

Imager and ImagerPlus

GRAPHICS PRE-PROCESSOR BOARDS

Copyright © 1993

by

Printek, Inc.
1517 Townline Road
Benton Harbor, MI 49022
616-925-3200

Printek Part Number 2753

11/09/98

Acknowledgements

QMS and Magnum are registered trademarks of Quality Micro Systems.

Printek is a registered trademark of Printek, Inc.

FormsPro, Imager and ImagerPlus are trademarks of Printek, Inc.

Epson is a registered trademark of Seiko Epson.

Proprinter is a registered trademark of International Business Machines Co.

Specifications subject to change without notice.

Table of Contents

Table of Contents	iii
Introduction	1
Imager and ImagerPlus	1
Theory of Operation	1
Notations and Conventions.....	3
Imager Installation and Setup	5
Installation Requirements	5
Imager Installation.....	5
Imager Setup.....	7
Imager Overview	11
Pass-Thru Mode.....	11
Filter Mode	11
Image Mode.....	11
Processing Characteristics	11
QMS Magnum and Imager Differences	12
Performance Considerations.....	12
Label Design Considerations.....	13
Buffered Overlay Overview	14
Basic Commands	17
Turning the Imager On and Off.....	17
Free Format.....	18
Ignore Character	18
Changing the Image Control Character.....	19
Printing the Image Control Character.....	19
Printing Normal Characters	19
Characters	21
Large Characters (ImagerPlus Only).....	21
Normal - Left-to-Right	21
Sideways - Top-to-Bottom	22
Sideways - Bottom-to-Top	23
Upside Down - Right-to-Left.....	24
Small Characters.....	24
7.5 Pitch.....	25
10 Pitch.....	25
12 Pitch.....	26
15 Pitch.....	26
High-Resolution Small Characters	27
Lower Case Descenders.....	28
Reverse Image	28

Half-Tones	29
Boxes and Lines (ImagerPlus Only).....	31
Solid Lines	31
Dashed Lines	32
Boxes	34
Forms.....	35
Bar Codes.....	37
Horizontal Bar Codes	37
Vertical Bar Codes.....	38
Standard Bar Code Table.....	40
Variable Ratio Bar Codes	41
Special Autoprint Options	42
Bar Code Symbolologies	43
Code 39.....	44
Codabar.....	45
MSI.....	46
Interleaved 2 of 5.....	46
UPCA 11 Digit.....	48
UPCE 10 Digit.....	49
UPCE0 6 Digit.....	49
UPCE1 6 Digit.....	50
EAN 13.....	51
EAN 8.....	52
Code 128 A/B/C	53
UCC-128.....	54
PostNet.....	54
Modification and Positioning Commands.....	57
Half-Dots	57
Modifying Height	58
Modifying Width.....	58
Modifying Justification (Vertical Positioning).....	59
Horizontal Tab.....	59
Tabbing in Image Mode.....	60
Modifying Reference Position in Filter Mode.....	60
Carriage Return.....	61
Line Feed.....	61
Form Feed.....	61
Line Slew.....	62
Dot Slew	62
Graphics.....	63
ImagerPlus Graphics (ImagerPlus Only).....	63
FormsPro 4000/4003 Graphics.....	65
Repetitive Printing.....	67

Vertical Repetition.....	67
Auto Increment/Decrement	68
Horizontal Repetition.....	69
Buffered Overlay	70
Image Interrupt	75
Reference Section	77
Pass-Thru Mode Introduction	77
Pass-Thru Mode Command List.....	77
^PY command.....	77
Filter Mode Introduction.....	77
Filter Mode "Printer Control" Commands.....	78
Filter Mode "Printer Control" Commands List.....	79
Filter Mode "Regular" Commands List.....	80
^A command.....	81
^B command.....	81
^D command.....	82
^E command	82
^F command	83
^G command.....	83
^H command.....	83
^K command.....	84
^L command	84
^M command	84
^N command.....	85
^O command.....	85
^PN command.....	85
^R command	86
^S command	86
^T command	86
^U command.....	87
^V command.....	88
^W command.....	89
^X command.....	89
^Y command.....	89
^Z command	90
^[command	90
Image Mode Introduction	91
Image Mode Commands List.....	91
^A command.....	92
^B command.....	92
^C command.....	93
^D command.....	93
^E command	94
^G command.....	95

^H command.....	95
^I command	95
^J command	96
^KF command.....	96
^KH command.....	96
^KL command	96
^KV command.....	97
^LB command.....	97
^LD command	97
^LS command.....	98
^LF command.....	98
^M command	99
^Q command.....	99
^R command.....	100
^S command	101
^T command.....	101
^U command.....	102
^V command.....	103
^W command.....	104
^X command.....	104
^Y command.....	104
^Z command.....	105
Decimal to Hexadecimal to ASCII Conversion	107
Imager Specifications.....	109

Introduction

The Imager™ and ImagerPlus™ are printer resident graphics pre-processor boards that have been designed specially for the Printek® FormsPro™ 4000/4003. Their primary function is to translate QMS® Magnum® Code V Version 1 commands into graphic commands for the printer. The image command syntax allows previously designed bar code "labels", which require QMS Magnum controllers, to print on the FormsPro 4000/4003 with comparable print results.

Imager and ImagerPlus

The Imager graphics pre-processor board is available in two versions. The ImagerPlus supports everything described in this manual; the standard Imager supports only a subset of what is described in this manual.

The standard Imager board is capable of generating all bar codes described in this manual, as well as non-scalable characters, in a "normal - left-to-right" orientation. The Imager will compute any necessary bar code check digits, insert any required start and stop characters, and convert all characters to the desired bar code format. If requested, the Imager will also automatically print the bar code data as human readable text below the bar code.

In addition to this, the ImagerPlus is capable of generating boxes, lines, and large (scalable) characters. The ImagerPlus can print any size character from 0.1 inch to 9.9 inches in height anywhere on the page. Characters can be printed sideways and upside down. Both horizontal and vertical lines can be easily drawn, and the thickness of the lines can be varied. Complete boxes can be drawn, to facilitate the generation of labels. All common bar codes can be printed either horizontally across the page (like a picket fence), or vertically down the page (like a ladder).

Both the Imager and ImagerPlus can easily repeat a label several times across the page and/or down the page, so it is only necessary to send one copy of the label.

In this manual, both versions of the board will frequently be referred to simply as the Imager. When describing some commands, a note saying "ImagerPlus only" may be added to avoid confusion.

Theory of Operation

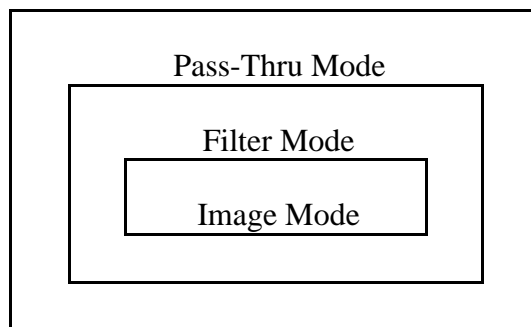
The Imager graphics pre-processor board is installed inside the printer, where it communicates directly with the main board of the printer. But conceptually, the Imager is a filter inserted in the pipeline between the host computer and the printer.



As a filter, the Imager will simply pass all data received on through to the printer, until it receives special image commands telling it to do otherwise. The image commands will be filtered out, processed, and when necessary replaced with native printer commands (ASCII control codes and escape sequences; in particular, graphics escape sequences).

A cornerstone to the operation of the Imager is the use of the image control character. This special character signals the Imager when an image command may have been encountered in the incoming data. In the event that the user cannot use the default image control character, the control character may be changed to a different character. This may be done through the printer's control panel or by sending the Imager a special command to change the control character. The Imager is "passive" to data which it receives from the host device until the image control character is encountered. That is, it simply sends all data it receives to the printer for printing until it receives a control character. If a control character is encountered in the incoming data, the Imager examines the two characters following the control character. If those two characters form the "activation" command, then the Imager will begin analyzing all incoming data for other special commands. If those two characters do not form the "activation" command, then no other action is taken on the incoming data and the data is simply passed on to the printer for printing.

The Imager operates in three different "modes" or levels of processing: pass-thru mode, filter mode, and image mode. Each mode represents a distinct level of operation where certain sequences of characters are interpreted as "commands". The action taken by the Imager when a command is encountered depends on which mode is active at the time. Typically, when a printer equipped with the Imager is powered on, pass-thru mode will be active. When the appropriate command is received by the Imager (while in pass-thru mode), it will activate filter mode. With filter mode active, other commands may be received which will either cause processing of data, change to image mode or change back to pass-thru mode . Each mode provides a different level and type of processing activity.



Briefly, when the **^PY** command is received in pass-thru mode, then filter mode will be activated. When the **^M, ^V, ^E, or ^U** command is received in filter mode, then image mode

will be activated. These commands, and many others, are discussed in more detail later in this manual.

Notations and Conventions

All hexadecimal numbers use the suffix character 'h' to distinguish them from decimal numbers (e.g. 5Eh is equivalent to 94 decimal).

All measurements required by the Imager in filter mode and image mode are specified in tenths of an inch and/or dots. The Imager board creates images based upon 60 dots per inch horizontally and 72 dots per inch vertically. All references to tenth inches horizontally are true tenth inches (6/60 of inches). All references to tenth inches vertically are, in reality, 7/72 of inches.

The Imager command set allows it to be fully controlled and utilized using only printable ASCII characters. ASCII control codes are not part of the Imager command repertoire. Imager commands are introduced by the image control character, which is by default the printable ASCII caret (^) character (94 decimal, 5E hex). Some people may refer to the caret character as the "hat" character.

Note in particular that the ESC control code (27 decimal, 1B hex) is not a part of the Imager command set. When the caret (^) character is seen in this manual, it does not mean ESC and it does not mean "control-_" !

The use of image commands is illustrated throughout this manual. In these illustrations, commentary is provided at the far right as italicized text. For example:

<code>^PY^-</code>	<i>activate filter mode</i>
<code>^F</code>	<i>eat all carriage control</i>
<code>^PN^-</code>	<i>return to pass-thru mode</i>

These italicized comments are not part of the image commands, and should not be appended to the image commands being described. The comments are provided only to improve the reader's understanding of what the image commands are doing.

The *Reference Section* of this manual contains a detailed description of the command format for each command. For example:

`^Hnn`

where: *nn* = *New universal height in 1/10 inches; two digits from 00 to 99.*

Characters in the command that must be typed exactly as shown (^**H**) are not italicized. Parameters that must be filled in (*nn*) are italicized. The italicized characters serve only as place holders for the actual values that will be specified. Note also that the exact number of characters

required have been reserved. The use of *nn*, for example, implies that two digits must be specified. If necessary, use a leading zero (3/10 inches would be specified as "03", not as '3').

Imager Installation and Setup

If your printer was ordered with the Imager option, the Imager graphics pre-processor board has already been installed in your printer at the factory. Although it may be necessary to modify some Imager setup parameters, there is a good chance that the board will work perfectly with the factory default values. You may want to try your printer as is before changing any values.

If you are installing the Imager option in the field, follow the directions below to install and setup your new Imager graphics pre-processor board.

Installation Requirements

The Imager can be installed in any FormsPro 4000 or FormsPro 4003 that has the standard Serial/Parallel interface, or the combined CX/TX/Parallel interface. It cannot be installed in a printer that has the older style CX or TX interface. The type of interface may be determined from the printer serial number. If the third letter of the serial number is an 'A' (serial/parallel) or an 'X' (CX/TX), the Imager can be installed. If the third letter of the serial number is a 'C' (CX) or a 'T' (TX), the Imager cannot be installed.

The printer must have firmware version 1.20 or later. To check the firmware version of your printer, turn the printer off, press and hold the ONLINE button on the control panel, and turn the printer on. Continue to hold the ONLINE button until the control panel displays "FormsPro 4000" or "FormsPro 4003". You may now release the ONLINE button and continue to watch the display. Installed options such as fonts and buffer size will be displayed followed by two firmware versions. These will be displayed as "MP: v1.20" and "PE: v1.20". Again, make sure that both display a value of v1.20 or higher. If your printer has an older version of firmware, contact the dealer or distributor where you purchased the Imager to purchase a firmware upgrade.

Note: Installation of the Imager should be performed by a qualified service representative using the steps described below. The devices used in the electronics of both the printer and the Imager are highly sensitive to static electricity. Take all necessary precautions when working near any of the electronics and when plugging or unplugging cables. Damage caused by improper handling during installation will not be covered under warranty.

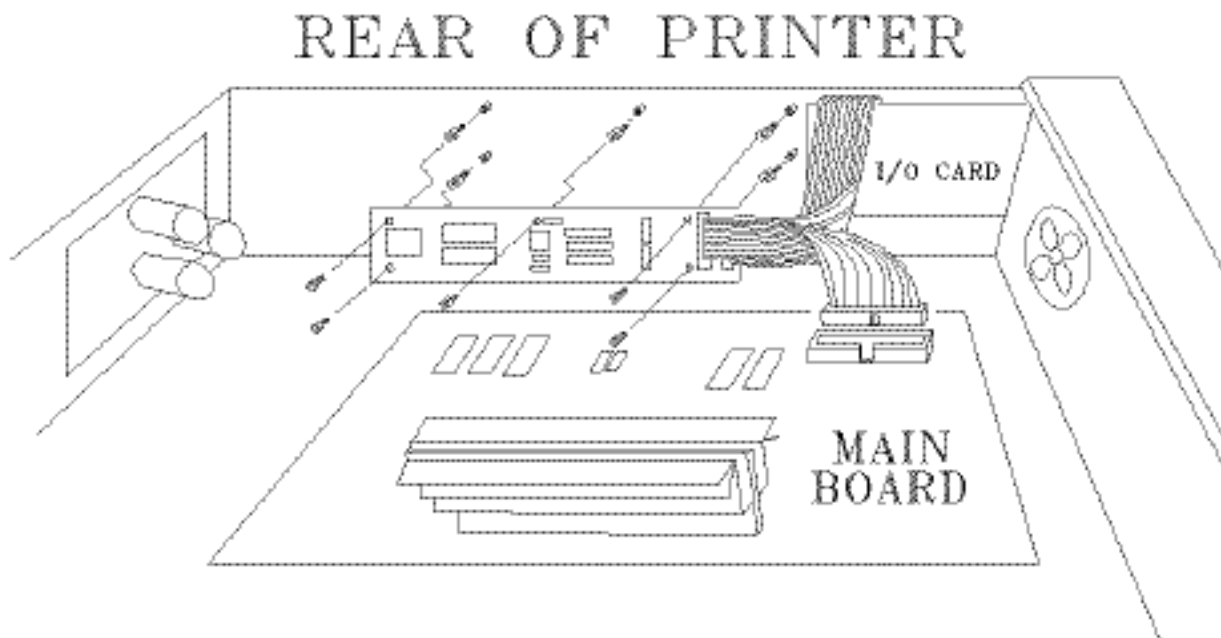
Imager Installation

Turn the printer off and unplug the power cord!

Remove the cover from the top of the printer. To accomplish this, remove the three screws securing the cover to the rear of the printer, and then slide the cover toward the rear of the printer until it exits the mounting slots.

If the printer has firmware older than version 1.20, upgrade the firmware to version 1.20 or later. Follow the instructions provided with the firmware upgrade to change the necessary EPROM's.

Mount the Imager board inside the printer. The board is designed to fit standoffs on the back panel of the printer, next to the I/O board. First, screw the five 1/4" spacers into the standoffs on the back panel of the printer. Tighten, using a socket wrench or pliers. Then, after unwrapping the Imager board, hold it in one hand and touch the back panel of the printer with your other hand to safely discharge static electricity. Finally, put the Imager board in place and secure it with the five screws provided. The screws are 3/16" cross recess fillister head screws. The lower profile fillister head has been used to prevent the screws from shorting against any traces on the Imager board.



Unplug the printer's I/O cable from the main board, and plug it into the connector on the Imager board. Plug the Imager cable into the main board where you just unplugged the I/O cable. The electrical connections are now complete.

Reinstall the cover on the top of the printer. To accomplish this, slide the cover back into the mounting slots and secure it to the rear of the printer with the three screws that were removed earlier.

Plug the power cord in and turn the printer on.

Enter setup mode on the printer's control panel, and specify that the Imager option is installed (see next section). If necessary, additional Imager parameters may also be set at this time. Exit setup mode, and wait for the printer to reset. You may get an "Imager Timeout" or "Imager Error" message. This is normal when the printer is first told that an Imager board is installed, because the Imager is normally initialized at power up.

Turn the printer off, wait a few seconds, and then turn the printer back on. The Imager board will now be correctly initialized and ready for use.

Information about Imager setup is contained in the following section.

Imager Setup

There is one parameter that must be set to indicate an Imager graphics pre-processor board is installed in the printer. There are seven additional parameters which can be set as the "power-up" configuration for the Imager board. These parameters are all set through the printer's control panel, in the options setup menu.

The setup menu for options parameters is the third setup menu. To enter this menu, first take the printer offline. Then press and hold the SETUP button for approximately four seconds until "Setup: OPTIONS" is displayed and the bell beeps three times to indicate that the third menu has been entered. (Note that "Setup: FORMS" and "Setup: INTERFACE" will be displayed before the options menu is displayed. Continue to hold the SETUP button to pass through these menus and proceed to the options menu.) At this time you will be able to view and/or edit the values for the various options functions. The functions relevant to the Imager board are described below. While you may select functions with either the FUNCTION UP or FUNCTION DOWN buttons, they are listed here in the order of FUNCTION UP. The factory default settings are indicated by an asterisk (*).

- **Imager: No*,Yes**

This parameter must be set to "Yes" when the Imager board is installed in the printer, or the printer will not recognize and communicate with the board. If this parameter is set to "No", the other parameters discussed below will not be available in setup mode.

(Note: If an ImagerPlus is installed in the printer, this parameter will be displayed as "ImagerPlus: Yes" after the printer has had a chance to communicate with the ImagerPlus board.)

- **Translation: Off*,On**

^PY filter mode translation processing inactive/active at power-up. If this parameter is set to "On", the Imager will power-up in filter mode. If filter mode is made active this way, it cannot be disabled by issuing the **^PN** "filter mode termination" command.

- **Free Format: Off*,On**

^F free format mode inactive/active (ignore carriage return, line feed and form feed characters) at power-up. The value of this parameter does not become meaningful unless the **^PY** "Translation" parameter discussed above is set to "On" or unless a **^PY** command is received in the data stream. Once this command becomes active, all carriage control characters are "eaten" (absorbed) by the Imager. This setting will remain active until the filter mode command **^O** is encountered in the incoming data stream or until the **^PN** "filter mode termination" command is encountered.

- **Terminator: LF*,CR**

Specifies a carriage return or line feed as the line terminator . Allows the user to dictate which character (a carriage return or a line feed) will be the last character on a line. The user should check the host device (computer) attached to the printer to determine what this setting should be.

- **Ignore Char: Off*,On**

^X ignore character mode inactive/active ('**^A**' required to begin printing). If this parameter is set to "**On**", all incoming data will be "eaten" (absorbed) until the command **^A** is encountered in the incoming data stream. This command is not meaningful unless the **^PY** "Translation" parameter discussed above is set to "On". This setting may be useful on systems where the host computer sends banner pages to the printer before the user application can send print information to the printer. Setting this parameter "**On**" will prevent anything from printing until the user sends the **^A** command to the printer from the host computer.

- **Zero: Normal,Slashed***

Selects normal (unslashed) or slashed zeros. This parameter applies only to expandable characters generated by the Imager board. It does not apply to the small character look-up fonts, including the high-resolution fonts. (A similar parameter is available for each form, in the "Setup: FORMS" menu. The parameter in the forms menu applies only to native printer fonts; it has no effect on characters generated by the Imager board.)

- **Ignore Col 1: No*,Yes**

Specifies whether or not to ignore column 1 (for IBM3287 emulation). If this parameter is set to "**Yes**", column 1 will be "eaten" (absorbed) when filter mode or image mode is active. While the Imager is in pass-thru mode, this setting does not apply. This parameter should be set to "**Yes**" only if the host computer system generates extra carriage-control information in column 1 and that information is not removed by a printer protocol converter.

- **Control Char:** ^*,~,!,?;,;',\

The **Image Control Character** is normally the caret (^) character, but can be changed to one of seven other characters:

Image Control Character List

Character	Description	HEX Value
^	caret	5E
~	tilde	7E
!	exclamation mark	21
?	question mark	3F
;	semicolon	3B
`	open single quote	60
	vertical bar	7C
\	backslash	5C

Imager Overview

Pass-Thru Mode

In **pass-thru mode**, passive, non-translation printing occurs. The print operations may include text and or graphics printing which is native to the FormsPro 4000/4003. Reports, listings, spread-sheets, and graphics software packages would be typical applications that would be used with pass-thru mode printing. In pass-thru mode, all printer action will occur as though there is no Imager board resident in the printer. The Imager simply allows all incoming data to be passed on to the printer without changing anything in the print data stream.

Filter Mode

In **filter mode**, the Imager intercepts information and performs operations on the incoming data before printing occurs. Filter mode is activated by the occurrence of the "filter mode activation" command, **^PY**. Filter mode is deactivated by the occurrence of the "filter mode deactivation" command, **^PN**. When filter mode is deactivated, the printer returns to pass-thru mode. Primarily, filter mode commands are used to "manipulate" the incoming data streams before they are printed. Filter mode commands do not actually perform graphics functions; those functions are handled by image mode commands.

Image Mode

Image mode commands are used to define graphic images such as bar codes, scaled (expanded) text, lines and boxes. There are four commands (**^M, ^V, ^E, and ^U**) which will activate image mode; each one is an **image prefix**. There are three commands (**^*, ^-, and ^**), which will terminate image mode and return to filter mode; each one is a **pass terminator**. An **image sequence** is initiated by the **^M, ^V, ^E, or ^U** command that activates image mode; the sequence includes all of the **image commands** that are processed before returning to filter mode. All of the graphic images defined by the image sequence constitute the **image pass**; the pass will be printed upon receipt of the pass terminator (**^*, ^-, or ^**).

Processing Characteristics

The default image control character is the caret (^) character. Although this character may be changed to a different character by the user, throughout this document, the control character will be referenced as the caret (^) character.

The only valid command which is recognized in pass-thru mode is the "filter mode activation" command, **^PY**. Until this command is encountered, the Imager will pass all incoming data to

the printer. There is no assumption on the part of the Imager as to the type of data being sent to the printer. It may be either text or graphics data.

After the "filter mode activation" command is encountered, the Imager will examine all incoming data for other filter mode commands. If filter mode commands are encountered, the commands will be operated on as defined by the commands' syntax. If a carriage return and/or a line feed character occurs within four character positions following the "filter mode activation" command, they will be absorbed by the Imager. Otherwise, the carriage return and/or the line feed characters will be passed on to the printer for the printer to process.

Once filter mode is active, an "image mode activation" command (^M, ^V, ^E, or ^U) will activate image mode. The printing of all scalable characters, boxes, lines, and bar codes occurs in image mode. Printing is done in passes. A pass may be very simple, printing only a single character, or may be very complex, printing a series of labels containing fixed and variable data. Printing does not occur until the pass is completed by a pass terminator. The pass terminator (^*, ^-, or ^,) forces printing to occur, terminates image mode, and returns to filter mode.

When the "filter mode termination" command ^PN is encountered while the Imager is in filter mode, the active filter mode commands will finish processing and the Imager will then return to pass-thru mode.

QMS Magnum and Imager Differences

Several bar code symbologies are considered proprietary or are no longer in active use. The Imager does not support those bar codes.

Several bar codes found in the QMS Magnum Code V Version 1 which are not supported by the Imager include: AGES, Delta Distance A, Matrix 2 of 5, MRC Edge Code, Rapistan and Identicon 2 of 5.

Bar codes which are supported include: Code 39, Codabar, MSI, Interleaved 2 of 5, UPCA 11 digit, UPCE 10 digit, UPCE0 6 digit, UPCE1 6 digit, EAN 13, EAN 8, Code 128, UCC-128 and PostNet.

UCC-128 and PostNet are not a QMS Code V Version 1 feature but they have been added to the Imager.

Performance Considerations

Text can be printed in pass-thru mode, filter mode or image mode. Printing text (or anything else for that matter) in pass-thru mode does not use any of the capabilities of the Imager; all printed images are handled by the printer's resident emulation. Since pass-thru mode printing does not actually use the Imager capabilities, only filter mode and image mode will be discussed.

Text which is printed in filter mode uses normal printer fonts. The user cannot use the expandable fonts nor can he print in a "sideways" orientation in filter mode. These fonts can only be used in image mode. If the user has the option of printing in either filter mode or image mode and can use the fonts available in either mode, then better performance will be achieved by printing in filter mode. The main reason for this is that filter mode text is printed in a "non-graphics" mode where image mode text is printed in a "raster graphics" mode. In filter mode, the printer can skip blank areas between lines of text. In image mode, the blank areas as well as the text are printed. Generally speaking, for this reason, graphics images do not print as quickly as straight text alone.

If the user requires the printing of expandable characters, sideways text, lines and/or boxes, bar codes or logo images, then image mode must be used. Many labels can be designed where text printing and graphics printing can occur with optimal performance. If graphics images can be printed at the top, middle or bottom of a label and not all the way from top-to-bottom of the label, then printing will occur more quickly. If "boxes" or "lines" are not required as a border around the label, for optimal throughput, do not use them. If bar codes can be printed horizontally instead of vertically, then this can have a big impact on the overall print speed of a print job. Utilize filter mode printing wherever possible and resort to image mode printing only when necessary to achieve the best overall performance.

Label Design Considerations

It is a good idea to know what a label is to look like before coding it for printing. Sketching the label on graph paper or better yet, a printer layout sheet before coding begins can eliminate a lot of time in the design process. A ruler with 1/10 inch, 1/6 inch and 1/8 inch markings will also be very handy. Label stock selection criteria should include: material makeup of the label which will work well with the printer, label size based on standard units (1/10 inches horizontal and 1/6 or 1/8 inches vertical) plus any other issues which are required to satisfy the printing requirements.

Before coding of a label begins, one of two different implementation methods should be decided upon. The Imager has two different methods for processing a label image:

- Buffered Overlay,
- In-line Coding.

When the *buffered overlay* method is used, the Imager will "save" the commands for printing a label image and store them. Then the user can simply send the data down to "fill in" the labels with variable data. The *in-line coding* method involves the repetitive transmission of commands to the Imager to print label images. Many previously designed labels use the in-line coding method. The primary reason for this was the limited memory space available on the older printer controllers; only small labels could use the buffered overlay method. Today, however, lack of memory space is not as big a problem as it once was. It is recommended that when designing and coding a label, the *buffered overlay* method be used whenever possible. There are two main

reasons for this recommendation: First, the volume of data (characters) transmitted to the printer is considerably less when the buffer overlay method is used. The label image only has to be sent one time. Thereafter, only data to fill it out is necessary. Second, the code is easier to maintain because the label commands are generated at one time and do not have to be interspersed with the data at print time by the user.

If the labels do not need to be "filled out" with variable data, then the buffered overlay method would probably not be necessary. This would include labels which may only be printed one time or labels which have incrementing or decrementing serial numbers and which utilize the Imager "repeat loop" command.

Buffered Overlay Overview

Label printing almost always starts with the **^PY^** command. This command informs the Imager to make the filter mode and image mode commands available for use. The second command will normally be a **^F** command. This tells the Imager to "eat" any carriage control character which it may encounter. This command is especially useful with systems which automatically generate carriage control commands with each line sent to the printer. With this command, the label designer has direct control over the movement of the label or paper in the printer. If the **^F** command is used, then the only way carriage returns, line feeds and form feeds will occur is if the designer programs them in with the **^-**, **^*** and **^**, "printer control" commands.

A typical buffered overlay label would have the following generalized format:

^PY^-	<i>turn image processing ON</i>
^F	<i>turn free format processing ON</i>
^B	<i>start buffer overlay definition</i>
(text and graphics label definitions)	
^]	<i>end of buffer overlay definition</i>
(variable data fields to "fill-out" the label)	
^G	<i>end of the variable data</i>
^O	<i>turn free format processing OFF</i>
^PN^-	<i>turn image processing OFF</i>

The "text and graphics label definitions" shown in the example may be made up of filter mode commands or printable text or image mode commands. The "variable data fields" are simply a list of data which will be mapped into predefined locations in the label image when it prints. More information concerning the "buffered overlay" command is available in *Filter Mode Commands* in the *Reference Section*.

As shown in the example, it is good practice to "turn OFF" functions which have been "turned ON" in the process of printing labels. This will leave the printer in a known state when a job completes and the printer will be ready for the next print job. Failure to deactivate certain functions after they have been activated can cause "mysterious" problems for the next job or user of the printer.

Basic Commands

The Imager command set allows it to be fully controlled and utilized using only printable ASCII characters. Image commands may be generated by a specially designed program, written in whatever programming language is convenient. Or, image commands may simply be placed in a file created with a standard text editor or word processor. When generating image commands, it will be helpful to keep the following points in mind.

Image commands are case sensitive. There is a difference between upper-case and lower-case letters in image commands. In most commands, upper-case letters are required.

If a command accepts parameters, there usually are no default values, so the parameters must be specified. When specifying parameters, the exact number of characters (including leading zeros) must appear in the command sequence.

Spaces may not be embedded within a command sequence (unless they are data to be printed).

Turning the Imager On and Off

In pass-thru mode, the Imager simply passes all data received on through to the printer. This may be thought of as passive processing, or you may think of the Imager as being turned off. Placing the Imager in filter mode enables translation processing, and may be thought of as turning it on.

Sending the command **^PY** to the printer will activate filter mode. Sending the command **^PN** will deactivate filter mode and return to pass-thru mode.

Some special rules should be followed when coding the **^PY** command:

- The **^PY** command must be the first printable characters (preceding spaces do not count) on a new line to be recognized.
- The **^PY** command should be terminated by a filter mode carriage return, **^-** (hat-dash).
- The **^PY^-** sequence should be terminated by an ASCII control character, i.e. carriage return or line feed. If a carriage return and/or line feed occurs within four character positions following the **^PY** command, they will be absorbed by the Imager, and no paper motion will occur. Otherwise, the carriage return and/or the line feed characters will be passed on to the printer for the printer to process.

Filter mode may also be activated by specifying "Translation: On" in the "Setup: Options" menu on the printer's control panel. If "Translation: On" has been set, the Imager will be initialized to filter mode at power-up or printer reset. In this case, the **^PN** command will not reset the Imager to pass-thru mode; that can only be accomplished by changing the "Translation" value through the printer's control panel.

Free Format

Free format disables the processing of control codes (ASCII codes 00h to 1Fh). All carriage-control commands, such as carriage return, line feed, and form feed, will be absorbed so they will not be acted upon by the printer.

Some operating systems and application software may automatically generate certain control codes in the data stream. Unexpected control codes could prematurely terminate the pass, cause undesired paper motion, etc. Free format will avoid these problems.

Free format also allows the programmer to separate image commands with line breaks, to greatly increase readability. With free format on, image commands on consecutive lines within a file are treated as a single command line. A pass terminator ends the command line.

Since free format causes all control characters to be ignored, the programmer must explicitly provide for forms control (line feed, form feed, etc.) with the appropriate image commands.

Free format is enabled by the **^F** command, and disabled by the **^O** command. Free format is also disabled when the **^PN** command is used to turn the Imager off.

Ignore Character

The **^X** command will cause all subsequent characters to be ignored, until a **^A** command is received. If a **^A** command is never received, then none of the incoming data will be processed or printed. Use of the ignore character command is illustrated below:

^X this text is absorbed (eaten) **^A** and this data is printed.

Ignore character may be useful if a word processor or programming language is sending undesired characters to the printer. Ignore character may also be used to embed comments among image commands:

^PY^ -	<i>activate filter mode</i>
^F	<i>eat all carriage control</i>
^X	<i>begin ignoring characters</i>
This is a comment	<i>the comment line will not be printed</i>
^A	<i>stop ignoring characters</i>
^PN^ -	<i>return to pass-thru mode</i>

When debugging Imager code, ignore character may be used to comment out certain lines of code while other lines are being debugged.

Changing the Image Control Character

Some applications require use of the caret (^) character, which is the default image control character for image commands.

The `^Nc` command may be used to change the image control character to any other printable character, *c*. For example, the command `^N~` would change the image control character to a tilde. If you change the image control character, be sure that all subsequent image commands use the new control character. If an inappropriate character (i.e. one that appears in the data stream for other reasons) is set as the image control character, unpredictable printing results will occur.

The default image control character may be changed via the printer's control panel, by changing the "Control Char" value in the "Setup: Options" menu.

Printing the Image Control Character

The image control character may be printed by selecting a new image control character, printing the old control character, and then changing back to the old control character. The image command sequence "`^N~^~N^`" will print as the single character '^'.

Printing Normal Characters

Normal characters, using the resident printer fonts and printing at the normal character print speeds, can be printed in pass-thru mode and filter mode.

Printing text (or anything else for that matter) in pass-thru mode does not use any of the capabilities of the Imager; all printed images are handled by the printer's resident emulation. All data is simply passed through the Imager to the printer.

Text which is printed in filter mode also uses the resident printer fonts and normal character print speeds, but a new wrinkle is added because translation processing has been activated. If free format is off, the provided text will print "as is". For example:

`^PY^-`

`^O`

First line of normal text.

Second line of normal text.

`^PN^-`

But if free format is on, an explicit line terminator command must be added at the end of each line of text. For example:

```
^PY^-  
^F  
First line of normal text.^-^*  
Second line of normal text.^-^*  
^PN^-
```

Normal characters using the resident printer fonts cannot be printed in image mode. Any characters printed in image mode will be generated by the Imager and printed in a "raster graphics" mode.

Characters

Large Characters (ImagerPlus Only)

The ImagerPlus can generate characters that range in size from 0.1 inch to 9.9 inches. The characters may be printed in four different orientations: "normal - left-to-right", "sideways - top-to-bottom", "sideways - bottom-to-top", and "upside down - right-to-left".

The height and width of characters may be varied independently. The width of a character includes an inter-character gap, used to separate it from the adjacent character. The inter-character gap is a space to the physical right of a normal or upside down character, and at the physical bottom of sideways characters. The size of the inter-character gap varies with the width and orientation of the character, but is not adjustable by the programmer.

Before using any of the commands to print large characters, a ^PY command must be used to activate filter mode. A command may print any number of large characters, within the constraint that the characters must fit horizontally on the paper. Data which extends beyond the paper will be lost. A pass terminator must be encountered before printing will actually occur.

Normal - Left-to-Right

The ^M command is used to print large characters in a "normal - left-to-right" orientation. The universal height, width, and vertical positioning (justification) are specified, followed by the characters to be printed. The command format is:

^Mhhwwjjdc...c

where: *hh* = Height of characters in 1/10 inches; two digits from 00 to 99 (e.g. 10 = 1.0 inch).
ww = Width of characters in 1/10 inches; two digits from 00 to 99 (e.g. 10 = 1.0 inch). Includes inter-character gap.
jjd = Justification from the current position. Indicates how far down from the top of the pass the characters will begin printing.
jj = 1/10 inches; two digits from 00 to 99 (e.g. 10 = 1.0 inch).
d = Dots (1/72 inches); one digit from 0 to 9 (e.g. 7 = 7/72 inch).
c...c = Character or characters to be printed.

For example, the command sequence "**^M1010000IMAGER^-**" will print the text string "**IMAGER**", using characters that are one inch high and one inch wide.

IMAGER

Sideways - Top-to-Bottom

The `^V` command is used to print large characters in a "sideways - top-to-bottom" orientation. The universal height, width, and vertical positioning (justification) are specified, followed by the characters to be printed. The command format is:

`^Vhhwwjjdc...c`

- where:
- hh* = Height of characters in 1/10 inches; two digits from 00 to 99 (e.g. 10 = 1.0 inch). Represents the physical (vertical) height. Since the characters are sideways, it really specifies what would classically be thought of as character width. Includes inter-character gap.
 - ww* = Width of characters in 1/10 inches; two digits from 00 to 99 (e.g. 10 = 1.0 inch). Represents the physical (horizontal) width. Since the characters are sideways, it really specifies what would classically be thought of as character height.
 - jjd* = Justification from the current position. Indicates how far down from the top of the pass the characters will begin printing.
 - jj* = 1/10 inches; two digits from 00 to 99 (e.g. 10 = 1.0 inch).
 - d* = Dots (1/72 inches); one digit from 0 to 9 (e.g. 7 = 7/72 inch).
 - c...c* = Character or characters to be printed.

For example, the command sequence "`^V0710000DOWN^-`" will print the text string "DOWN", using characters that are one inch high and 0.7 inch wide. The characters will print sideways, from top to bottom.



DOWN

Sideways - Bottom-to-Top

The **^E** command is used to print large characters in a "sideways - bottom-to-top" orientation. The universal height, width, and vertical positioning (justification) are specified, followed by the characters to be printed. The command format is:

^Ehhwwjjdc...c

where: *hh* = Height of characters in 1/10 inches; two digits from 00 to 99 (e.g. 10 = 1.0 inch). Represents the physical (vertical) height. Since the characters are sideways, it really specifies what would classically be thought of as character width. Includes inter-character gap.

ww = Width of characters in 1/10 inches; two digits from 00 to 99 (e.g. 10 = 1.0 inch). Represents the physical (horizontal) width. Since the characters are sideways, it really specifies what would classically be thought of as character height.

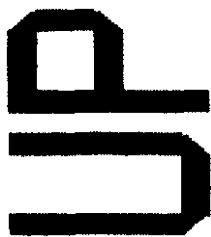
jjd = Justification from the current position. Indicates how far down from the top of the pass the characters will begin printing.

jj = 1/10 inches; two digits from 00 to 99 (e.g. 10 = 1.0 inch).

d = Dots (1/72 inches); one digit from 0 to 9 (e.g. 7 = 7/72 inch).

c...c = Character or characters to be printed.

For example, the command sequence "**^E0710000UP^-**" will print the text string "**UP**", using characters that are one inch high and 0.7 inch wide. The characters will print sideways, from bottom to top.



Upside Down - Right-to-Left

The **^U** command is used to print large characters in an "upside down - right-to-left" orientation. The universal height, width, and vertical positioning (justification) are specified, followed by the characters to be printed. The command format is:

^Uhhwwjjdc...c

where: *hh* = Height of characters in 1/10 inches; two digits from 00 to 99 (e.g. 10 = 1.0 inch).
ww = Width of characters in 1/10 inches; two digits from 00 to 99 (e.g. 10 = 1.0 inch). Includes inter-character gap.
jjd = Justification from the current position. Indicates how far down from the top of the pass the characters will begin printing.
jj = 1/10 inches; two digits from 00 to 99 (e.g. 10 = 1.0 inch).
d = Dots (1/72 inches); one digit from 0 to 9 (e.g. 7 = 7/72 inch).
c...c = Character or characters to be printed.

For example, the command sequence "**^U1010000IMAGER^-**" will print the text string "**IMAGER**", using characters that are one inch high and one inch wide. The characters will print upside down, from right to left.



Small Characters

The same image commands (**^M**, **^V**, **^E**, **^U**) that are used for printing large characters may also be used for printing small, non-scalable characters. The characters may be printed in four different orientations: "normal - left-to-right", "sideways - top-to-bottom", "sideways - bottom-to-top", and "upside down - right-to-left". (ImagerPlus only. The Imager will only print small characters in the "normal - left-to-right" orientation.)

The small characters may be generated in four different pitch sizes: 7.5 cpi, 10 cpi, 12 cpi, and 15 cpi. The 7.5 cpi characters are 0.2 inch high; all of the other characters are 0.1 inch high.

Before using any of the commands to print small characters, a **^PY** command must be used to activate filter mode. A command may print any number of small characters, within the constraint that the characters must fit horizontally on the paper. Data which extends beyond the paper will be lost. A pass terminator must be encountered before printing will actually occur.

7.5 Pitch

The following commands may be used to print small characters at 7.5 cpi. The characters will be 0.2 inch high. The vertical positioning (justification) is specified, followed by the characters to be printed. The command formats are:

^M0000jdc...c	normal - left-to-right
^V0000jdc...c	sideways - top-to-bottom
^E0000jdc...c	sideways - bottom-to-top
^U0000jdc...c	upside down - right-to-left

where: *jjd* = Justification from the current position. Indicates how far down from the top of the pass the characters will begin printing.
jj = 1/10 inches; two digits from 00 to 99 (e.g. 10 = 1.0 inch).
d = Dots (1/72 inches); one digit from 0 to 9 (e.g. 7 = 7/72 inch).
c...c = Character or characters to be printed.

For example, "**^M0000000ABCDEF GHIJKLMNOPQRSTUVWXYZ0123456789^-**" will print the text string "**ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789**", using a character pitch of 7.5 cpi.

ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789

10 Pitch

The following commands may be used to print small characters at 10 cpi. The characters will be 0.1 inch high. The vertical positioning (justification) is specified, followed by the characters to be printed. The command formats are:

^M0101jdc...c	normal - left-to-right
^V0101jdc...c	sideways - top-to-bottom
^E0101jdc...c	sideways - bottom-to-top
^U0101jdc...c	upside down - right-to-left

where: *jjd* = Justification from the current position. Indicates how far down from the top of the pass the characters will begin printing.
jj = 1/10 inches; two digits from 00 to 99 (e.g. 10 = 1.0 inch).
d = Dots (1/72 inches); one digit from 0 to 9 (e.g. 7 = 7/72 inch).
c...c = Character or characters to be printed.

For example, "**^M0101000ABCDEF GHIJKLMNOPQRSTUVWXYZ0123456789^-**" will print the text string "**ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789**", using a character pitch of 10 cpi.

ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789

12 Pitch

The following commands may be used to print small characters at 12 cpi. The characters will be 0.1 inch high. The vertical positioning (justification) is specified, followed by the characters to be printed. The command formats are:

^M0001jdc...c	normal - left-to-right
^V0001jdc...c	sideways - top-to-bottom
^E0001jdc...c	sideways - bottom-to-top
^U0001jdc...c	upside down - right-to-left

where: *jjd* = Justification from the current position. Indicates how far down from the top of the pass the characters will begin printing.
jj = 1/10 inches; two digits from 00 to 99 (e.g. 10 = 1.0 inch).
d = Dots (1/72 inches); one digit from 0 to 9 (e.g. 7 = 7/72 inch).
c...c = Character or characters to be printed.

For example, "**^M0001000ABCDEFGHIJKLMN0PQRSTUVWXYZ0123456789^-**" will print the text string "**ABCDEFGHIJKLMN0PQRSTUVWXYZ0123456789**", using a character pitch of 12 cpi.

ABCDEFGHIJKLMN0PQRSTUVWXYZ0123456789

15 Pitch

The following commands may be used to print small characters at 15 cpi. The characters will be 0.1 inch high. The vertical positioning (justification) is specified, followed by the characters to be printed. The command formats are:

^M0100jdc...c	normal - left-to-right
^V0100jdc...c	sideways - top-to-bottom
^E0100jdc...c	sideways - bottom-to-top
^U0100jdc...c	upside down - right-to-left

where: *jjd* = Justification from the current position. Indicates how far down from the top of the pass the characters will begin printing.
jj = 1/10 inches; two digits from 00 to 99 (e.g. 10 = 1.0 inch).
d = Dots (1/72 inches); one digit from 0 to 9 (e.g. 7 = 7/72 inch).
c...c = Character or characters to be printed.

For example, "**^M0100000ABCDEFGHIJKLMN0PQRSTUVWXYZ0123456789^-**" will print the text string "**ABCDEFGHIJKLMN0PQRSTUVWXYZ0123456789**", using a character pitch of 15 cpi.

ABCDEFGHIJKLMN0PQRSTUVWXYZ0123456789

High-Resolution Small Characters

Six different high-resolution fonts are available through the ^S command for generating small characters. Pitch sizes vary, but all of the characters are 0.1 inch high.

High-resolution characters can be printed in a "normal - left-to-right" orientation and an "upside down - right-to-left" orientation. They cannot be printed in a "sideways" orientation. Reverse image is not allowed. The OCR-A font contains only upper case characters.

Before using the ^S command to print small characters, a ^PY command must be used to activate filter mode, and a ^M command must be used to activate image mode "normal - left-to-right" graphics processing. Any number of small characters may be printed, within the constraint that the characters must fit horizontally on the paper. Data which extends beyond the paper will be lost. A pass terminator must be encountered before printing will actually occur.

To change between high-resolution fonts within a sequence, just issue another ^S command. To change to large characters or standard resolution small characters, another ^M, ^V, ^E, or ^U command must be issued, but it is not necessary to start a new pass.

The command format is:

^Sn

where: *n* = Font number; one digit from 1 to 6.

Font#	Font Description
1	10 cpi (characters per inch)
2	12 cpi
3	13.33 cpi
4	15 cpi
5	17.14 cpi
6	10 cpi OCR-A

For example, the command sequence "**^M0101000^S6OCR-A FONT^-**" will print the text string "**OCR-A FONT**", using OCR-A characters at a character pitch of 10 cpi.

OCR-A FONT

Lower Case Descenders

Some lower case characters (g, j, p, q, y) have descenders which are designed to print below the font base line. The Imager can print these characters in two different ways: aligned at the font base line, or descending the appropriate distance below the base line.

The **^D** command is used to toggle from "descenders off" to "descenders on" and vice-versa. The command format is:

^D

For example, the command sequence "**^M0202000THE gjqy BASE ^Dgjqy^D LINE^-**" will print the text string "**THE gjqy BASE gjqy LINE**". The first "gjqy" will have descenders aligned at the base line. The second "gjqy" will have descenders extending below the font base line.

THE gjqy BASE gjqy LINE

When image mode is activated, the pass always begins with "descenders off". The first **^D** will set "descenders on". A second **^D** will set "descenders off", as will a pass terminator. The state of descenders may be toggled as many times as necessary within the pass.

Setting "descenders on" within an image pass will produce an under-character gap that would not otherwise be present. This is true even if no lower case characters with descenders are printed.

This command is valid for image mode graphics fonts which include the expandable characters. It does not include the "high-resolution fonts" which are activated with the image mode **^S** command. The high-resolution fonts always print the descender characters below the font base line.

Reverse Image

By default, the Imager generates black print on a white background (i.e. black characters on white paper). Reverse image will change this to white print on a black background (i.e. white characters on black paper).

The **^R** command is used to toggle reverse image on and off. The command format is:

^R

For example, the command sequence "**^M0505000^RREVERSE^RIMAGE^-**" will print the text string "**REVERSE**" with white characters, followed immediately by the text string "**IMAGE**" in black characters.

REVERSE IMAGE

When image mode is activated, the pass always begins with reverse image off. The first ^R will turn reverse image on. The second ^R will turn reverse image off, as would the pass terminator. Reverse image may be toggled as many times as necessary within the pass.

Only characters (not boxes, lines, or bar codes) may be printed in reverse image. Printing in reverse image will increase the size of the character cell, since additional space is required to create the black border around the character. Selecting a new font will turn reverse image off.

Half-Tones

Half-tone shading is created by printing every other dot. The ^KH command is used to toggle half-tone shading on and off. The command format is:

^KH

For example, the command sequence "**^M0505000^KHhalf-^KHtone^-**" will print the text string "**half-**" with half-tone shading, followed immediately by the text string "**tone**" with normal shading. The first ^KH will turn half-tone shading on. The second ^KH will turn half-tone shading off. Half-tone shading may be toggled as many times as necessary within the pass.

half -tone

Half-tone shading is not turned off by the pass terminator, or reset when image mode is activated. Once it is turned on, half-tone shading must be explicitly turned off or it will stay on, even for subsequent image passes.

Although by default every other dot is turned off to create half-tones, the actual shading pattern may be modified. The ^KL command is used to define half-tone shading patterns. It defines a 6 bit pattern which is replicated horizontally across an image and which is alternated with the "reverse image" of the same pattern vertically down the image. The command format is:

^KLpp

where: *pp* = Pattern; two hexadecimal digits from C0 to FF, or 00, 01, or 04.

There are 64 user definable patterns, from 'C0' through 'FF'. There are three "built-in" shading patterns, designated by 'pp' values of '00', '01', and '04'.

<i>pp</i>	Shading Pattern
00	vertical stripes
01	diagonal stripes, lower left to upper right
04	diagonal stripes, upper left to lower right
C0 to FF	user definable patterns

The ^KL command will turn half-tone shading on, in addition to changing the pattern for half-tone shading. The ^KH command must still be used to turn half-tone shading off.

Once the half-tone shading pattern has been changed, the new pattern will remain in effect until a different pattern is explicitly set. The pattern is not reset to its default value by the pass terminator, or when image mode is activated. A pattern value of 'EA' will restore the default shading pattern of every other dot, as shown in the example below:

```
^M0504000^KH^KL00STRIPES^KLEADEFAULT^KH^-
```

This image sequence will print the text string "STRIPES" with the special "built-in" half-tone shading pattern that creates vertical stripes, and will then print the text string "DEFAULT" with the standard half-tone shading pattern.

Half-tone shading can be applied only to characters, and the reverse image background of characters. Half-tones cannot be used with boxes, lines, or bar codes.

Boxes and Lines (ImagerPlus Only)

The ImagerPlus is capable of drawing solid and dashed lines, boxes, and forms. Lines may be drawn in varying lengths and widths, and may be drawn horizontally and vertically. Diagonal lines and curved lines are not supported.

All commands to draw lines, boxes, and forms must be contained within an image sequence. But they are drawn according to absolute height and width dimensions that will not be rotated to a different orientation. Any prefix (^M, ^V, ^E, ^U) may be used to enter image mode to draw lines, boxes, and forms; the implied orientation and any universal height and width specifications will not affect the drawing of these structures (but it will affect characters contained in the pass).

Lines, boxes, and forms are drawn at the current print position. Horizontal tabbing and justification commands may be used to reach the desired print position.

Solid Lines

The ^LS command is used to define a solid line. The command requires the height and width to be entered. The command format is:

^LShhhdvvvd

where: *hhh* = Horizontal dimension in 1/10 inches;
three digits from 000 to 136 (e.g. 050 = 5.0 inches).
d = Horizontal dimension additional dots (1/60 inches);
one digit from 0 to 9 (e.g. 7 = 7/60 inch).
vvv = Vertical dimension in 1/10 inches;
three digits from 000 to 132 (e.g. 050 = 5.0 inches).
d = Vertical dimension additional dots (1/72 inches);
one digit from 0 to 9 (e.g. 7 = 7/72 inch).

Although this command is used to draw "lines", it actually defines a rectangular solid, and may be used to draw solid boxes. In thinking of these rectangular boxes as lines, the direction of the line is determined by which dimension (horizontal or vertical) has the greater value. The length and thickness of a line must be at least one dot.

For example, the command sequence "**^M^LS04000003^-**" will draw a horizontal line that is four inches long and three dots thick.



The command sequence "**^M^LS00060200^-**" will draw a vertical line that is two inches long and six dots thick.



The command sequence "**^M^LS01000100^-**" will draw a solid one inch square.



Dashed Lines

The **^LD** command is used to define a dashed line. The command requires the height and width to be entered. The direction of the line is determined by which dimension (horizontal or vertical) has the greater value. The command format is:

^LDhhhvvvd

where: *hhh* = Horizontal dimension in 1/10 inches;
three digits from 000 to 136 (e.g. 050 = 5.0 inches).
d = Horizontal dimension additional dots (1/60 inches);
one digit from 0 to 9 (e.g. 7 = 7/60 inch).
vvv = Vertical dimension in 1/10 inches;
three digits from 000 to 132 (e.g. 050 = 5.0 inches).
d = Vertical dimension additional dots (1/72 inches);
one digit from 0 to 9 (e.g. 7 = 7/72 inch).

When a dashed line is drawn, the line is dashed in 1/10 inch segments. The odd segments of the line are printed and the even segments are left blank. If the length of the line is an even number of tenths, the last tenth will be blank, so the line will appear to be one tenth shorter than specified. If the length of the line contains an additional number of dots, the dots will only be printed if they fall within an odd segment. Space will be reserved for the full length of the line, even if the final dots are not printed.

For example, the command sequence "**^M^LD04000003^-**" will draw a horizontal dashed line that is four inches long and three dots thick.



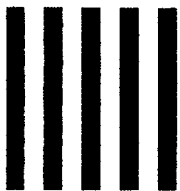
The command sequence "**^M^LD00060200^-**" will draw a vertical dashed line that is two inches long and six dots thick.



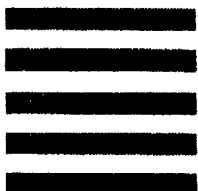
The last tenth inch of these lines will be blank, so they will actually appear to be 3.9 and 1.9 inches long. The command sequences "**^M^LD03900002^-**" and "**^M^LD00060190^-**" would draw lines that are identical in appearance to the previously drawn lines, but they would reserve only 3.9 and 1.9 inches of space, instead of 4.0 and 2.0 inches of space.

Remember that the direction of the line is determined by which dimension (horizontal or vertical) has the greater value. A horizontal line will be dashed along the horizontal axis. A vertical line will be dashed along the vertical axis. If the horizontal and vertical dimensions of a line are identical, the line is considered to be a horizontal line.

For example, the command sequence "**^M^LD01000100^-**" will draw a series of vertical bars, each one inch tall and one tenth inch wide, in a picket fence pattern.



Similarly, the command sequence "**^M^LD01000110^-**" will draw a series of horizontal bars in a ladder pattern.



Boxes

It is possible to create boxes using only the `^LS` line drawing command, by combining two horizontal and two vertical lines (along with the necessary tab and justification commands). But it is much easier to create boxes by simply using the box command.

The `^LB` command is used to define a rectangular box. The command requires the height, width, and box "side" thicknesses to be entered. The command format is:

`^LBhhhdvvdhv`

where: *hhh* = Horizontal dimension in 1/10 inches;
 three digits from 000 to 136 (e.g. 050 = 5.0 inches).
d = Horizontal dimension additional dots (1/60 inches);
 one digit from 0 to 9 (e.g. 7 = 7/60 inch).
vvv = Vertical dimension in 1/10 inches;
 three digits from 000 to 132 (e.g. 050 = 5.0 inches).
d = Vertical dimension additional dots (1/72 inches);
 one digit from 0 to 9 (e.g. 7 = 7/72 inch).
h = Horizontal sides thickness in dots (1/72 inches);
 one digit from 0 to 9 (e.g. 7 = 7/72 inch).
v = Vertical sides thickness in dots (1/60 inches);
 one digit from 0 to 9 (e.g. 7 = 7/60 inch).

The horizontal dimension must be greater than twice the thickness of the vertical sides. The vertical dimension must be greater than twice the thickness of the horizontal sides.

For example, the command sequence "`^M^LB0400010032^-`" will draw a box that is four inches wide and one inch high. The horizontal sides will be three dots thick; the vertical sides will be two dots thick.



Forms

The "form" command allows the programmer to print vertical lines within a box without sending separate commands to define the box and each line in the box.

The `^LF` command is used to define a "form", by printing vertical lines within a box. The vertical lines are drawn from the top edge to the bottom edge of the box. The command requires

the height, width, and "side" thicknesses of the box to be entered, as well as the position and thickness of each vertical line within the box. The command format is:

`^LFhhhdvvdhvppdt...ppdt^G`

where: *hhh* = Horizontal box dimension in 1/10 inches;
 three digits from 000 to 136 (e.g. 050 = 5.0 inches).
d = Horizontal box dimension additional dots (1/60 inches);
 one digit from 0 to 9 (e.g. 7 = 7/60 inch).
vvv = Vertical box dimension in 1/10 inches;
 three digits from 000 to 132 (e.g. 050 = 5.0 inches).
d = Vertical box dimension additional dots (1/72 inches);
 one digit from 0 to 9 (e.g. 7 = 7/72 inch).
h = Horizontal sides thickness in dots (1/72 inches);
 one digit from 0 to 9 (e.g. 7 = 7/72 inch).
v = Vertical sides thickness in dots (1/60 inches);
 one digit from 0 to 9 (e.g. 7 = 7/60 inch).
ppp = Vertical line placement in 1/10 inches;
 three digits from 000 to 136 (e.g. 050 = 5.0 inches).
d = Vertical line placement additional dots (1/60 inches);
 one digit from 0 to 9 (e.g. 7 = 7/60 inch).
t = Vertical line thickness dots (1/60 inches);
 one digit from 0 to 9 (e.g. 7 = 7/60 inch).

The command has a variable number of parameters, with each *ppdt* representing another vertical line to be drawn in the box. The first vertical line drawn in the box is positioned relative to the left side of the box. Subsequent vertical lines are positioned relative to the previous vertical line; not relative to the left side of the box. Since an arbitrary number of parameters may be specified, a **`^G`** is used to terminate the command.

For example, the command sequence "**`^M^LF04000100320100102001^-`**" will draw a box that is four inches wide and one inch high. The horizontal sides will be three dots thick; the vertical sides will be two dots thick. Two vertical lines, each one dot thick, will be drawn in the box at a one inch and two inch interval.



Bar Codes

The Imager is capable of printing many types of bar codes, both horizontally and vertically (ImagerPlus only). Horizontal bar codes resemble a picket fence; vertical bar codes resemble a ladder. Human readable text can be automatically printed beneath horizontal bar codes, and to the right of vertical bar codes.

The Imager automatically calculates and includes check digits in all bar codes that require them. The check digits are calculated from the supplied bar code data, and included somewhere in the bar code along with the data. Check digits are used by bar code readers to insure that a bar code has been read correctly, or to detect the input error.

The programmer can specify the height of the bar codes. The actual width of a bar code will depend upon the bar code type and density, the data encoded, and the ratio of the bars and spaces in the bar code.

The Imager is pre-programmed with numerous bar code types, all capable of printing at the most popular ratios (see the standard bar code table). Variable ratio bar codes may also be printed, giving an almost infinite variety of bar code sizes.

Bar code commands must be contained within an image sequence. Any prefix (^M, ^V, ^E, ^U) may be used to enter image mode to draw bar codes; the universal height or width specification will affect the "height" of bar codes, but the implied orientation will not affect how the bar codes are drawn.

Bar codes are drawn at the current print position. Horizontal tabbing and justification commands may be used to reach the desired print position.

Horizontal Bar Codes

The **^B** command is used for printing horizontal bar codes. The height parameter of the preceding image command (^M, ^V, ^E, ^U, or ^H) will determine the bar code height. The command format is:

^Batd...d^G

- where: *a* = Human readable autoprint indicator,
one character, valid entries are: 'Y'es, 'N'o or 'O'CR.
t = Bar code type, one character, see the Standard Bar Code Table.
d...d = Data to encode as a bar code,
a variable number of characters and/or digits.
^G = Bar code sequence terminator.

For example, the following Imager code:

<code>^PY^-</code>	<i>activate filter mode</i>
<code>^F</code>	<i>absorb all carriage control</i>
<code>^L06</code>	<i>label height = 6 lines (1 inch)</i>
<code>^M05</code>	<i>activate image mode & height = 0.5 inches</i>
<code>^BYAHELLO^G</code>	<i>define bar code; code 39 w/autoprint</i>
<code>^-</code>	<i>terminate image mode and print</i>
<code>^,</code>	<i>issue a label-feed (form feed)</i>
<code>^O</code>	<i>turn ^F command off</i>
<code>^PN^-</code>	<i>return to pass-thru mode</i>

Will print the following bar code:



Vertical Bar Codes

The `^C` command is used for printing vertical bar codes. The width parameter of the preceding image command (`^M`, `^V`, `^E`, `^U`, or `^W`) will determine the bar code "height". The OCR font is not available for printing human readable text with vertical bar codes. The command format is:

`^Catd...d^G`

where: *a* = Human readable autoprint indicator,
one character, valid entries are: 'Y'es or 'N'o.
t = Bar code type, one character, see the Standard Bar Code Table.
d...d = Data to encode as a bar code,
a variable number of characters and/or digits.
`^G` = Bar code sequence terminator.

For example, the following Imager code:

<code>^PY^-</code>	<i>activate filter mode</i>
<code>^F</code>	<i>absorb all carriage control</i>
<code>^L12</code>	<i>label height = 12 lines (2 inches)</i>
<code>^M05</code>	<i>activate image mode & height = 0.5 inches</i>
<code>^CYAHELLO^G</code>	<i>define bar code; code 39 w/autoprint</i>
<code>^-</code>	<i>terminate image mode and print</i>
<code>^,</code>	<i>issue a label-feed (form feed)</i>
<code>^O</code>	<i>turn ^F command off</i>
<code>^PN^-</code>	<i>return to pass-thru mode</i>

Will print the following bar code:



Standard Bar Code Table

INDEX	BAR CODE	CHECK DIGITS	RATIO
A	Code 39	None	1:1:3:3
B	Code 39	None	1:2:4:5
C	Code 39	Mod 43	1:1:3:3
D	Codabar	None	1:2:3:4:1:1:1:1
F	MSI	None	1:1:2:2
G	MSI	Mod 10	1:1:2:2
H	MSI	Mod 10/Mod 10	1:1:2:2
I	MSI	Mod 11/Mod 10	1:1:2:2
K	Interleaved 2 of 5	None	1:1:3:3
L	Interleaved 2 of 5	None	1:2:4:5
P	UPCA 11 Digit	Mod 10	1:1:2:2:3:3:4:4
Q	UPCE 10 Digit	Mod 10	1:1:2:2:3:3:4:4
R	UPCE0 6 Digit	Mod 10	1:1:2:2:3:3:4:4
S	UPCE1 6 Digit	Mod 10	1:1:2:2:3:3:4:4
T	EAN 13	Mod 10	1:1:2:2:3:3:4:4
U	EAN 8	Mod 10	1:1:2:2:3:3:4:4
X	MSI	Mod 11	1:1:2:2
Z	Code 128 (Auto Select A,B,C)	Pseudo Mod 103	1:1:2:2:3:3:4:4
1	UCC-128	Pseudo Mod 10/ Pseudo Mod 103	1:1:2:2:3:3:4:4
2	PostNet	Mod 10	

Variable Ratio Bar Codes

The **^B** and **^C** commands that are used to print standard bar codes are also used to print variable ratio bar codes. In fact, printing variable ratio bar codes is done in an identical manner except that the desired ratio must also be specified. This ratio will override the default ratio that is normally used to print the standard bar code. The height of the bar code will be determined by the preceding image commands. The OCR font is not available for printing human readable text with vertical bar codes. The command formats are:

^Ba9tr...rd...d^G

or

^Ca9tr...rd...d^G

where: *a* = Human readable autoprint indicator,
one character, valid entries are: 'Y'es, 'N'o or 'O'CR.
9 = '9' indicating a user defined ratio follows.
t = Bar code type, one character, see the Standard Bar Code Table.
r...r = User defined ratio; a variable number of digits,
depending on bar code type, each digit from 1 to F.
d...d = Data to encode as a bar code,
a variable number of characters and/or digits.
^G = Bar code sequence terminator.

The *r...r* ratio is specified in pairs of digits representing bar/space ratios. The number of digits that must be specified depends on the bar code type; it will be the same as the number of digits in the default ratio in the Standard Bar Code Table. For example, code 39 requires the specification of four digits, representing the width of narrow bars, narrow spaces, wide bars, and wide spaces respectively.

For example, the following Imager code:

^PY^-	<i>activate filter mode</i>
^F	<i>absorb all carriage control</i>
^L06	<i>label height = 6 lines (1 inch)</i>
^M05	<i>activate image mode & height = 0.5 inches</i>
^BY9A1234HELLO^G	<i>define bar code; code 39 w/ratio 1:2:3:4</i>
^-	<i>terminate image mode and print</i>
^,	<i>issue a label-feed (form feed)</i>
^O	<i>turn ^F command off</i>
^PN^-	<i>return to pass-thru mode</i>

Will print the following bar code:



Special Autoprint Options

The bar code commands (^B and ^C) usually use a 'Y' or 'N' to indicate autoprinting of human readable text under a bar code. Several additional options are available.

If a number from '1' to '6' is used instead of the 'Y', then the high-resolution font corresponding to the ^S font select command will be used. For example, the sequence "^M05^B1A12345^G^-" will print the following bar code, using high-resolution 10 cpi characters for autoprint.



Scalable (large) characters may be used for autoprinting by replacing the 'Y' with a string of the form "9hhww", where "hh" is the character height and "ww" is the character width in tenth inches. For example, the sequence "^M07^B90303A12345^G^-" will print the following bar code, using autoprint characters that are three tenths of an inch high and wide.



The same approach will also work for variable ratio bar codes. The sequence "^M15^B907079A226612345^G^-" will print the following bar code at twice its default ratio, using autoprint characters that are seven tenths of an inch high and wide.



It is also possible to force the autoprint characters to print above or below the bar code. For horizontal bar codes, the autoprint characters print below the bar code by default. If the 'Y' is preceded by an 'A', the characters will print above the bar code. For example, the sequence "^M05^BAYA12345^G^-" will print the following bar code.



For vertical bar codes, the autoprint characters print "above" the bar code by default. If the 'Y' is preceded by a 'B', the characters will print "below" the bar code. For example, the sequence `"^M05^CBYA12345^G^-"` will print the following bar code.



Bar Code Symbolologies

A bar code is a graphical representation of characters. A bar code symbol contains a sequence of varying width bars and spaces representing the characters encoded in the bar code. The actual pattern necessary to encode a particular character is dependent upon the type of bar code. The length of a bar code is dependent upon the type of bar code, the number of data characters encoded, and the bar/space ratios.

Different bar code symbolologies support different character sets. Numeric symbolologies can encode only numbers. Other symbolologies can encode alphanumeric characters, and some can encode the entire ASCII character set.

Bar code symbolologies may be discrete or continuous. In a discrete code, each character can stand alone and be decoded independently from the adjacent characters. Each character begins and ends with a bar, and is separated from its neighbor by a loosely tolerated intercharacter gap that contains no information. In a continuous code, each character begins with a bar and ends with a space. The end of one character is indicated by the start of the next character. There are no intercharacter gaps. A continuous code is denser, requiring less symbol length to encode a given amount of data.

A bar code symbol encodes data in the widths of the bars and spaces. A bar code symbology may employ only two element widths (wide and narrow), or may employ multiple widths. The widths of the elements are specified relative to the nominal width of the narrow elements (both bars and spaces). For example, a bar code with ratio 1:1:3:3 (narrow bar : narrow space : wide bar : wide space) has wide elements that are three times the width of the narrow elements.

Some bar code symbolologies are designed to encode data of a fixed length. Others should be used only in a fixed length environment because of data security reasons (a partial scan may appear to be valid). Some symbolologies can safely encode variable length data.

Bar code symbologies differ in the amount of data that can be encoded in a given distance. When comparing the relative densities of different symbologies, it is customary to compare codes printed with the same nominal width narrow elements.

A symbology is considered to be self-checking if a single printing defect will not cause a character to be transposed into another valid character in the same symbology.

A start code is a particular pattern of bars and spaces placed at the beginning of a bar code to indicate to the scanner where the symbol begins. A stop code is a pattern placed at the end of a bar code to indicate where the symbol ends. The start and stop codes may also indicate the direction of the scan.

A check character may be placed in a predetermined position in a bar code symbol. The value of the check character is mathematically calculated from the other characters encoded in the symbol. The scanner uses the check character to validate that correct data has been decoded. If the check character can only assume numeric values (0-9), it is often called a check digit.

All bar codes require a quiet zone at each end to permit a scan to begin and end in a blank area. The quiet zones should be at least 0.25 inches wide and be completely blank to allow accurate reading of the start/stop codes and to prevent adjacent bar codes from overlapping. The programmer is responsible for providing adequate quiet zones when printing bar codes.

The different bar code symbologies supported by the Imager are described below. The description includes start and stop codes, valid character set, the data field, check digits, and any other information necessary for the creation of valid bar codes.

Code 39

Code 39 was the first alphanumeric symbology to be developed. It is widely used, having become the de facto standard for nonretail bar codes. It is a discrete, self-checking, variable length symbology.

An asterisk (*) is used for the start and stop code. They will be included automatically by the Imager; they should not be included in the data field by the programmer.

The Code 39 character set contains 43 characters: 0-9, A-Z, -, ., \$, /, +, %, and space. All characters are constructed from five bars and four intervening spaces. Of these nine elements, three are wide and six are narrow. A bar code may contain from 1 to 40 characters.

Code 39 may be printed with or without a check digit. Code 39 type C (see the Standard Bar Code Table) includes a modulo 43 check digit, which is automatically generated by the Imager.

Code 39 bar codes require a four digit ratio, with the digits representing: narrow bar, narrow space, wide bar, wide space.

For example, the command sequence "**^M05^BYA12345^G**" will generate the following bar code, using the default ratio of 1:1:3:3.



The command sequence "**^M08^BY9A226612345^G**" will generate the following bar code, using double the default ratio.



The command sequence "**^M05^BYC12345^G**" will generate the following bar code, containing an automatically generated modulo 43 check digit.



Codabar

Codabar is commonly used in libraries, blood banks, and air parcel express applications. It is a discrete, self-checking, variable length symbology.

Four start/stop code characters (A, B, C, D) are available in any combination as start/stop codes. The start and stop code characters must be included as part of the data field to be produced with the bar code.

The Codabar character set contains 16 characters: the digits 0-9, and the characters \$, :, /, .., +, -. All characters are constructed from four bars and three intervening spaces. A bar code may contain from 1 to 40 characters.

Codabar bar codes require an eight digit ratio, with the digits representing: narrow bar, narrow space, wide bar, wide space, ignored, intercharacter gap, ignored, ignored.

For example, the command sequence "**^M05^BYDA1234B^G**" will generate the following bar code, using the default ratio of 1:2:3:4:1:1:1:1.



The command sequence "**^M08^BY9D24681211A2468B^G**" will generate the following bar code, using double the default ratio.



MSI

MSI Code is primarily used for the marking of retail shelves. It is a derivative of Plessey Code, which is a "pulse width modulated" code. MSI symbols are variable length, low density, continuous, and not self-checking.

The MSI character set contains the ten digits 0-9. Each character consists of four bars and four spaces, with each bar-space pair representing one bit of information. Zero bits consist of a narrow bar followed by a wide space. One bits consist of a wide bar followed by a narrow space. An MSI symbol includes a start code, data characters, optionally one or two check digits, and a stop code. The start code, stop code, and check digit(s) will be included automatically by the Imager. A bar code may contain from 1 to 40 digits.

MSI bar codes require a four digit ratio, with the digits representing: narrow bar, narrow space, wide bar, wide space.

For example, the command sequence "**^M05^BYG0123456789^G**" will generate the following bar code, using the default ratio of 1:1:2:2.



The command sequence "**^M08^BY9G22440123456789^G**" will generate the following bar code, using double the default ratio.



Interleaved 2 of 5

Interleaved 2 of 5 is a high density, continuous, self-checking, variable length numeric symbology. It is used primarily in the distribution industry.

The start code consists of two narrow bars and two narrow spaces. The stop code consists of one wide bar, a narrow space, and a narrow bar. They will be included automatically by the Imager.

The Interleaved 2 of 5 character set contains the ten digits 0-9. Each Interleaved 2 of 5 character actually encodes two digits; one in the bars and one in the spaces. There are five bars, two of

which are wide and three of which are narrow. Likewise, there are five spaces, two of which are wide and three of which are narrow. All of the odd-positioned digits are encoded in the bars, and all of the even-positioned digits are encoded in the spaces. The interleaving process requires an even number of digits. If an odd number of digits is specified, a leading zero is automatically inserted by the Imager. A bar code may contain from 1 to 40 digits.

Interleaved 2 of 5 bar codes require a four digit ratio, with the digits representing: narrow bar, narrow space, wide bar, wide space.

For example, the command sequence "**^M05^BYK123456^G**" will generate the following bar code, using the default ratio of 1:1:3:3.



The command sequence "**^M08^BY9K2266123456^G**" will generate the following bar code, using double the default ratio.



A partial scan (a scan that does not include both quiet zones) of an Interleaved 2 of 5 bar code has a high probability of decoding as a valid, but shorter symbol. This is due to the simple structure of the start and stop codes -- a partial scan of an Interleaved 2 of 5 digit can appear to be a start or stop code. Because of this, Interleaved 2 of 5 is best used in a fixed length application, where the scanner is programmed to look for a specific number of digits. Leading zeros may be added to maintain fixed length strings.

Another alternative is to add protection stripes, also called bearer bars, to the top and bottom of the Interleaved 2 of 5 bar code. These prevent a partial scan from being decoded as a valid symbol. The following code will generate an Interleaved 2 of 5 bar code with bearer bars.

<code>^PY^-</code>	<i>activate filter mode</i>
<code>^F</code>	<i>turn free format on</i>
<code>^M10^KF</code>	<i>activate image mode, 1.0 inch height, hi-res on</i>
<code>^T0000^J000^LS01730004</code>	<i>print top bearer bar</i>
<code>^T0003^J000^BYK0123456789^G</code>	<i>print bar code</i>
<code>^T0000^J083^LS01730004</code>	<i>print bottom bearer bar</i>
<code>^KF^-</code>	<i>hi-res off</i>
<code>^O</code>	<i>turn free format off</i>
<code>^PN^-</code>	<i>return to pass-thru mode</i>



UPCA 11 Digit

The Universal Product Code (UPC) has been used in the supermarket industry since 1973. It is a fixed length, continuous, numeric symbology. There are three versions of the UPC symbol: Version A, which encodes 12 digits; Version E, which encodes six digits; and Version D, which encodes variable length data. Version D is rarely used, and is not supported by the Imager.

A UPC Version A symbol consists of a left guard pattern, six numeric digits, a center guard pattern, six more numeric digits, and a right guard pattern. The first digit is the UPC number system digit. The next five digits are the UPC manufacturer's code. The following five digits are the UPC product code. The last digit is a modulo 10 check digit.

For the UPCA 11 Digit bar code (type P), the programmer must specify exactly 11 digits, representing the number system, manufacturer's code, and product code. The Imager will automatically generate the left, center, and right guard patterns, as well as the modulo 10 check digit.

The UPC symbology uses multiple element widths to encode characters. Each character has seven modules, which may be either black or white, to create two bars and two spaces of varying width. UPC bar codes require an eight digit ratio, with the digits representing:

- 1 module wide bar
- 1 module wide space
- 2 module wide bar
- 2 module wide space
- 3 module wide bar
- 3 module wide space
- 4 module wide bar
- 4 module wide space.

For example, the command sequence "**^M05^BYP01234567890^G**" will generate the following bar code, using the default ratio of 1:1:2:2:3:3:4:4.



The command sequence "**^M08^BY9P2244668801234567890^G**" will generate the following bar code, using double the default ratio.



UPCE 10 Digit

The Universal Product Code (UPC) has been used in the supermarket industry since 1973. It is a fixed length, continuous, numeric symbology. There are three versions of the UPC symbol: Version A, which encodes 12 digits; Version E, which encodes six digits; and Version D, which encodes variable length data. Version D is rarely used, and is not supported by the Imager.

UPC Version E bar codes are special zero-suppressed Universal Product Codes that compress 10 data characters down to six characters using specific rules. A UPC Version E symbol consists of a left guard pattern, six numeric digits, and a right guard pattern.

For the UPCE 10 Digit bar code (type Q), the programmer must specify exactly 10 digits, representing the manufacturer's code and product code. The Imager will automatically compress these 10 digits to six digits, generate the left and right guard patterns, and implicitly encode the number system digit (always a zero) and the modulo 10 check digit in the bar code.

The UPC symbology uses multiple element widths to encode characters. Each character has seven modules, which may be either black or white, to create two bars and two spaces of varying width. UPC bar codes require an eight digit ratio, with the digits representing:

- 1 module wide bar
- 1 module wide space
- 2 module wide bar
- 2 module wide space
- 3 module wide bar
- 3 module wide space
- 4 module wide bar
- 4 module wide space.

For example, the command sequence "**^M05^BYQ1230000064^G**" will generate the following bar code, using the default ratio.



UPCE0 6 Digit

The Universal Product Code (UPC) has been used in the supermarket industry since 1973. It is a fixed length, continuous, numeric symbology. There are three versions of the UPC symbol: Version A, which encodes 12 digits; Version E, which encodes six digits; and Version D, which encodes variable length data. Version D is rarely used, and is not supported by the Imager.

UPC Version E bar codes are special zero-suppressed Universal Product Codes that compress 10 data characters down to six characters using specific rules. A UPC Version E symbol consists of a left guard pattern, six numeric digits, and a right guard pattern.

For the UPCE0 6 Digit bar code (type R), the programmer must specify exactly six digits. These digits represent the manufacturer's code and product code, but have already been compressed from 10 digits to six digits. The Imager will automatically generate the left and right guard patterns, and implicitly encode the number system digit (always a zero) and the modulo 10 check digit in the bar code.

The UPC symbology uses multiple element widths to encode characters. Each character has seven modules, which may be either black or white, to create two bars and two spaces of varying width. UPC bar codes require an eight digit ratio, with the digits representing:

- 1 module wide bar
- 1 module wide space
- 2 module wide bar
- 2 module wide space
- 3 module wide bar
- 3 module wide space
- 4 module wide bar
- 4 module wide space.

For example, the command sequence "**^M05^BYR123643^G**" will generate the following bar code, using the default ratio.



UPCE1 6 Digit

The Universal Product Code (UPC) has been used in the supermarket industry since 1973. It is a fixed length, continuous, numeric symbology. There are three versions of the UPC symbol: Version A, which encodes 12 digits; Version E, which encodes six digits; and Version D, which encodes variable length data. Version D is rarely used, and is not supported by the Imager.

UPC Version E bar codes are special zero-suppressed Universal Product Codes that compress 10 data characters down to six characters using specific rules. A UPC Version E symbol consists of a left guard pattern, six numeric digits, and a right guard pattern.

For the UPCE1 6 Digit bar code (type S), the programmer must specify exactly six digits. These digits represent the manufacturer's code and product code, but have already been compressed from 10 digits to six digits. The Imager will automatically generate the left and right guard patterns, and implicitly encode the number system digit (always a one) and the modulo 10 check digit in the bar code.

The UPC symbology uses multiple element widths to encode characters. Each character has seven modules, which may be either black or white, to create two bars and two spaces of varying width. UPC bar codes require an eight digit ratio, with the digits representing:

- 1 module wide bar
- 1 module wide space
- 2 module wide bar
- 2 module wide space
- 3 module wide bar
- 3 module wide space
- 4 module wide bar
- 4 module wide space.

For example, the command sequence "**^M05^BYS123643^G**" will generate the following bar code, using the default ratio.



EAN 13

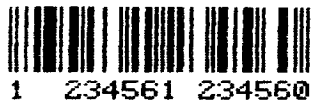
The European Article Numbering system (EAN) is a superset of UPC, and is the international standard bar code for retail food packages. An EAN scanner can decode UPC, but a UPC scanner typically cannot decode EAN. Like UPC, EAN is a fixed length, continuous, numeric symbology.

An EAN 13 symbol consists of a left guard pattern, six numeric digits, a center guard pattern, six numeric digits, and a right guard pattern. An EAN 13 symbol contains the same number of bars as UPC Version A, but encodes a thirteenth digit into the parity pattern of the left six digits. Two digits are used as flag digits to represent a country code, and one digit is a modulo 10 check digit. The programmer must specify exactly twelve digits. The Imager will automatically generate the guard patterns and the check digit.

The EAN symbology uses multiple element widths to encode characters. Each character has seven modules, which may be either black or white, to create two bars and two spaces of varying width. EAN bar codes require an eight digit ratio, with the digits representing:

- 1 module wide bar
- 1 module wide space
- 2 module wide bar
- 2 module wide space
- 3 module wide bar
- 3 module wide space
- 4 module wide bar
- 4 module wide space.

For example, the command sequence "**^M05^BYT123456123456^G**" will generate the following bar code, using the default ratio.



EAN 8

The European Article Numbering system (EAN) is a superset of UPC, and is the international standard bar code for retail food packages. An EAN scanner can decode UPC, but a UPC scanner typically cannot decode EAN. Like UPC, EAN is a fixed length, continuous, numeric symbology.

An EAN 8 symbol consists of a left guard pattern, four numeric digits, a center guard pattern, four numeric digits, and a right guard pattern. The eight numeric digits that are encoded include two flag digits that represent a country code, five data digits, and one modulo 10 check digit. The programmer must specify exactly seven digits. The Imager will automatically generate the guard patterns and the check digit.

The EAN symbology uses multiple element widths to encode characters. Each character has seven modules, which may be either black or white, to create two bars and two spaces of varying width. EAN bar codes require an eight digit ratio, with the digits representing:

- 1 module wide bar
- 1 module wide space
- 2 module wide bar
- 2 module wide space
- 3 module wide bar
- 3 module wide space
- 4 module wide bar
- 4 module wide space.

For example, the command sequence "**^M05^BYU4015347^G**" will generate the following bar code, using the default ratio.



Code 128 A/B/C

Code 128 is a very high density, continuous, self-checking, variable length alphanumeric symbology. The Code 128 symbology is capable of encoding:

- the full 128 character ASCII character set
- four function code characters (FNC1, FNC2, FNC3, FNC4)
- four code set selection characters (Code A, Code B, Code C, Shift)
- three start characters (Start A, Start B, Start C)
- one stop character.

These characters are encoded using three alternate character sets, A, B, and C. Each set includes start codes and shift codes to control which set is to be used. A given bar/space pattern can have three different meanings, depending upon which character set is selected. The Imager will automatically optimize the way a bar code is printed, selecting the appropriate character set(s) to minimize the length of the bar code, and inserting the necessary start code, shift codes, and stop code. The Imager will also automatically insert a modulo 103 check digit.

Character set C contains the 100 two-digit pairs 00 to 99. When printing numeric data, the use of character set C effectively doubles the density of the bar code. The Imager shifts to character set C when four or more contiguous numeric digits are encountered.

A bar code may contain from 1 to 40 characters. The data can include any printable ASCII characters. The Imager does not provide a way to specify ASCII control codes, or Code 128 function codes.

The Code 128 symbology uses multiple element widths to encode characters. Each character has 11 modules, which may be either black or white, to create three bars and three spaces of varying width.

Code 128 bar codes require an eight digit ratio, with the digits representing:

- 1 module wide bar
- 1 module wide space
- 2 module wide bar
- 2 module wide space
- 3 module wide bar
- 3 module wide space
- 4 module wide bar
- 4 module wide space.

For example, the command sequence "**^M05^BYZABC123456^G**" will generate the following bar code, using the default ratio of 1:1:2:2:3:3:4:4.



The command sequence "**^M08^BY9Z22446688ABC123456^G**" will generate the following bar code, using double the default ratio.



UCC-128

UCC-128 is a variant of Code 128 that is used in retail distribution applications for serialized carton tracking. The standard Code 128 character set is used, except that every symbol begins with a Start C character, followed by a Function Code 1 character, and only numeric data is encoded. In addition to the modulo 103 check digit that is calculated from all the characters (excluding the stop code) in the bar code, a modulo 10 check digit is calculated for the numeric data.

The Imager automatically generates the Start C character, the Function Code 1 character, the two check digits, and the stop code. A UCC-128 bar code must contain 19 data digits; a modulo 10 check digit will be calculated for these 19 data digits. If 20 data digits are specified, the last digit will be interpreted as the modulo 10 check digit, and it will be checked for validity. The 19 digit data string must begin with "00" to be valid. If it begins with other digits, the Imager will print a Code 128 bar code instead of a UCC-128 bar code.

For example, the command sequence "**^M05^BY10000012345555555555^G**" will generate the following bar code, using the default ratio of 1:1:2:2:3:3:4:4.



PostNet

The Postal Numeric Encoding Technique (POSTNET) was developed by the U.S. Postal Service to provide an optimized bar code system for encoding ZIP Code information on letter mail.

POSTNET may be used to encode five digit ZIP Codes, nine digit ZIP+4 Codes, and 11 digit Advanced Bar Codes (ABC). ABC's encode a ZIP+4 Code plus a 2 digit Delivery Point.

A POSTNET symbol consists of a left frame bar; five, nine, or 11 data digits; a check digit; and a right frame bar. The Imager will automatically generate the frame bars and check digit; the programmer must supply the appropriate number of data digits.

Unlike other bar codes which use variable width bars to encode data, POSTNET uses variable height bars. Each numeric digit is encoded by a sequence of five bars, two of which are full height bars and three of which are half height bars.

For example, the command sequence "**^M01^BN212345^G**" will encode a five digit ZIP Code in the following bar code.



The command sequence "**^M01^BN2123456789^G**" will encode a nine digit ZIP+4 Code in the following bar code.



The command sequence "**^M01^BN212345678912^G**" will encode an 11 digit ABC in the following bar code.



Because POSTNET does not use variable width bars, and the absolute size of the bars is precisely specified by the U.S. Postal Service, variable ratio POSTNET bar codes should not be specified.

Modification and Positioning Commands

Half-Dots

Half-dot shading is created by printing an additional dot at each half-dot position, thereby doubling the number of dots printed horizontally and/or vertically. This provides high-resolution printing, and the increase in density results in a darker image.

The primary use of high-resolution is for enhancing the quality of printed bar codes; it will provide the best quality bar codes possible with the printer. The "downside" is that it forces the printer to print more slowly, increases print head wear, and depletes ribbon ink faster. Therefore, use high-resolution only when it is necessary to achieve the highest quality printer output.

The **^KF** command is used to activate and deactivate (toggle) horizontal high-resolution printing. The command format is:

^KF

(Note: All bar codes, both horizontal and vertical, are printed using horizontal high-resolution printing. This is necessary to guarantee readable bar codes of the best possible quality, and will be done whether or not any ^KF commands are issued.)

The **^KV** command is used to activate and deactivate (toggle) vertical high-resolution printing. The command format is:

^KV

(Note: Bar codes are printed without using vertical high-resolution printing, unless the ^KV command is used. Vertical high-resolution printing may make bar codes less readable, because the additional dots will force the bars out of spec.)

The command sequence "**^M0504000LIGHT^KFDARK^KVDARKER^KV^KF^-**" will print the text string "**LIGHT**" with half-dotting off, the text string "**DARK**" with horizontal half-dots, and the text string "**DARKER**" with horizontal and vertical half-dots.

LIGHTDARKDARKER

Half-dots will then be turned off. Half-dot shading may be toggled as many times as necessary within the pass. Half-dot shading will be turned off by the pass terminator, at the end of the pass. Half-dot shading may be applied to characters, the reverse image background of characters, boxes, lines, and bar codes. Half-dot shading may be combined with character half-tones to create even more shading options for characters.

Modifying Height

The **^H** command may be used to change the universal height value within a sequence of image commands. When this command is encountered, the universal height would have already been set with a **^M**, **^V**, **^E** or **^U** command. This command is available to change that value for expandable text or a bar code. The command format is:

^Hnn

where: *nn* = *New universal height in 1/10 inches; two digits from 00 to 99*
(e.g. 10 = 1.0 inch).

For example, the command sequence "**^M1006000TALL^H03SHORT^-**" will print the text string "**TALL**" with 1.0 inch high characters, followed by the text string "**SHORT**" with 0.3 inch high characters. Both text strings will be printed with characters that are 0.6 inch wide.



Height may be changed as many times as necessary within the pass. Height is always measured along the vertical axis of the paper, regardless of the orientation of the images being printed. Note that only height is affected by this command. Width remains unchanged.

Modifying Width

The **^W** command may be used to change the universal width value within a sequence of image commands. When this command is encountered, the universal width would have already been set with a **^M**, **^V**, **^E** or **^U** command. This command is available to change that value for expandable text. The command format is:

^Wnn

where: *nn* = *New universal width in 1/10 inches; two digits from 00 to 99*
(e.g. 10 = 1.0 inch).

For example, the command sequence "**^M1008000WIDE^W04NARROW^-**" will print the text string "**WIDE**" with 0.8 inch wide characters, followed by the text string "**NARROW**" with 0.4 inch wide characters. Both text strings will be printed with characters that are 1.0 inch high.

WIDENARROW

Width may be changed as many times as necessary within the pass. Width is always measured along the horizontal axis of the paper, regardless of the orientation of the images being printed. Note that only width is affected by this command. Height remains unchanged.

Modifying Justification (Vertical Positioning)

The **^J** command may be used to change the justification (vertical positioning) value within a sequence of image commands. It moves the vertical position for the next image down the distance specified from the top of the pass. When this command is encountered, the justification may have already been set with a **^M**, **^V**, **^E** or **^U** command. This command is available to change that value. The command format is:

^Jjd

where: *jjd* = Justification from the current position. Indicates how far down from the top of the pass the next image will begin printing.
jj = 1/10 inches; two digits from 00 to 99 (e.g. 10 = 1.0 inch).
d = Dots (1/72 inches); one digit from 0 to 9 (e.g. 7 = 7/72 inch).

For example, the command sequence "**^M0504000STAIR^J050STEP^J100DOWN^-**" will print the text strings "STAIR", "STEP", and "DOWN" with each string being 0.5 inch further down the page.

STAIR
 STEP
 DOWN

Justification may be changed as many times as necessary within the pass. Justification is always measured along the vertical axis of the paper, relative to the top of the pass, regardless of the orientation of the images being printed.

Horizontal Tab

Horizontal tabbing allows the programmer to set the horizontal print position for the next image to be defined. It is also possible to change the reference position for horizontal tabs from the left printable edge to some other horizontal position. Both of these functions are accomplished with a **^T** command, but one is accomplished in image mode and the other is accomplished in filter mode. There really are two separate commands, which both just happen to use the **^T** command mnemonic. Be careful to avoid confusing these two different commands.

Tabbing in Image Mode

In image mode, the **^T** command may be used to tab to a new horizontal print position within a sequence of image commands. It is an absolute tab that sets the horizontal position for the next image the distance specified from the left edge of the printable area. When an image is defined within an image sequence, the print position for the next image in the sequence begins at the right hand edge of the preceding image. The **^T** command should be used to position the cursor to the correct horizontal position for the next image. The command format is:

^T*hhhd*

where: *hhh* = Horizontal position in 1/10 inches;
 three digits from 000 to 136 (e.g. 050 = 5.0 inches).
 d = Horizontal position in dots (1/60 inches);
 one digit from 0 to 9 (e.g. 7 = 7/60 inch).

For example, the command sequence "**^M000000ZERO^T0400FOUR^-**" will print the text string "**ZERO**" at the left edge of the printable area, and the text string "**FOUR**" four inches from the left. Horizontal tabs may be used to correctly position each image in the pass.

ZERO

FOUR

The **^T** horizontal tab command will also reset the font selection to a non-high-resolution font. When using the **^S** command with the **^T** command, the **^S** must follow the **^T** to be effective. This is illustrated by the following two lines of code:

```
^M0101000^S1^T0100Standard Font^-  
^M0101000^T0100^S1High-Resolution Font^-
```

Horizontal tabs always move along the horizontal axis of the paper, regardless of the orientation of the images being printed. Horizontal tabs are usually specified relative to the left edge of the printable area, but this can be changed by the **^T** command in filter mode. Do not confuse the **^T** command in image mode (which has been described here) with the **^T** command in filter mode.

Modifying Reference Position in Filter Mode

In filter mode, the **^T** command is a universal "horizontal tab" command. It is used to set a tab value which is added to all subsequent horizontal tab values used in image mode. The effect is to change the reference position for horizontal tabs in image mode, so they are no longer specified relative to the left edge of the printable area. This allows the horizontal placement of printed information to be "adjusted" without having to change all the individual image mode tab commands. (The reference position will change for all objects created in image mode, whether or not they have been positioned by a horizontal tab in image mode.) The command format is:

^T

where: *hhh* = Horizontal reference position in 1/10 inches;
 three digits from 000 to 136 (e.g. 050 = 5.0 inches).
 d = Horizontal reference position in dots (1/60 inches);
 one digit from 0 to 9 (e.g. 7 = 7/60 inch).

For example, the command sequence "**^T0100^M0000000ZERO^T0400FOUR^-**" will print the text string "**ZERO**" one inch, not zero inches, from the left edge of the printable area, and the text string "**FOUR**" five inches, not four inches, from the left.

ZERO

FOUR

Do not confuse the **^T** command in filter mode (which has been described here) with the **^T** command in image mode.

Carriage Return

In filter mode, the **^-** command (hat-dash) will generate a carriage return control code (0Dh). If free format is on, this is the only way to send a carriage return to the printer.

In image mode, the **^-** command acts as a pass terminator. It will force any defined images to print, and return to filter mode, but it will not generate a carriage return.

Line Feed

In filter mode, the **^*** command (hat-star) will generate a line feed control code (0Ah). If free format is on, this is the only way to send a line feed to the printer.

In image mode, the **^*** command acts as a pass terminator. It will force any defined images to print, and return to filter mode, but it will not generate a line feed. In image mode, the **^*** command is functionally equivalent to the **^-** carriage return command.

Form Feed

In filter mode, the ^, command (hat-comma) will generate a form feed control code (0Ch). If free format is on, this is the only way to send a form feed to the printer.

In image mode, the ^, command acts as a pass terminator. It will force any defined images to print, and return to filter mode, but it will not generate a form feed. In image mode, the ^, command is functionally equivalent to the ^- carriage return command.

Line Slew

The **^K** and **^W** commands may be used to vertically slew a specified number of lines. Each command has a parameter which represents the number of line feeds to generate. The actual distance slewed will depend upon the line spacing that is set in the printer, typically 1/6 inch or 1/8 inch per line. The **^K** and **^W** commands are identical and may be used interchangeably. The command format is:

^Knn
or
^Wnn

where: *nn* = Number of line feeds to generate; two digits from 01 to 99.

For example, the command sequence "**^K02^-**" will skip two lines. The command sequence "**^W02^-**" will do the same.

These commands must be used in filter mode; they may not be embedded in a sequence of image commands in image mode.

Dot Slew

The **^D** command may be used to vertically slew a specified number of dots (1/72 inches). The command format is:

^Dnn

where: *nn* = Number of dots to slew; two digits from 01 to 99.

For example, the command sequence "**^D72^-**" will skip one inch. This command must be used in filter mode; it may not be embedded in a sequence of image commands in image mode. Printing text without entering image mode to do so after using this command may cause the printer to "re-align" itself to the next print line after the text is printed.

Graphics

Graphics images may be generated in two different ways: using commands that are native to the ImagerPlus in image mode, or using commands that are native to the FormsPro 4000/4003, with the Imager in pass-thru mode.

ImagerPlus Graphics (ImagerPlus Only)

The **^Q** command is used to print logos and other "bitmapped" graphics images. These graphics images are represented in a 7 bit vertical graphics format.

Graphics images are printed horizontally across a label in "bands". Each band is 7 dots high by as many dots wide as necessary to print across a complete graphics image. If a graphics image is taller than 7 dots, then printing multiple bands will be necessary to print the entire graphics image.

Each column of 7 dots in a band is represented by pairs of characters. These characters are comprised of the characters '0' through '9' and 'A' through 'F' (16 possible characters). This representation is referred to as **pseudo-hexadecimal** or "four-bit hex". That is, the characters give the appearance of being hexadecimal characters but they are actually 8 bit ASCII characters.

The character pair which would activate all 7 dots (print black) in a column would be the pair '7F'. The character pair which would deactivate all 7 dots (print white) would be the pair '00'. The value required to activate the dots in a column is determined by assigning a different value to each dot in a column. The values assigned to each dot, from the bottom of a column to the top, is **1, 2, 4, 8, 16, 32, and 64**. The bottom dot would be activated with a value of decimal 1. The top dot would be activated with a value of decimal 64. To turn on any (or all) of the other dots, simply add their values together. The value required to turn on all 7 dots would be 1+2+4+8+16+32+64 or 127. Converting decimal 127 to hexadecimal (or pseudo-hexadecimal) would yield '7F' as the pair of characters to use.

The command format is:

^Qxx,....,xx^G

Where: *xx* = 7 bit pseudo-hex pair.

^G = Graphics terminator.

NOTE: Commas may be used between the character pairs in order to make reading easier but they are not required.

Example: The following code would print 2 "bands" of an image which forms a "left pointing arrow".

```

^PY^-          activate filter mode
^F             absorb carriage control
^M0000000^T0000^Q0103070F1F3F7F070707070707^G      band #1
^M0000007^T0000^Q00406070787C7E606060606060^G      band #2
^-            terminate image pass
^O            deactivate the ^F command
^PN^-        return to pass-thru mode
    
```

Using a '*' for a dot and a 'O' for a white space, the above example would look something like the following:

<u>value</u>	<u>bit map</u>	
64	000000*0000000	-start band #1
32	00000**0000000	
16	0000***0000000	
8	000****0000000	
4	00*****0000000	
2	0*****0000000	
1	*****0000000	
64	0*****0000000	-start band #2
32	00*****0000000	
16	000****0000000	
8	0000***0000000	
4	00000**0000000	
2	000000*0000000	
1	00000000000000	

When actually printed, the following result is obtained.



The ^Q graphics command must be contained within an image sequence. Any prefix (^M, ^V, ^E, ^U) may be used to enter image mode before issuing a ^Q command; the implied orientation and any universal height and width specifications will not affect the graphics command (but it will affect characters contained in the pass).

Graphics dots will always be printed at a resolution of 60x72 dpi. Graphics images are drawn at the current print position. Horizontal tabbing and justification commands may be used to reach the desired print position.

FormsPro 4000/4003 Graphics

The FormsPro 4000/4003 supports 8 bit vertical graphics in its resident Epson®, Proprinter®, and Printek emulations. There are several good reasons to use the native FormsPro 4000/4003 graphics commands instead of the ImagerPlus commands:

- If you are using an Imager, not an ImagerPlus, there is no choice. The standard Imager does not support the ^Q graphics command.
- The native FormsPro graphics commands are more efficient and flexible than the ImagerPlus ^Q command. An 8 bit column is specified instead of a 7 bit column, and only one byte instead of two is required to represent each column. This can cut the amount of graphics data by more than half, potentially improving throughput. And more graphics densities are available.
- Many software programs have drivers for Epson and Proprinter; this graphics format has become a de facto industry standard.

There is also a very good reason not to use native FormsPro graphics in conjunction with an Imager board. Graphics data tends to be very random in nature; some of the data bytes may appear to be Imager commands. All it takes is a hex '5E' graphics data byte (the '^' image control character) to send the Imager on a rampage; the actual results will depend upon the random graphics data that follows this byte.

To minimize the chance of triggering the Imager with random graphics data, make sure the Imager is in pass-thru mode, not filter mode, before sending native FormsPro graphics commands. Toward this end, make sure that "Translation: Off" is specified in the "Setup: OPTIONS" menu on the printer's control panel. In pass-thru mode, the random graphics data must contain a "^PY" sequence before problems can occur. The chance of three consecutive bytes being "^PY" is relatively low (but not impossible). If a "^PY" does by chance occur, then all it takes is a single '^' to start causing trouble, and the chance of that is relatively high.

For more information on the native FormsPro 4000/4003 graphics commands, see the **FormsPro 4000 Programmer's Manual**.

Repetitive Printing

The Imager is capable of printing multiple copies of an image. The data for the image needs to be supplied only once. Vertical repetition may be used to print from 1 to 9999 copies of the image down the page. Horizontal repetition may be used to print copies of the image across the page. Each copy of the image will be identical, except that numeric fields may be incremented or decremented when repeating an image vertically.

Vertical Repetition

The **^R** command is used to define an image repeat "loop" to print multiple copies of an image vertically down the page. Commands and other data which come after this command and occur prior to its terminator, will be printed multiple times. The command will cause continuous printing until the repeat "count" has been satisfied. "Repeat" commands may be "nested". That is, a repeat command may contain other repeat commands. The maximum depth for the repeats is 10 levels. The repeat terminator is the **^Z** "terminator" command. If repeat commands are nested, one **^Z** command will terminate all repeats. Incrementing numbers on an inside and outside loop increment at the same rate. The command format is:

^Rnnnn

where: *nnnn* = Repeat count; four digits from 0001 to 9999.

For example, the following repeat loops:

^PY^-	<i>activate filter mode</i>
^F	<i>eat all carriage control</i>
^R0003	<i>set outside repeat count to 3</i>
^_ ^*	<i>CR and LF</i>
OuterLoop	<i>text to print 3 times</i>
^R0005	<i>set inside repeat count to 5</i>
InnerLoop	<i>text to print 15 times</i>
^Z	<i>repeat terminator</i>
^O	<i>turn ^F command OFF</i>
^PN^-	<i>return to pass-thru mode</i>

Will generate the following print:

```
OuterLoopInnerLoopInnerLoopInnerLoopInnerLoopInnerLoop
OuterLoopInnerLoopInnerLoopInnerLoopInnerLoopInnerLoop
OuterLoopInnerLoopInnerLoopInnerLoopInnerLoopInnerLoop
```

Auto Increment/Decrement

The **^Y** command is used to automatically increment or decrement numeric fields. These numeric fields may be text, bar code data or other command parameters. Use of this command is most powerful when used inside of a repeating sequence which is controlled by the **^R** ("repeat") command or the **^B** ("buffered overlay") command. The command format is:

^Ynnnsiii^G

where: *nnn* = Numeric data to be modified.
s = Sign; + = increment, - = decrement.
iii = Increment/decrement value.
^G = Command terminator.

The following example will print five Code 39 bar codes. The first bar code will have the value 1234 and each successive bar code value will be incremented by 1. The last bar code will have a value of 1238.

^PY^-	<i>activate filter mode</i>
^F	<i>absorb all carriage control</i>
^R0005	<i>set repeat loop for 5 times</i>
^M	<i>activate image mode</i>
05	<i>set height to 0.5 inches</i>
^BYA	<i>define a bar code</i>
^Y1234+1^G	<i>define an increment field</i>
^G	<i>terminate the bar code</i>
^_	<i>terminate image mode and print</i>
^_^*	<i>CR and LF</i>
^Z	<i>repeat until 5 have printed</i>
^O	<i>deactivate the ^F command</i>
^PN^-	<i>return to pass-thru mode</i>

The result is shown below.



Horizontal Repetition

The `^S` command is used to "spread" label images across a page. It is used to make identical copies of label images from a single label image. This command may have two parameters or it may have no parameters. If it is used without parameters, then it deactivates the current "spread" function. The command format is:

`^Snnww`

where: *nn* = Number of times to "replicate" the label.

ww = Width of the label to copy in 1/10 inches.

Includes inter-label gap.

For example, the command sequence "`^S0320^M05^BYA123^G^-^S^-`" will produce the following result.



Buffered Overlay

The buffer overlay definition command is used to define the beginning of a block of data which describes a label image. When the Imager encounters the ^B command, it will continue accepting data until a "buffer overlay termination" command (^]) is found. All data between the ^B and the ^] will be saved and processed as a label. Other filter mode and image mode commands may be in this buffer overlay sequence. The data which "fills out" the label image is supplied immediately following the ^] command. The command has no parameters and has the following format:

^B

An example of buffer overlay follows.

^PY^-	<i>activate filter mode</i>
^F	<i>eat all carriage control</i>
A BUFFER OVERLAY	<i>text to print one time</i>
^B	<i>start buffer overlay</i>
FIRST NAME: (15 BYTES) =	<i>constant text</i>
^[015	<i>variable field #1</i>
^_ ^*	<i>CR and LF</i>
LAST NAME: (20 BYTES) =	<i>constant text</i>
^[020	<i>variable field #2</i>
^_ ^* ^*	<i>CR and 2 LFs</i>
^]	<i>buffer overlay terminator</i>
JOE^-	<i>variable data - field #1</i>
BLOW^-	<i>variable data - field #2</i>
FRED^-	<i>variable data - field #1</i>
SMITH^-	<i>variable data - field #2</i>
LISA^-	<i>variable data - field #1</i>
JONES^-	<i>variable data - field #2</i>
^G	<i>terminate variable data</i>
^O	<i>turn ^F command OFF</i>
^PN^-	<i>return to pass-thru mode</i>

It will produce the following result.

```
A BUFFER OVERLAY

FIRST NAME: (15 BYTES) = JOE
LAST NAME:  (20 BYTES) = BLOW

FIRST NAME: (15 BYTES) = FRED
LAST NAME:  (20 BYTES) = SMITH

FIRST NAME: (15 BYTES) = LISA
LAST NAME:  (20 BYTES) = JONES
```

Normally, the variable data fields must be the same size as what the "variable field descriptor" commands (^[]) define. However, a ^-, ^* or ^, within the variable data will pad the rest of the field with spaces. With the ^* or ^, the rest of the fields in the buffer will also be padded with spaces. When a ^G is encountered in the variable data, the buffered overlay processing is terminated.

There are two optional commands which can be used in conjunction with the "buffered overlay" command sequence and which provide additional data manipulation capabilities: The ^R ("define variable repeat loop") command (not to be confused with the regular filter mode "define repeat loop" command) and the ^C ("block copy") command. These two special "buffer overlay" commands are mutually exclusive. That is, they cannot be used within the same buffer overlay (label).

The ^R command used in conjunction with the ^B command, unlike the regular filter mode "define repeat loop" command, has no associated parameters. It performs a similar function of "repeating" a command sequence but the repeat count is supplied as part of the variable data immediately following the "buffer overlay termination" command.

The following buffer overlay example contains a repeat.

^PY^-	<i>activate filter mode</i>
^F	<i>eat all carriage control</i>
A BUFFER OVERLAY & REPEAT <i>data to print one time</i>	
^B	<i>start buffer overlay</i>
^R	<i>start repeat sequence</i>
FIRST NAME: (15 BYTES) =	<i>constant text</i>
^[015	<i>variable field #1</i>
^_^*	<i>CR and LF</i>
LAST NAME: (20 BYTES) =	<i>constant text</i>
^[020	<i>variable field #2</i>
^_^*^*	<i>CR and 2 LFs</i>
^Z	<i>terminate the ^R command</i>
^]	<i>buffer overlay terminator</i>
0002	<i>print field #1 & #2 twice</i>
JOE^-	<i>variable data field #1</i>
BLOW^-	<i>variable data field #2</i>
0003	<i>print field #1 & #2 3 times</i>
FRED^-	<i>variable data field #1</i>
SMITH^-	<i>variable data field #2</i>
0001	<i>print field #1 & #2 once</i>
LISA^-	<i>variable data field #1</i>
JONES^-	<i>variable data field #2</i>
0000	<i>no print</i>
^G	<i>end of data</i>
^O	<i>turn ^F command OFF</i>
^PN^-	<i>return to pass-thru mode</i>

It will produce the following result.

```
A BUFFER OVERLAY & REPEAT

FIRST NAME: (15 BYTES) = JOE
LAST NAME:  (20 BYTES) = BLOW

FIRST NAME: (15 BYTES) = JOE
LAST NAME:  (20 BYTES) = BLOW

FIRST NAME: (15 BYTES) = FRED
LAST NAME:  (20 BYTES) = SMITH

FIRST NAME: (15 BYTES) = FRED
LAST NAME:  (20 BYTES) = SMITH

FIRST NAME: (15 BYTES) = FRED
LAST NAME:  (20 BYTES) = SMITH

FIRST NAME: (15 BYTES) = LISA
LAST NAME:  (20 BYTES) = JONES
```

The ^C command used in conjunction with the ^B command has a two digit parameter. It performs a code duplication service. All code occurring after the ^C command and prior to the ^Z "termination" command, is duplicated the number of times defined by the ^C parameter.

The following buffer overlay example contains a block copy.

^PY^-	<i>activate filter mode</i>
^F	<i>eat all carriage control</i>
A BUFFER OVERLAY & COPY	<i>data to print one time</i>
^B	<i>start buffer overlay</i>
^C02	<i>duplicate to ^Z 2 times</i>
FIRST NAME: (15 BYTES) =	<i>constant text</i>
^[015	<i>variable field #1</i>
^_^*	<i>CR and LF</i>
^Z	<i>terminate the ^C command</i>
LAST NAME: (20 BYTES) =	<i>constant text</i>
^[020	<i>variable field #2</i>
^_^*^*	<i>CR and 2 LFs</i>
^]	<i>buffer overlay terminator</i>
JOE^-	<i>variable data field #1</i>
E.^-	<i>variable data field #2</i>
BLOW^-	<i>variable data field #3</i>
FRED^-	<i>variable data field #1</i>
F.^-	<i>variable data field #2</i>
SMITH^-	<i>variable data field #3</i>
LISA^-	<i>variable data field #1</i>
J.^-	<i>variable data field #2</i>
JONES^-	<i>variable data field #3</i>
^G	<i>end of data</i>
^O	<i>turn ^F command OFF</i>
^PN^-	<i>return to pass-thru mode</i>

It will produce the following result.

```

A BUFFER OVERLAY & COPY

FIRST NAME: (15 BYTES) = JOE
FIRST NAME: (15 BYTES) = E.
LAST NAME: (20 BYTES) = BLOW

FIRST NAME: (15 BYTES) = FRED
FIRST NAME: (15 BYTES) = F.
LAST NAME: (20 BYTES) = SMITH

FIRST NAME: (15 BYTES) = LISA
FIRST NAME: (15 BYTES) = J.

```

LAST NAME: (20 BYTES) = JONES

Image Interrupt

The **^I** "interrupt" command is used to "interrupt" the current image mode processing sequence and start another one. It allows a second image mode sequence to begin printing while the previous image mode sequence is still printing. Originally, this command was employed because printer graphic controllers lacked the memory to process larger image mode sequences. This command allowed the user to "get around" the memory limitation. Today, this command is not necessary and its use is not recommended. The effect this command has on the image mode sequence following the **^I** command is to alter its vertical cursor position by the sum of the interrupt value plus the current image mode sequence's origin. The interrupt process is terminated when the **^I000** sequence is encountered at the end of an image mode sequence.

The command format is:

^Ittd

where: *tt* = 1/10 inches to change the vertical position.
d = Dots (1/72 inches) to change the vertical position.

The following code illustrates the use of the interrupt command.

^PY^-	<i>activate filter mode</i>
^F	<i>absorb all carriage control</i>
^M	<i>activate image mode</i>
05	<i>set height to 0.5 inches</i>
05	<i>set width to 0.5 inches</i>
000	<i>set justification to 0</i>
TEXT	<i>text to print</i>
^I020	<i>interrupt after printing 2/10 inches</i>
^-	<i>terminate image mode; start printing</i>
^M	<i>activate image mode (next sequence)</i>
^T0200	<i>position (tab) right 2 inches</i>
MORE TEXT	<i>text to print</i>
^I000	<i>turn off interrupts</i>
^-	<i>terminate image mode; start printing</i>
^O	<i>turn off the ^F command</i>
^PN^-	<i>return to pass-thru mode</i>

It will produce the following result.

TEXT MORE TEXT

Reference Section

Pass-Thru Mode Introduction

In pass-thru mode, the Imager simply passes all data received on through to the printer. This may be thought of as passive processing, or you may think of the Imager as being turned off. The only command recognized in pass-thru mode is the **^PY** command.

Pass-Thru Mode Command List

^PY	Activate filter mode processing.
-----	----------------------------------

^PY command

This command is used to activate filter mode. This string must be the first printable characters (preceding spaces do not count) on a new line to be recognized. This command has no parameters. The command format is:

^PY

Filter Mode Introduction

In order to activate filter mode, the command **^PY** must be sent to the printer or "Translation: On" must be specified in the "Setup: Options" menu on the printer's control panel. If "Translation: On" has been set, the Imager will be initialized to filter mode on power-up or printer reset. In this case, the **^PN** command will not reset the Imager to pass-thru mode; that can only be accomplished by changing the "Translation" value through the printer's control panel.

Filter mode commands are divided in to two catagories: "printer control" commands and "regular filter mode" commands. The printer control commands are simple, single character commands which, when encountered, send a single, non-printable character to the printer. The regular filter mode commands perform more elaborate operations on the incoming data.

Filter Mode "Printer Control" Commands

Printer control commands are used to send non-printable single characters to the printer. These include the "escape" character, carriage return, line feed, form feed plus numerous other characters which the user may desire to send to the printer. "Printer control" commands are normally used in conjunction with the "regular filter mode" commands.

The following example could be used to send a form feed to the printer:

<code>^PY^-</code>	<i>activate filter mode</i>
<code>^F</code>	<i>absorb carriage control characters</i>
<code>^,</code>	<i>generate a form feed (hex 0C)</i>
<code>^O</code>	<i>turn the ^F command OFF</i>
<code>^PN^-</code>	<i>return to pass-thru mode</i>

Filter Mode "Printer Control" Commands List

^SP	(caret space) generates hex '00' (null)
^!	generates hex '01'
^"	generates hex '02'
^#	generates hex '03'
^\$	generates hex '04'
^%	generates hex '05'
^&	generates hex '06'
^'	generates hex '07'
^(generates hex '08'
)	generates hex '09'
^*	generates hex '0A' (line feed)
^+	generates hex '0B'
^,	generates hex '0C' (form feed)
^-	generates hex '0D' (carriage return)
^.	generates hex '0E'
^/	generates hex '0F'
^0	generates hex '10'
^1	generates hex '11'
^2	generates hex '12'
^3	generates hex '13'
^4	generates hex '14'
^5	generates hex '15'
^6	generates hex '16'
^7	generates hex '17'
^8	generates hex '18'
^9	generates hex '19'
^:	generates hex '1A'
^;	generates hex '1B' (escape)
^<	generates hex '1C'
^=	generates hex '1D'
^>	generates hex '1E'
^?	generates hex '1F'

Filter Mode "Regular" Commands List

^A	Begin acknowledging input characters again. Used after the '^X' command.
^B	Begin definition of a buffered overlay sequence. Must be terminated with a '^]' command.
^D	Define the number of vertical dots to slew (dot-feed). Dots are 72 per inch.
^E	Begin image mode processing <u>and</u> define vertical (bottom to top) expandable characters width and height.
^F	Activate "free-format" processing; absorb carriage control characters.
^K	Define the number of vertical lines to slew (line feed).
^L	Define the height of a label in number of lines.
^M	Begin image mode processing <u>and</u> define upright (normal) expandable characters height and width.
^N	Define a new control character (normally the '^' until changed).
^O	Deactivate "free-format" processing; quit absorbing carriage control characters.
^PN	Deactivate filter mode processing and reactivate pass-thru mode processing.
^R	Begin repeat loop definition. Loop must be terminated by a '^Z'.
^S	Define horizontal duplicate (spread).
^T	Modify the horizontal cursor absolute ordinate (absolute tab).
^U	Begin image mode processing <u>and</u> define upside-down expandable characters height and width.
^V	Begin image mode processing <u>and</u> define vertical (top-to-bottom) expandable characters width and height.
^W	Define the number of vertical lines to slew (line feed).
^X	Absorb all incoming data until a '^A' command is encountered.
^Z	The "repeat loop" command ('^R') terminator.
^]	Buffered overlay sequence terminator. Sequence must begin with a '^B' command.

^A command

This command will cause all subsequent characters to not be ignored, thereby counteracting any ^X command that was issued previously. When the ^A command is encountered, the Imager will stop absorbing or "eating" all incoming data and will begin examining the data again for other filter mode commands. The command format is:

^A

^B command

The buffer overlay definition command is used to define the beginning of a block of data which describes a label image. When the Imager encounters the ^B command, it will continue accepting data until a "buffer overlay termination" command (^]) is found. All data between the ^B and the ^] will be saved and processed as a label. Other filter mode and image mode commands may be in this buffer overlay sequence. The data which "fills out" the label image is supplied immediately following the ^] command. The command has no parameters and has the following format:

^B

Normally, the variable data fields must be the same size as what the "variable field descriptor" commands (^[]) define. However, a ^-, ^* or ^, within the variable data will pad the rest of the field with spaces. With the ^* or ^,, the rest of the fields in the buffer will also be padded with spaces. When a ^G is encountered in the variable data, the buffered overlay processing is terminated.

There are two optional commands which can be used in conjunction with the "buffered overlay" command sequence and which provide additional data manipulation capabilities: The ^R ("define variable repeat loop") command (not to be confused with the regular filter mode "define repeat loop" command) and the ^C ("block copy") command. These two special "buffer overlay" commands are mutually exclusive. That is, they cannot be used within the same buffer overlay (label).

The ^R command used in conjunction with the ^B command, unlike the regular filter mode "define repeat loop" command, has no associated parameters. It performs a similar function of "repeating" a command sequence but the repeat count is supplied as part of the variable data immediately following the "buffer overlay termination" command.

The ^C command used in conjunction with the ^B command has a two digit parameter. It performs a code duplication service. All code occurring after the ^C command and prior to the ^Z "termination" command, is duplicated the number of times defined by the ^C parameter.

^D command

This command may be used to vertically slew a specified number of dots (1/72 inches). The command format is:

^Dnn

where: *nn* = Number of dots to slew; two digits from 01 to 99.

^E command

This command is used to print large characters in a "sideways - bottom-to-top" orientation. The universal height, width, and vertical positioning (justification) are specified, followed by the characters to be printed. The command format is:

^Ehhwwjjdc...c

where: *hh* = Height of characters in 1/10 inches; two digits from 00 to 99.
Represents the physical (vertical) height. Since the characters are sideways, it really specifies what would classically be thought of as character width. Includes inter-character gap.

ww = Width of characters in 1/10 inches; two digits from 00 to 99.
Represents the physical (horizontal) width. Since the characters are sideways, it really specifies what would classically be thought of as character height.

jjd = Justification from the current position. Indicates how far down from the top of the pass the characters will begin printing.

jj = 1/10 inches; two digits from 00 to 99.

d = Dots (1/72 inches); one digit from 0 to 9.

c...c = Character or characters to be printed.

This command will activate image mode. The parameters of this command are optional, but will be processed if they exist. If while processing the parameters, an invalid parameter such as a non-digit character is encountered, that parameter plus any remaining unprocessed parameters are assumed to be zero.

This command is also used to print small characters in a "sideways - bottom-to-top" orientation. If the height and width values are greater than "01", then the active font will be the expandable font with the height and width assigned by this command. However, if the values are either "00" or "01", then non-expandable fonts are selected. The fonts are:

Regular Image Mode Fonts		
<i>ww</i>	<i>hh</i>	Selected font
00	00	7.5 cpi
01	01	10 cpi
00	01	12 cpi
01	00	15 cpi

^F command

This command will turn on free format mode, causing all carriage-control commands to be absorbed so that they will not be acted on by the printer. The command format is:

^F

^G command

This command is the general purpose "terminating" command for certain other filter mode and image mode commands. It has no parameters and has no meaning unless another command requires its use. This terminator is normally used with commands that have a variable (non-fixed) number of parameters. The command format is:

^G

^H command

This command is used to set the "height" of a label. If this command (or the ^L command) is not issued, the assumed height of the label is 66 lines. The command format is:

^Hnn

where: *nn* = Number of lines for the label; two digits from 01 to 99.

This command is identical in operation to the ^L command. The two commands are interchangeable.

^K command

This command may be used to vertically slew a specified number of lines. The command has one parameter which represents the number of line feeds to generate. The actual distance slewed will depend upon the line spacing that is set in the printer, typically 1/6 inch or 1/8 inch per line. This command is identical to the filter mode ^W command. The command format is:

^Knn

where: *nn* = Number of line feeds to generate; two digits from 01 to 99.

^L command

This command is used to set the "height" of a label. If this command (or the ^H command) is not issued, the assumed height of the label is 66 lines. The command format is:

^Lnn

where: *nn* = Number of lines for the label; two digits from 01 to 99.

This command is identical in operation to the ^H command. The two commands are interchangeable.

^M command

This command is used to print large characters in a "normal - left-to-right" orientation. The universal height, width, and vertical positioning (justification) are specified, followed by the characters to be printed. The command format is:

^Mhhwwjjdc...c

where: *hh* = Height of characters in 1/10 inches; two digits from 00 to 99.

ww = Width of characters in 1/10 inches; two digits from 00 to 99.
Includes inter-character gap.

jjd = Justification from the current position. Indicates how far down from the top of the pass the characters will begin printing.

jj = 1/10 inches; two digits from 00 to 99.

d = Dots (1/72 inches); one digit from 0 to 9.

c...c = Character or characters to be printed.

This command will activate image mode. The parameters of this command are optional, but will be processed if they exist. If while processing the parameters, an invalid parameter such as a non-digit character is encountered, that parameter plus any remaining unprocessed parameters are assumed to be zero.

This command is also used to print small characters in a "normal - left-to-right" orientation. If the height and width values are greater than "01", then the active font will be the expandable font with the height and width assigned by this command. However, if the values are either "00" or "01", then non-expandable fonts are selected. The fonts are:

<i>ww</i>	<i>hh</i>	Selected font
00	00	7.5 cpi
01	01	10 cpi
00	01	12 cpi
01	00	15 cpi

^N command

This command is used to change the image control character to any other printable character. If you change the image control character, be sure that all subsequent image commands use the new control character. If an inappropriate character (i.e. one that appears in the data stream for other reasons) is set as the image control character, unpredictable printing results will occur. The command format is:

^N*c*

where: *c* = *New image control character; any printable character.*

^O command

This command will turn off free format mode if it has been activated by the **^F** command. If free format mode is not active, this command will have no effect. After this command has been encountered, all carriage control characters will be processed by the printer. The command format is:

^O

^PN command

This command is used to terminate filter mode and return to pass-thru mode. The command format is:

^PN

^R command

This command is used to define an image repeat "loop" to print multiple copies of an image vertically down the page. Commands and other data which come after this command and occur prior to its terminator, will be printed multiple times. The command will cause continuous printing until the repeat "count" has been satisfied. "Repeat" commands may be "nested". That is, a repeat command may contain other repeat commands. The maximum depth for the repeats is 10 levels. The repeat terminator is the ^Z "terminator" command. If repeat commands are nested, one ^Z command will terminate all repeats. The command format is:

^Rnnnn

where: *nnnn* = Repeat count; four digits from 0001 to 9999.

^S command

This command is used to "spread" label images across a page. It is used to make identical copies of label images from a single label image. This command may have two parameters or it may have no parameters. If it is used without parameters, then it deactivates the current "spread" function. The command format is:

^Snnww

where: *nn* = Number of times to "replicate" the label.
ww = Width of the label to copy in 1/10 inches.

^T command

In filter mode, the ^T command is a universal "horizontal tab" command. It is used to set a tab value which is added to all subsequent horizontal tab values used in image mode. The effect is to change the reference position for horizontal tabs in image mode, so they are no longer specified relative to the left edge of the printable area. It allows the horizontal placement of printed information to be "adjusted" without having to change all other individual image mode tab commands. The command format is:

^Thhhd

where: *hhh* = Horizontal reference position in 1/10 inches;
three digits from 000 to 136.
d = Horizontal reference position in dots (1/60 inches);
one digit from 0 to 9.

^U command

This command is used to print large characters in an "upside down - right-to-left" orientation. The universal height, width, and vertical positioning (justification) are specified, followed by the characters to be printed. The command format is:

^Uhhwwjjdc...c

where: *hh* = Height of characters in 1/10 inches; two digits from 00 to 99.
ww = Width of characters in 1/10 inches; two digits from 00 to 99.
Includes inter-character gap.
jjd = Justification from the current position. Indicates how far down from the top of the pass the characters will begin printing.
jj = 1/10 inches; two digits from 00 to 99.
d = Dots (1/72 inches); one digit from 0 to 9.
c...c = Character or characters to be printed.

This command will activate image mode. The parameters of this command are optional, but will be processed if they exist. If while processing the parameters, an invalid parameter such as a non-digit character is encountered, that parameter plus any remaining unprocessed parameters are assumed to be zero.

This command is also used to print small characters in a "upside down - right-to-left" orientation. If the height and width values are greater than "01", then the active font will be the expandable font with the height and width assigned by this command. However, if the values are either "00" or "01", then non-expandable fonts are selected. The fonts are:

Regular Image Mode Fonts

<i>ww</i>	<i>hh</i>	Selected font
00	00	7.5 cpi
01	01	10 cpi
00	01	12 cpi
01	00	15 cpi

^V command

This command is used to print large characters in a "sideways - top-to-bottom" orientation. The universal height, width, and vertical positioning (justification) are specified, followed by the characters to be printed. The command format is:

^Vhhwwjjdc...c

- where: *hh* = Height of characters in 1/10 inches; two digits from 00 to 99.
Represents the physical (vertical) height. Since the characters are sideways, it really specifies what would classically be thought of as character width. Includes inter-character gap.
- ww* = Width of characters in 1/10 inches; two digits from 00 to 99.
Represents the physical (horizontal) width. Since the characters are sideways, it really specifies what would classically be thought of as character height.
- jjd* = Justification from the current position. Indicates how far down from the top of the pass the characters will begin printing.
- jj* = 1/10 inches; two digits from 00 to 99.
- d* = Dots (1/72 inches); one digit from 0 to 9.
- c...c* = Character or characters to be printed.

This command will activate image mode. The parameters of this command are optional, but will be processed if they exist. If while processing the parameters, an invalid parameter such as a non-digit character is encountered, that parameter plus any remaining unprocessed parameters are assumed to be zero.

This command is also used to print small characters in a "sideways - top-to-bottom" orientation. If the height and width values are greater than "01", then the active font will be the expandable font with the height and width assigned by this command. However, if the values are either "00" or "01", then non-expandable fonts are selected. The fonts are:

Regular Image Mode Fonts

<i>ww</i>	<i>hh</i>	Selected font
00	00	7.5 cpi
01	01	10 cpi
00	01	12 cpi
01	00	15 cpi

^W command

This command may be used to vertically slew a specified number of lines. The command has one parameter which represents the number of line feeds to generate. The actual distance slewed will depend upon the line spacing that is set in the printer, typically 1/6 inch or 1/8 inch per line. This command is identical to the filter mode ^K command. The command format is:

^Wnn

where: *nn* = Number of line feeds to generate; two digits from 01 to 99.

^X command

This command will cause all subsequent characters to be ignored, until a ^A command is received. If a ^A command is never received, then none of the incoming data will be processed or printed. When ignore character is active, all data will be absorbed or "eaten". Except for the ^A command, no commands are recognized while ignoring characters. The command format is:

^X

^Y command

This command is used to automatically increment or decrement numeric fields. These numeric fields may be text, bar code data or other command parameters. Use of this command is most powerful when used inside of a repeating sequence which is controlled by the ^R ("repeat") command or the ^B ("buffered overlay") command. The command format is:

^Ynnnsiii^G

where: *nnn* = Numeric data to be modified.
s = Sign; + = increment, - = decrement.
iii = Increment/decrement value.
^G = Command terminator.

^Z command

This command is the filter mode repeat (^R) and block copy (^C) command terminator. The command format is:

^Z

^[command

This command is the "variable field descriptor" command. It is used to define a variable field for data to be merged into. Use of this command without first issuing a **^B** ("buffered overlay") command would be meaningless. The command format is:

^[*nnn*

where: *nnn* = *Maximum length of the variable data field.*

Image Mode Introduction

Image mode processing is the "graphics" mode of operation for the Imager. In order to activate image mode, one of the commands **^M**, **^E**, **^U** or **^V** must be used. To exit image mode and return to filter mode, one of the commands **^-**, **^*** or **^**, is required if the **^F** command was previously issued. If the **^F** command was not previously issued, then a carriage return, line feed or form feed will also cause an exit to filter mode.

Image Mode Commands List

^A	Begin acknowledging input characters again. Used after the '^X' command.
^B	Define a horizontal bar code.
^C	Define a vertical bar code.
^D	Toggle the expandable lower case characters with descender on or off (depending on the previous state).
^E	Define the current vertical (bottom to top) expandable characters.
^G	The general purpose command sequence terminator. Used with commands which use a variable number of parameters such as "bar code" commands and "auto-increment" commands.
^H	Change the current expandable character height.
^I	Define the interrupt position. No longer required but still valid.
^J	Change the vertical cursor position.
^Kx	Defines the character and line shading patterns. Valid list is: KF, KH, KL and KV.
^LB	Define an open box.
^LD	Define a dashed (dotted) line.
^LF	Define an open box with multiple vertical lines (form).
^LS	Define a solid line.
^M	Define the current upright (normal) expandable characters.
^Q	Define a graphic image.
^R	Toggle character reverse image on or off (depending on the previous state).
^S	Define the current non-expandable font.
^T	Change the horizontal cursor position.
^U	Define the current upside-down expandable characters.
^V	Define the current vertical (top-to-bottom) expandable characters.
^W	Change the current expandable character width.
^X	Absorb all incoming data until a '^A' command is encountered.
^Y	Define auto-increment/decrement values.
^Z	Select a resident logo. ^Z1 = {reg} and ^Z2 = {copyright}
^*	Image mode terminator. Returns to filter mode.
^-	Image mode terminator. Returns to filter mode.
^,	Image mode terminator. Returns to filter mode.
^[Define a variable data field used by the buffered overlay function '^B'.

^A command

This command will cause all subsequent characters to not be ignored, thereby counteracting any ^X command that was issued previously. When the ^A command is encountered, the Imager will stop absorbing or "eating" all incoming data and will begin examining the data again for other image mode commands. The command format is:

^A

^B command

This command is used for printing horizontal bar codes. The height parameter of the preceding image command (^M, ^V, ^E, ^U, or ^H) will determine the bar code height. This command has two possible formats. One format allows a user to specify a bar code with a default "ratio" while the other format allows the user to override the default "ratio". The command formats are:

^Batd...d^G

or

^Ba9tr...rd...d^G

where: *a* = Human readable autoprint indicator,
 valid entries are: 'Y'es, 'N'o or 'O'CR.
 9 = '9' indicating a user defined ratio follows.
 t = Bar code type, see the Standard Bar Code Table.
 r...r = User defined ratio.
 d...d = Data to encode as a bar code.
 ^G = Bar code sequence terminator.

^C command

This command is used for printing vertical bar codes. The width parameter of the preceding image command (^M, ^V, ^E, ^U, or ^W) will determine the bar code "height". The OCR font is not available for printing human readable text with vertical bar codes. This command has two possible formats. One format allows a user to specify a bar code with a default "ratio" while the other format allows the user to override the default "ratio". The command formats are:

^Catd...d^G
or
^Ca9tr...rd...d^G

where: *a* = *Human readable autoprint indicator,*
valid entries are: 'Y'es or 'N'o.
9 = '9' indicating a user defined ratio follows.
t = *Bar code type, see the Standard Bar Code Table.*
r...r = *User defined ratio.*
d...d = *Data to encode as a bar code.*
^G = Bar code sequence terminator.

^D command

Some lower case characters (g, j, p, q, y) have descenders which are designed to print below the font base line. The Imager can print these characters in two different ways: aligned at the font base line, or descending the appropriate distance below the base line.

The **^D** command is used to toggle from "descenders off" to "descenders on" and vice-versa. The command format is:

^D

When image mode is activated, the pass always begins with "descenders off". The first ^D will set "descenders on". A second ^D will set "descenders off", as will a pass terminator. The state of descenders may be toggled as many times as necessary within the pass.

Setting "descenders on" within an image pass will produce an under-character gap that would not otherwise be present. This is true even if no lower case characters with descenders are printed.

This command is valid for image mode graphics fonts which include the expandable characters. It does not include the "high-resolution fonts" which are activated with the image mode **^S** command. The high-resolution fonts always print the descender characters below the font base line.

^E command

This command is used to print large characters in a "sideways - bottom-to-top" orientation. It has the same format in image mode as it does in filter mode. The universal height, width, and vertical positioning (justification) are specified, followed by the characters to be printed. The command format is:

^Ehhwwjjdc...c

- where: *hh* = Height of characters in 1/10 inches; two digits from 00 to 99.
 Represents the physical (vertical) height. Since the characters are sideways, it really specifies what would classically be thought of as character width. Includes inter-character gap.
- ww* = Width of characters in 1/10 inches; two digits from 00 to 99.
 Represents the physical (horizontal) width. Since the characters are sideways, it really specifies what would classically be thought of as character height.
- jjd* = Justification from the current position. Indicates how far down from the top of the pass the characters will begin printing.
- jj* = 1/10 inches; two digits from 00 to 99.
- d* = Dots (1/72 inches); one digit from 0 to 9.
- c...c* = Character or characters to be printed.

The parameters of this command are optional, but will be processed if they exist. If while processing the parameters, an invalid parameter such as a non-digit character is encountered, that parameter plus any remaining unprocessed parameters are assumed to be zero.

This command is also used to print small characters in a "sideways - bottom-to-top" orientation. If the height and width values are greater than "01", then the active font will be the expandable font with the height and width assigned by this command. However, if the values are either "00" or "01", then non-expandable fonts are selected. The fonts are:

Regular Image Mode Fonts

<i>ww</i>	<i>hh</i>	Selected font
00	00	7.5 cpi
01	01	10 cpi
00	01	12 cpi
01	00	15 cpi

^G command

This command is the general purpose "terminating" command for certain other filter mode and image mode commands. It has no parameters and has no meaning unless another command requires its use. This terminator is normally used with commands that have a variable (non-fixed) number of parameters. The command format is:

^G

^H command

This command may be used to change the universal height value within a sequence of image commands. When this command is encountered, the universal height would have already been set with a **^M**, **^V**, **^E** or **^U** command. This command is available to change that value for expandable text or a bar code. The command format is:

^Hnn

where: *nn* = *New universal height in 1/10 inches; two digits from 00 to 99.*

^I command

This is the "interrupt" command. It is used to "interrupt" the current image mode processing sequence and start another one. It allows a second image mode sequence to begin printing while the previous image mode sequence is still printing. Originally, this command was employed because printer graphic controllers lacked the memory to process larger image mode sequences. This command allowed the user to "get around" the memory limitation. Today, this command is not necessary and its use is not recommended. The effect this command has on the image mode sequence following the **^I** command is to alter its vertical cursor position by the sum of the interrupt value plus the current image mode sequence's origin. The interrupt process is terminated when the **^I000** sequence is encountered at the end of an image mode sequence.

The command format is:

^Ittd

where: *tt* = *1/10 inches to change the vertical position.*
d = *Dots (1/72 inches) to change the vertical position.*

^J command

This command may be used to change the justification (vertical positioning) value within a sequence of image commands. It moves the vertical position for the next image down the distance specified from the top of the pass. When this command is encountered, the justification may have already been set with a **^M**, **^V**, **^E** or **^U** command. This command is available to change that value. The command format is:

^Jjd

where: *jjd* = Justification from the current position. Indicates how far down from the top of the pass the next image will begin printing.
jj = 1/10 inches; two digits from 00 to 99.
d = Dots (1/72 inches); one digit from 0 to 9.

^KF command

This command is used to activate and deactivate (toggle) horizontal high-resolution printing. The command format is:

^KF

^KH command

This command is used to toggle half-tone shading on and off. The command format is:

^KH

The actual shading pattern may be defined with the **^KL** command.

^KL command

This command is used to define half-tone shading patterns. It defines a 6 bit pattern which is replicated horizontally across an image and which is alternated with the "reverse image" of the same pattern vertically down the image. The command format is:

^KLpp

where: *pp* = Pattern; two hexadecimal digits from C0 to FF, or 00, 01, or 04.

There are 64 user definable patterns, from 'C0' through 'FF'. There are three "built-in" shading patterns, designated by 'pp' values of '00', '01', and '04'.

<i>pp</i>	Shading Pattern
00	vertical stripes
01	diagonal stripes, lower left to upper right
04	diagonal stripes, upper left to lower right
C0 to FF	user definable patterns

To utilize a shading pattern that has been defined, the ^KH command must be used to turn half-tone shading on.

^KV command

This command is used to activate and deactivate (toggle) vertical high-resolution printing. The command format is:

^KV

^LB command

This command is used to define a rectangular box. The command requires the height, width, and box "side" thicknesses be entered. The command format is:

^LBhhhdvvdhv

where: *hhhd* = Horizontal dimension in 1/10 inches & dots (1/60 inches).
vvvd = Vertical dimension in 1/10 inches & dots (1/72 inches).
h = Horizontal sides thickness in dots (1/72 inches).
v = Vertical sides thickness in dots (1/60 inches).

^LD command

This command is used to define a dashed line. The command requires the height and width to be entered. The direction of the line is determined by which dimension (horizontal or vertical) has the greater value. The command format is:

^LDhhhdvvd

where: *hhhd* = Horizontal dimension in 1/10 inches & dots (1/60 inches).
vvvd = Vertical dimension in 1/10 inches & dots (1/72 inches).

^LS command

This command is used to define a solid line. The command requires the height and width to be entered. The direction of the line is determined by which dimension (horizontal or vertical) has the greater value. The command format is:

^LShhdvvvd

where: *hhhd* = Horizontal dimension in 1/10 inches & dots (1/60 inches).
vvvd = Vertical dimension in 1/10 inches & dots (1/72 inches).

^LF command

The **^LF** command is used to define a "form", by printing vertical lines within a box. The vertical lines are drawn from the top edge to the bottom edge of the box. The command requires the height, width, and "side" thicknesses of the box to be entered, as well as the position and thickness of each vertical line within the box. The command format is:

^LFhhdvvvdvhppppdt...ppppdt^G

where: *hhhd* = Horizontal box dimension in 1/10 inches & dots (1/60 inches).
vvvd = Vertical box dimension in 1/10 inches & dots (1/72 inches).
h = Horizontal sides thickness in dots (1/72 inches).
v = Vertical sides thickness in dots (1/60 inches).
pppd = Vertical line placement in 1/10 inches & dots (1/60 inches).
t = Vertical line thickness in dots (1/60 inches).

The command has a variable number of parameters, with each **pppd** representing another vertical line to be drawn in the box. Each vertical line is positioned relative to the previous vertical line; not relative to the left side of the box. Since an arbitrary number of parameters may be specified, a **^G** is used to terminate the command.

^M command

This command is used to print large characters in a "normal - left-to-right" orientation. It has the same format in image mode as it does in filter mode. The universal height, width, and vertical positioning (justification) are specified, followed by the characters to be printed. The command format is:

^Mhhwwjjdc...c

where: *hh* = Height of characters in 1/10 inches; two digits from 00 to 99.
ww = Width of characters in 1/10 inches; two digits from 00 to 99.
Includes inter-character gap.
jjd = Justification from the current position. Indicates how far down from the top of the pass the characters will begin printing.
jj = 1/10 inches; two digits from 00 to 99.
d = Dots (1/72 inches); one digit from 0 to 9.
c...c = Character or characters to be printed.

The parameters of this command are optional, but will be processed if they exist. If while processing the parameters, an invalid parameter such as a non-digit character is encountered, that parameter plus any remaining unprocessed parameters are assumed to be zero.

This command is also used to print small characters in a "normal - left-to-right" orientation. If the height and width values are greater than "01", then the active font will be the expandable font with the height and width assigned by this command. However, if the values are either "00" or "01", then non-expandable fonts are selected. The fonts are:

Regular Image Mode Fonts

<i>ww</i>	<i>hh</i>	Selected font
00	00	7.5 cpi
01	01	10 cpi
00	01	12 cpi
01	00	15 cpi

^Q command

This command is used to print logos and other "bitmapped" graphics images. These graphics images are represented in a 7 bit vertical graphics format.

Graphics images are printed horizontally across a label in "bands". Each band is 7 dots high by as many dots wide as necessary to print across a complete graphics image. If a graphics image is taller than 7 dots, then printing multiple bands will be necessary to print the entire graphics image.

Each column of 7 dots in a band is represented by pairs of characters. These characters are comprised of the characters '0' through '9' and 'A' through 'F' (16 possible characters). This representation is referred to as **pseudo-hexadecimal** or "four-bit hex". That is, the characters give the appearance of being hexadecimal characters but they are actually 8 bit ASCII characters.

The character pair which would activate all 7 dots (print black) in a column would be the pair '7F'. The character pair which would deactivate all 7 dots (print white) would be the pair '00'. The value required to activate the dots in a column is determined by assigning a different value to each dot in a column. The values assigned to each dot, from the bottom of a column to the top, is **1, 2, 4, 8, 16, 32, and 64**. The bottom dot would be activated with a value of decimal 1. The top dot would be activated with a value of decimal 64. To turn on any (or all) of the other dots, simply add their values together. The value required to turn on all 7 dots would be 1+2+4+8+16+32+64 or 127. Converting decimal 127 to hexadecimal (or pseudo-hexadecimal) would yield '7F' as the pair of characters to use.

The command format is:

^Qxx,....,xx^G

Where: *xx* = 7 bit pseudo-hex pair.
 ^G = graphics terminator.

NOTE: Commas may be used between the character pairs in order to make reading easier but they are not required.

^R command

This command is used to toggle reverse image on and off. The command format is:

^R

When image mode is activated, the pass always begins with reverse image off. The first ^R will turn reverse image on. The second ^R will turn reverse image off, as would the pass terminator. Reverse image may be toggled as many times as necessary within the pass.

Only characters may be printed in reverse image. Printing in reverse image will increase the size of the character cell, since additional space is required to create the black border around the character. Selecting a new font will turn reverse image off.

^S command

This command is used change the current font selection to one of the high-resolution fonts. The command format is:

^Sn

where: *n* = Font number; one digit from 1 to 6.

Font#	Font Description
1	10 cpi (characters per inch)
2	12 cpi
3	13.33 cpi
4	15 cpi
5	17.14 cpi
6	10 cpi OCR-A

^T command

This command may be used to tab to a new horizontal print position within a sequence of image commands. It is an absolute tab that sets the horizontal position for the next image the distance specified from the left edge of the printable area. The command format is:

^Thhd

where: *hhh* = Horizontal position in 1/10 inches; three digits from 000 to 136.
d = Horizontal position in dots (1/60 inches); one digit from 0 to 9.

^U command

This command is used to print large characters in an "upside down - right-to-left" orientation. It has the same format in image mode as it does in filter mode. The universal height, width, and vertical positioning (justification) are specified, followed by the characters to be printed. The command format is:

^Uhhwwjjdc...c

where: *hh* = Height of characters in 1/10 inches; two digits from 00 to 99.
ww = Width of characters in 1/10 inches; two digits from 00 to 99.
Includes inter-character gap.
jjd = Justification from the current position. Indicates how far down from the top of the pass the characters will begin printing.
jj = 1/10 inches; two digits from 00 to 99.
d = Dots (1/72 inches); one digit from 0 to 9.
c...c = Character or characters to be printed.

The parameters of this command are optional, but will be processed if they exist. If while processing the parameters, an invalid parameter such as a non-digit character is encountered, that parameter plus any remaining unprocessed parameters are assumed to be zero.

This command is also used to print small characters in a "upside down - right-to-left" orientation. If the height and width values are greater than "01", then the active font will be the expandable font with the height and width assigned by this command. However, if the values are either "00" or "01", then non-expandable fonts are selected. The fonts are:

Regular Image Mode Fonts

<i>ww</i>	<i>hh</i>	Selected font
00	00	7.5 cpi
01	01	10 cpi
00	01	12 cpi
01	00	15 cpi

^V command

This command is used to print large characters in a "sideways - top-to-bottom" orientation. It has the same format in image mode as it does in filter mode. The universal height, width, and vertical positioning (justification) are specified, followed by the characters to be printed. The command format is:

^Vhhwwjjdc...c

- where: *hh* = Height of characters in 1/10 inches; two digits from 00 to 99.
Represents the physical (vertical) height. Since the characters are sideways, it really specifies what would classically be thought of as character width. Includes inter-character gap.
- ww* = Width of characters in 1/10 inches; two digits from 00 to 99.
Represents the physical (horizontal) width. Since the characters are sideways, it really specifies what would classically be thought of as character height.
- jjd* = Justification from the current position. Indicates how far down from the top of the pass the characters will begin printing.
- jj* = 1/10 inches; two digits from 00 to 99.
- d* = Dots (1/72 inches); one digit from 0 to 9.
- c...c* = Character or characters to be printed.

The parameters of this command are optional, but will be processed if they exist. If while processing the parameters, an invalid parameter such as a non-digit character is encountered, that parameter plus any remaining unprocessed parameters are assumed to be zero.

This command is also used to print small characters in a "sideways - top-to-bottom" orientation. If the height and width values are greater than "01", then the active font will be the expandable font with the height and width assigned by this command. However, if the values are either "00" or "01", then non-expandable fonts are selected. The fonts are:

Regular Image Mode Fonts

<i>ww</i>	<i>hh</i>	Selected font
00	00	7.5 cpi
01	01	10 cpi
00	01	12 cpi
01	00	15 cpi

^W command

This command may be used to change the universal width value within a sequence of image commands. When this command is encountered, the universal width would have already been set with a **^M**, **^V**, **^E** or **^U** command. This command is available to change that value for expandable text or a bar code. The command format is:

^Wnn

where: *nn* = *New universal width in 1/10 inches; two digits from 00 to 99.*

^X command

This command will cause all subsequent characters to be ignored, until a **^A** command is received. If a **^A** command is never received, then none of the incoming data will be processed or printed. When ignore character is active, all data will be absorbed or "eaten". Except for the **^A** command, no commands are recognized while ignoring characters. The command format is:

^X

^Y command

This command is used to automatically increment or decrement numeric fields. These numeric fields may be text, bar code data or other command parameters. Use of this command is most powerful when used inside of a repeating sequence which is controlled by the **^R** ("repeat") command or the **^B** ("buffered overlay") command. The command format is:

^Ynnnsiii^G

where: *nnn* = *Numeric data to be modified.*
s = *Sign; + = increment, - = decrement.*
iii = *Increment/decrement value.*
^G = *Command terminator.*

^Z command

This command is used to select resident logos. These include the registered trademark and copyright symbols.

The command format is:

^Zn

where: *n* = Resident logo number.

Valid resident logo numbers are:

Logo #	Logo Description
0	unassigned
1	registered trademark
2	copyright symbol
3 - 9	unassigned

Decimal to Hexadecimal to ASCII Conversion

<u>Dec</u>	<u>Hex</u>	<u>ASCII</u>	<u>Dec</u>	<u>Hex</u>	<u>ASCII</u>	<u>Dec</u>	<u>Hex</u>	<u>ASCII</u>
0	00	NUL	43	2B	+	86	56	V
1	01	SOH	44	2C	,	87	57	W
2	02	STX	45	2D	-	88	58	X
3	03	ETX	46	2E	.	89	59	Y
4	04	EOT	47	2F	/	90	5A	Z
5	05	ENQ	48	30	0	91	5B	[
6	06	ACK	49	31	1	92	5C	\
7	07	BEL	50	32	2	93	5D]
8	08	BS	51	33	3	94	5E	^
9	09	HT	52	34	4	95	5F	_
10	0A	LF	53	35	5	96	60	`
11	0B	VT	54	36	6	97	61	a
12	0C	FF	55	37	7	98	62	b
13	0D	CR	56	38	8	99	63	c
14	0E	SO	57	39	9	100	64	d
15	0F	SI	58	3A	:	101	65	e
16	10	DLE	59	3B	;	102	66	f
17	11	DC1	60	3C	<	103	67	g
18	12	DC2	61	3D	=	104	68	h
19	13	DC3	62	3E	>	105	69	i
20	14	DC4	63	3F	?	106	6A	j
21	15	NAK	64	40	@	107	6B	k
22	16	SYN	65	41	A	108	6C	l
23	17	ETB	66	42	B	109	6D	m
24	18	CAN	67	43	C	110	6E	n
25	19	EM	68	44	D	111	6F	o
26	1A	SUB	69	45	E	112	70	p
27	1B	ESC	70	46	F	113	71	q
28	1C	FS	71	47	G	114	72	r
29	1D	GS	72	48	H	115	73	s
30	1E	RS	73	49	I	116	74	t
31	1F	US	74	4A	J	117	75	u
32	20	SP	75	4B	K	118	76	v
33	21	!	76	4C	L	119	77	w
34	22	"	77	4D	M	120	78	x
35	23	#	78	4E	N	121	79	y
36	24	\$	79	4F	O	122	7A	z
37	25	%	80	50	P	123	7B	{
38	26	&	81	51	Q	124	7C	
39	27	'	82	52	R	125	7D	}
40	28	(83	53	S	126	7E	~

41 29)
42 2A *

84 54 T
85 55 U

127 7F DEL

Imager Specifications

- Emulates QMS® Magnum® Code V Version 1.

Non-Scalable Characters

- Draft quality characters at 10 cpi, 12 cpi, and 15 cpi.
- High-resolution characters at 10 cpi, 12 cpi, 13.33 cpi, 15 cpi, and 17.14 cpi.
- 0.2 inch high characters at 7.5 cpi.
- OCR-A characters at 10 cpi.

Scalable Characters

- Variable size characters from 0.1 to 9.9 inches high and wide (ImagerPlus Only).

Rotated Print

- Normal - left-to-right.
- Sideways - top-to-bottom (ImagerPlus Only).
- Sideways - bottom-to-top (ImagerPlus Only).
- Upside down - right-to-left (ImagerPlus Only).

Forms and Labels

- Horizontal and vertical lines (ImagerPlus Only).
- Boxes and forms (ImagerPlus Only).
- Horizontal and vertical duplication of labels.
- Auto increment/decrement of numeric fields.

Bar Codes

- Horizontal bar codes.
 - Vertical bar codes (ImagerPlus Only).
 - Standard and variable ratio bar codes.
 - Automatically printed human readable text.
 - Automatically calculated check digits.
-
- Code 39, Codabar, MSI, Interleaved 2 of 5, UPCA 11 digit, UPCE 10 digit, UPCE0 6 digit, UPCE1 6 digit, EAN 13, EAN 8, Code 128 A/B/C, UCC-128, PostNet.

Graphics

- 60 x 72 dpi bit addressable graphics (ImagerPlus Only).

Reverse Image and Shading

- Reverse image characters.
- Variable half-tone shading.
- Horizontal and vertical half-dot high resolution modes.