The host computer can determine if the printer has returned back to normal operating mode (has successfully exited from the Flash Memory Reprogramming Mode) by reading, or receiving, the printer's "default" status information.

For the Serial Data Interface this is accomplished by receiving (over the reverse data channel) the automatically sent, unsolicited status characters (the ACK (06H), followed by the XON (11H). Receiving the ACK, followed by the XON characters, indicates reprogramming success and that printer has returned to its normal operating mode. Receiving the NAK character (15H) indicates that the printer has returned to the start of the Flash Memory Reprogramming Mode and is waiting for the data to be sent again. Note: the XON character is only sent if the XON/XOFF flow control is selected.

For the Parallel Data Interface (or USB, via an adapter cable) this is accomplished by reading of the Parallel Interface Port. This read operation triggers a hardware interface level request, to the printer, to send its default status information (over the reverse data channel). Note: no forward data command is required and one should not be sent. Finding that this status data has been returned, by the printer, is indication of the success of the reprogramming operation. The lack of this returned status data indicates that the printer has returned to the start of the Flash Memory Reprogramming Mode and is waiting for the data to be sent again.

(14): The host computer now knows, from the results of (13), that the printer is back to its normal operating mode. The complete status data can now be requested (<SC>) and from the returned data, the new version identification string, for the newly installed firmware, can be read. This information is used to close that loop by verifying that the new firmware version is present in the printer's status data.

(Exit): Successful flash reprogramming exit.

(Error Exit): If the process was unsuccessful it is a good error exit strategy to send the abort file (ABORT_FR.BIN) to the printer. If flash memory had been erased this abort file will be ignored.

If flash had NOT been erased the printer will return to normal operation. The host computer can then make an attempt to collect the complete status for the printer (<SC>). If this status is returned then the printer can remain functional (with the old firmware) while a system level debug is done to determine the cause of the reprogramming failure. The most common problem is that the incorrect Firmware Binary Data file was sent and rejected by the printer.

If the flash had been erased the printer will remain in the Flash Memory Reprogramming Mode. The two conditions that in can be in, at this point, are:

-1- with the Boot-Sector in tact (not erased). This condition can be visually identified, on-site, by observing the Front Panel's yellow LED. If this LED is flashing then the printer has its Boot-Sector is in tact. Even if the printer lost power it could still be reprogrammed using the data interface. This unit can be easily restored to normal operation, by loading the correct Firmware Binary Data File over the data interface. The printer will always return to the Flash Memory Reprogramming Mode when it is powered on.

-2- with the Boot-Sector erased. This condition is represented by the Front Panel's yellow LED remaining on continuously. This condition is tricky, in that if power is removed, or it is externally reset, the printer will lose all function. If the on-site service person is able to identify this condition, and then, without resetting or powering down the printer, load the correct Firmware Binary Data File, over the data interface, the printer can be recovered to normal operation. Failing this, it would require factory, or depot level, reprogramming.

ASCII CHARACTER SET

21 ASCII Character Sets

Char.	Dec.	Hex.	Char.	Dec.	Hex.	Char.	Dec.	Hex.
NUL	0	0	+	43	2B	V	86	56
SOL	1	1	'	44	2C	W	87	57
STX	2	2	-	45	2D	X	88	58
ETX	3	3	.	46	2E	Y	89	59
EOT	4	4	/	47	2F	Z	90	5A
ENQ	5	5	0	48	30	[(1)	91	5B
ACK	6	6	1	49	31	\ (1)	92	5C
BEL	7	7	2	50	32] (1)	93	5D
BS	8	8	3	51	33	^	94	5E
HT	9	9	4	52	34	~ (2)	95	5F
LF	10	A	5	53	35	¯ (3)	96	60
VT	11	B	6	54	36	a	97	61
FF	12	C	7	55	37	b	98	62
CR	13	D	8	56	38	c	99	63
SO	14	E	9	57	39	d	100	64
SI	15	F	:	58	3A	e	101	65
DLE	16	10	;	59	3B	f	102	66
DC1	17	11	<	60	3C	g	103	67
DC2	18	12	=	61	3D	h	104	68
DC3	19	13	>	62	3E	I	105	69
DC4	20	14	?	63	3F	j	106	6A
NAK	21	15	@ (1)	64	40	k	107	6B
SYN	22	16	A	65	41	l	108	6C
ETB	23	17	B	66	42	m	109	6D
CAN	24	18	C	67	43	n	110	6E
EM	25	19	D	68	44	o	111	6F
SUB	26	1A	E	69	45	p	112	70
ESC	27	1B	F	70	46	q	113	71
FS	28	1C	G	71	47	r	114	72
GS	29	1D	H	72	48	s	115	73
RS	30	1E	I	73	49	t	116	74
US	31	1F	J	74	4A	u	117	75
SP	32	20	K	75	4B	v	118	76
!	33	21	L	76	4C	w	119	77
"	34	22	M	77	4D	x	120	78
#	35	23	N	78	4E	y	121	79
\$	36	24	O	79	4F	z	122	7A
%	37	25	P	80	50	{ (1)	123	7B
&	38	26	Q	81	51	(1)	124	7C
'	39	27	R	82	52	} (1)	125	7D
(40	28	S	83	53	¯ (4)	126	7E
)	41	29	T	84	54	DEL	127	7F
*	42	2A	U	85	55			


- (1) - German characters
- (2) - OCRA special characters
- (3) - OCRA special character or British pound sign
- (4) - OCRA special character or British pound sign

ASCII CHARACTER SET


21.6 Font 13 with LEGEND


	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOL	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	SP	!	“	#	\$	%	&	'	()	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	?	?	?	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	␣
8	€	ü	é	â	ä	à	ä	ç	ê	ë	è	ï	î	ï	Ä	Å
9	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	ƒ	£	©	x	f
A	á	í	ó	ú	ñ	Ñ	±	²	¿	®	¬	¼	¼	¡	«	»
B	⌘	⌘	⌘			À	Ä	Á	©	¶	¶	¶	¶	¶	¶	¶
C	L	L	T	T	-	+	ã	Ä	ℓ	ℓ	ℓ	ℓ	ℓ	ℓ	=	†
D	Ö	Ð	È	È	È	ı	İ	İ	İ	J	ı	■	■	ı	İ	■
E	Ó	ß	Ö	Ó	ö	Ö	μ	p	P	Ü	Ü	Ü	ý	Ý	-	'
F	.	±	=	¾	¶	S	÷	,	°	“	.	1	3	2	■	

LEGEND:

 = Unsupported/ Un-used Control Characters

 = Supported Control Characters

 = Character not implemented in font

 = Space Character not implemented in font

ASCII CHARACTER SET

21.7 Actual Printouts of Fonts 1-13

FONT 1 (5 x 7)

3x3 FONT1
0123456789ABCDEF
2 !"#%&'(<)*+,-./
3 0123456789:;<=>?
4 \$ABCDEFGHIJKLMNO
5 PQRSTUVWXYZÄÖÜ^_
6 &abcdefghijklmnop
7 pqrstuvwxyzäöüß€
8 €ü??ä?????????Ä?
9 ?????ö????öÜ?????
A ??????????????????
B ??????????????????
C ??????????????????
D ??????????????????
E ?ß?????????????????
F ??????§?????????????

FONT 4 (5 x 9)

3 X 3 FONT4
0123456789ABCDEF
2 !"#%&'(<)*+,-./
3 0123456789:;<=>?
4 ABCDEFGHIJKLMNO
5 PQRSTUVWXYZ^Y
6 -
7 -
8 -
9 -
A -
B -
C -
D -
E -
F -
αβΓπΣσμτφθΩ&*εη
ç

FONT 2 (8 x 16)

2x2 FONT2
0123456789ABCDEF
2 !"#%&'(<)*+,-./
3 0123456789:;<=>?
4 \$ABCDEFGHIJKLMNO
5 PQRSTUVWXYZÄÖÜ^_
6 &abcdefghijklmnop
7 pqrstuvwxyzäöüß€
8 €ü??ä?????????Ä?
9 ?????ö????öÜ?????
A ??????????????????
B ??????????????????
C ??????????????????
D ??????????????????
E ?ß?????????????????
F ??????§?????????????

FONT 5 (8 x 16)

2x2 FONTS
0123456789ABCDEF
2 !"#%&'(<)*+,-./
3 0123456789:;<=>?
4 \$ABCDEFGHIJKLMNO
5 PQRSTUVWXYZÄÖÜ^_
6 &abcdefghijklmnop
7 pqrstuvwxyzäöüß€
8 €ü??ä?????????Ä?
9 ?????ö????öÜ?????
A ??????????????????
B ??????????????????
C ??????????????????
D ??????????????????
E ?ß?????????????????
F ??????§?????????????

FONT 3 (17 x 31)

1x1 FONT3
0123456789ABCDEF
2 !"#%&'(<)*+,-./
3 0123456789:;<=>?
4 \$ABCDEFGHIJKLMNO
5 PQRSTUVWXYZÄÖÜ^_
6 &abcdefghijklmnop
7 pqrstuvwxyzäöüß€
8 €ü??ä?????????Ä?
9 ?????ö????öÜ?????
A ??????????????????
B ??????????????????
C ??????????????????
D ??????????????????
E ?ß?????????????????
F ??????§?????????????

FONT 6 (30 x 52)

RL 1x1 FONT6
0123456789ABCDEF
2 !"#%&'(<)*+,-./
3 0123456789:;<=>?
4 \$ABCDEFGHIJKLMNO
5 PQRSTUVWXYZÄÖÜ^_
6 &abcdefghijklmnop
7 pqrstuvwxyzäöüß€
8 €ü??ä?????????Ä?
9 ?????ö????öÜ?????
A ??????????????????
B ??????????????????
C ??????????????????
D ??????????????????
E ?ß?????????????????
F ??????§?????????????

ASCII CHARACTER SET

FONT 7 (17 x 31)

RL 1x1 FONT7

```
0 123456789ABCDEF
2 !"#%&'()*+,-./
3 0123456789:;<=>?
4 @ABCDEFGHIJKLMNO
5 PQRSTUVWXYZ[\]^_
6 `abcdefghijklmno
7 pqrstuvwxyz €
8 €????????????????
9 ?????????????????
A ?????????????????
B ?????????????????
C ?????????????????
D ?????????????????
E ?????????????????
F ?????????????????
```

FONT 10 (25 x 41)

RL 1x1 FONT10

```
0 123456789ABCDEF
2 !"#%&'()*+,-./
3 0123456789:;<=>?
4 $ABCDEFGHIJKLMNO
5 PQRSTUVWXYZÄÖÜ^_
6 £abcdefghijklmno
7 pqrstuvwxyzäöüß€
8 €ü??ä????????A?
9 ??????ö????ÖÜ?????
A ?????????????????
B ?????????????????
C ?????????????????
D ?????????????????
E ?ß?????????????
F ??????§?????????
```

FONT 8 (18 x 30)

1x2 FONT8

```
0 123456789ABCDEF
2 !"#%&'()*+,-./
3 0123456789:;<=>?
4 $ABCDEFGHIJKLMNO
5 PQRSTUVWXYZÄÖÜ^_
6 £abcdefghijklmno
7 pqrstuvwxyzäöüß€
8 €ü??ä????????A?
9 ??????ö????ÖÜ?????
A ?????????????????
B ?????????????????
C ?????????????????
D ?????????????????
E ?ß?????????????
F ??????§?????????
```

FONT 11 (25 x 49)

RL 1x1 FONT11

```
0 123456789ABCDEF
2 !"#%&'()*+,-./
3 0123456789:;<=>?
4 @ABCDEFGHIJKLMNO
5 PQRSTUVWXYZ[\]^_
6 `abcdefghijklmno
7 pqrstuvwxyz €
8 €????????????????
9 ?????????????????
A ?????????????????
B ?????????????????
C ?????????????????
D ?????????????????
E ?????????????????
F ?????????????????
```

FONT 9 (13 x 20)

1x2 FONT9

```
0 123456789ABCDEF
2 !"#%&'()*+,-./
3 0123456789:;>=<?
4 $ABCDEFGHIJKLMNO
5 PQRSTUVWXYZÄÖÜ^_
6 £abcdefghijklmno
7 pqrstuvwxyzäöüß€
8 €ü??ä????????A?
9 ??????ö????ÖÜ?????
A ?????????????????
B ?????????????????
C ?????????????????
D ?????????????????
E ?ß?????????????
F ??????§?????????
```

FONT 12 (46 x 91)

FONT12 LOWER

```
0 123456789ABCDEF
2 !"#%&'()*+,-./
3 0123456789:;<=>?
4 @ABCDEFGHIJKLMNO
5 PQRSTUVWXYZ?????
```

ASCII CHARACTER SET

FONT 13 (20 x 40)

RL 1x1 FONT13

0 1 2 3 4 5 6 7 8 9 A B C D E F
0 1 2 3 4 5 6 7 8 9 A B C D E F
1 " # \$ % & ' () * + , - . /
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 [\] ^ _
` 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 { | } ~ ¯
€ ü é á â ã ä å ç è é ê ë ì í î ï Ì Å
É æ Ø ö ò ù ð ÿ Ö Ü ø £ ¤ × ÷ ÷
À Á Â Ã Ä Å Æ Ç È É Ê Ë Ì Í Î Ï « »
: ; < = > ? @ 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 { | } ~ ¯ Ñ Ò Ó Ô Õ Ö × Ø Ù Ú Û Ü Ý Þ ß à á â ã
ä å æ ç è é ê ë ì í î ï ð ñ ò ó ô õ ö ø ù ú û ü ý þ ÿ