Apollo
and A-Series

Programming Guide

August 2002

Tharo Systems, Inc
2866 Nationwide Parkway
P.O. Box 798
Brunswick, Ohio 44212-0798

Phone: (330)273-4408
Fax: (330)225-0099
Email: techsupport@tharo.com
http://www.tharo.com

cab-Produkttechnik GmbH
Wilhelm-Schicklard-Str. 14
76131 Karlsruhe
Deutschland

Telefon: 721 / 66 26-0
Telefax: 721 / 66 26-259
Email: info@cabgmbh.com
http://www.cabgmbh.com

Copyright and Trademark Notices

==========

This page intentionally left blank

Table of Contents

Table of Contents

==========

Table of Contents

This page intentionally left blank

Command / Data Types

There are three basic types of commands used on the Apollo printer, and one special type of data. Each of the following are described in separate sections of this manual:

- Immediate Commands are a single lower case letter. They perform a variety of printer functions. When the printer receives an Immediate command, it will immediately perform the command function, regardless of any other operation taking place. The effects of this command remain in place until the printer is reset.

- ESC Sequence Commands are a specialized group of commands frequently used for program control of network attached printers, when hands-on access to the control panel is not available.

- Label Format Commands are specified as a single upper case letter. They define the label and information to be printed on it, and are in effect for only one label job.

- Special Content Fields are used within Label Format commands. They consist of specific predefined words coded in brackets, [ ], that provide for various data insertion and data manipulation functions.

Command Syntax

- No special characters are needed to create a label format. Any text editor may be used to enter commands.
- There is no strict format within a command.
- Where<CR> is shown, it may be a CR, a LF, or a CR/LF.
- For readability, parameters may be aligned with SPACES, TABS or additional ZEROES in numeric parameters.
- Parameters are separated with a comma or semicolon.
- Comment lines can be included by coding a semicolon (;) in the first position.
- Optional parameters are shown in command definitions within brackets [ ]. Special content fields are an exception - their optional fields are shown in { }.

Command List

Command List

Command List

Command List

Command List

Command List

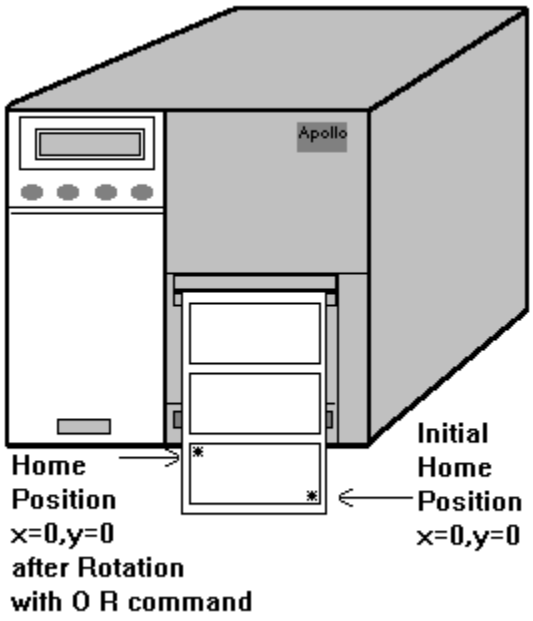
Print Orientation / Home Position for Commands

It is important to understand print orientation and home position before attempting to use label format commands. Orientation refers to where the top of the label is found for layout purposes. Home Position refers to the top-left corner of the label. The figure below shows the Apollo printer and the Home positions that are possible.

As shown below, the initial Home position is on the leading right edge of the label, the first edge that exits the printer. As viewed from the front of the printer, this would appear to make the label orientation upside-down. This is the default position.

As an alternative, the orientation can be “rotated” using the "O", Print Options command, with the "R", Rotate parameter. After executing this command, the new Home position is on the trailing edge of the label, the last edge that exits the printer. With this orientation, as viewed from the front of the printer, the label would appear to be right side up. When the O command with R parameter is used, it remains in effect only for the current label command set.

For easier understanding throughout this manual, many examples use the O command with R parameter to present them with a top to bottom approach, as if viewed from in front of the printer. The important thing to remember is that all parameters specified are in reference to the top-left, or Home position of the label. Whether that Home position is on the leading or trailing edge of the label, does not matter.



; - Comment Line

The ; identifies a comment line. A comment may be placed anywhere within the command set, on a line by itself. The printer ignores comment lines.

Command Format / Usage:

; comments <CR> Marks this line as a comment

a - Start ASCII Dump

The **a** command places the printer in ASCII dump mode. After the command is sent to the printer, the printer's LCD panel will display "ASCII Dump Mode." Any commands or label formats sent to the printer after this point are printed as the printer receives them, without interpretation. Pressing the on-line (ONL) button on the printer's front panel resets the printer to its normal mode of operation. This mode can also be entered by holding down the form feed key while powering on the printer.

Command Format / Usage:

a <CR> Set ASCII dump mode on.

c - Direct Cut

The **c** command initiates an immediate cycle of the printer's cutter. If necessary, the printer will perform a formfeed to find a label edge, prior to making the cut.

Command Format / Usage:

c <CR> Cuts label immediately.

d - Download Data

The **d** command accomplishes downloading of data files to the Apollo. Most often, it may be used to download graphics or additional fonts. The option [SAVE] will download and simultaneously create a copy on the PCMCIA memory card. There are two methods shown below for including the data with the command. Method 1 is the most reliable, but may require editing of the data before download. Method 2 will use whatever data is presented, but may occasionally misinterpret imbedded ESC-sequences in the data as a command. If at all possible, use of method 1 is recommended.

Command Formats:

d type; name[SAVE] [B:±value] <CR> $^{E}S_{C}$. data **$^{E}S_{C}$.** ***(method 1)***

d type; name[SAVE] [B:±value] <CR> $^{E}S_{C}$: data **$^{E}S_{C}$"end-of-data"** ***(method 2)***

Where:

d = Download data command.

type = The type of data that will follow, using standard file name extensions:
- GIF - Graphic Interchange Format
- PCX - Paintbrush format
- TIF - TIFF Format© Aldus Corp
- IMG - GEM Image format
- BMP - Windows bitmap format
- SPD - Speedo™ font format
- TTF - TrueType font format
- MAC - MacPaint format
- DBF - dBASE Database
- TMP - ASCII Serial file

name = The name to be associated with the downloaded data for later referencing by other commands.

[SAVE] The option for downloading to the PCMCIA memory card. (See Memory Card Section for further instructions and examples)

B:±value = Controls the brightness of dithering on color graphics. Valid up to ±20. i.e. B:-10 makes the picture 10 steps darker.

d - Download Data

ESC. data **ESC.** = Method 1 for delineating data. Data is in binary format, enclosed with ESC. (escape dot, which is an ASCII character 27, followed by ASCII character 46) at the start and end. For this method to work, any single ESC characters found in the data must be replaced by double escapes, ESCESC, prior to download. Other ESC commands for the Apollo will continue to work during this transmission. (See Appendix B for example of graphic download program with filter statements to replace ESC with ESCESC automatically).

Example: d PCX;logoname <CR> ESC**.** binary data ESC**.**

ESC: data **ESC"end-of-data"** = Method 2 for delineating data. Data is in binary format, preceded by ESC: (escape colon, which is ASCII character 27, followed by ASCII character 58) and followed by ESC"end-of-data"(escape "end-of-data", which is ASCII character 27, followed by ASCII text string "end-of-data"). There may be ESC sequences present in the data, but there must not be any ESC"end-of-data" imbedded in the data stream, as this marks the end of the data stream. Note: This method will not work on a RS485 network.

Example: d PCX;logoname<CR> ESC**:** data ESC"end-of-data"

e - Erase Data

The **e** command will erase various types of data, for example fonts and graphics, from the printer's memory. The **e** command does not erase data from the PCMCIA memory card. See the M - Memory Card Access to delete files from the memory card.

Command Format:

e type; name <CR>

Where:

e = Erase data command.

type = The type of data being removed, equivalent to standard file name extensions: GIF, PCX, TIF, IMG, BMP, SPD, TTF, MAC, FNT.

name= The name attached to the font or graphic when it was sent to the printer. A "*****" may be used as the name to delete all files of the same type.

Example:

e PCX;***** <CR>	Erases all PCX graphics currently in the printer's memory

f - Formfeed

The **f** command feeds one label until the top-of-form is under the printhead. This command has the same effect as pressing the "FF" form-feed button on the front panel.

Command Format / Usage:

f <CR> Causes an immediate formfeed.

l - Change Language/Country

The **l** command changes the language and country settings. Country affects the date formats and currency. Using the l command does not affect the language displayed on the LCD panel of the printer. To change the language of the printer's front panel prompts, you must select the 'Country' option in the printer's setup.

Command Format:

l name <CR>

Where:

l = Change language/country command.

name = The DOS short keyboard code for the country from the following:
BE - Belgium, French
CZ - Czech Republic
DK - Denmark
FR - France
GR - Germany
UK - Great Britain
IT - Italy
SP - Spain
SU - Suomi (Finland)
SF - Switzerland, French
SG - Switzerland, German
US - United States

Example:

l SP <CR> Changes printer language/country to Spain. If the date is Tuesday February 11, 1997 and the Language is set to Spain, then Martes Febrero 11, 1997 will print on the label. See the Special Content Fields Section for more on printing dates.

m - Set Measuring Unit

The **m** command specifies the unit of measure in effect for all following label format commands. The Apollo's default unit of measure depends on the Country assignment from the front panel setup. For country US, default measuring unit is inches. For all other countries, default measuring unit is millimeters. This command does not effect the printers default measuring unit. The measuring unit is only changed for the individual format being printed.

Command Format:

m unit<CR>

Where:

m = Set measuring unit command.

unit = The measuring system desired, **m** for metrics (millimeters) or **i** for inches (inches, tenths and hundredths of an inch).

Example:

m i <CR> Sets measuring unit to inches.

p - Pause Printer

The **p** command places the printer in a pause status, or removes it from pause status.

Command Format:

p status <CR>

Where:

p = Pause printer command.

status = The pause status, with **0** for pause off and **1** for pause on.

Example:

p 1 <CR> Sets printer pause on.

q - Query Printer

The **q** command provides a way to query the printer and obtain various types of information. The command has different formats depending on the information desired.

Command Formats / Usage:

q d; name<CR> — **d**atabase inquiry. Asks the printer if the database (DBF) file called "name" is available on the PCMCIA memory card. The printer's response will be Y or N.

q e; name<CR> — m**e**dia inquiry. Asks the printer if the media (FMT) file called "name" is available. The printer's response will be Y or N.

q f <CR> — **f**ree memory. Reports the free (available) memory, which may be used for downloaded data, to the serial port. Response will look like “0512345 bytes free”.

q i; name<CR> — **i**mage inquiry. Asks the printer if the image (IMG) file called "name" is available. The printer’s response will be Y if in memory, C if on the memory card, or N if not available.

q l; name<CR> — **l**abel inquiry. Asks the printer if the label (LBL) file called "name" is available. The printer's response will be Y or N.

q m <CR> — **m**emory card display. Displays the type of memory card currently inserted. Response examples: "No Card", or "SRAM, 512kByte", etc.

q p <CR> — **p**eripherals inquiry. Reports the type of peripheral devices that are connected. Possible responses are:

NONE, CUTTER, DEMAND SENSOR, BLOW ON, TRIGGER

q s;name <CR> — **s**caleable font inquiry. Asks the printer if the scaleable font (FNT) called "name" is available. The response will be Y if in memory, C if on the memory card, or N if not available.

r - Reset Printer

The **r** command resets all settings on the printer to the original default values and de-fragments the printer's memory. This command does not change the printer's configuration settings.

Command Format / Usage:

r <CR> Resets printer setting to default values.

s - Set Date/Time

The **s** command sets the printer's date and time to the value specified.

Command Format:

s datetime <CR>

Where:

s = Set date/time command.

datetime = A string value representing the date and time in the format of yymmddhhmmss.

Example:

s 981015082000 <CR> Sets printer date and time to:
October 15, 1998 8:20 a.m.

t - Run Printer Self-Test

The **t** command starts a self-test print. Apollo's self-test print includes: patterns for head check, state of the printer, list of fonts. This command has the same effect as holding the "ONL" (on-line) button down during printer power-up.

Command Format / Usage:

t <CR> Initiates printer self-test.

v - Firmware Version

The **v** command will inquire about the current firmware revision level and printer model. The response will include the level, date and model.

Command Format / Usage:

v <CR> Firmware level inquiry; will produce response such as: 2.37 Dec 20 1996 (Apollo 1)

x - Set Peripheral Signal Bits

The **x** commands set the signal bits for the peripheral connector on the front of the Apollo (output pins). This makes possible the control of a peripheral device, which is usually other than a standard cutter or applicator. The four bits available for use on the connector may be used individually or as a group. Pin/bit assignments and usage are:

Pin 3 = Control bit 0, set on at the start of printing a label
Pin 11 = Control bit 1, toggled when a new job is started
Pin 4 = Control bit 2 , set on for error
Pin 12 = Control bit 3, set on when label is in peel-off position

Each of the bits may be set or reset individually. These bit signals may be used to switch simple mechanical devices or enable bar code verifiers.

Note: These peripheral signal bits are not reset when an Immediate Command "r" is executed. To reset these bits, use ESC!ESC! (see ESC Sequence commands)

x m; mask <CR> — Set the bits to system or user defined mode. **mask** is a hex nibble. Where mask is bit-on, selects the system status value for the bit, and where mask is bit-off selects the user defined value for the bit.

Example: x m; E <CR> — Sets bit 0 to user defined mode. Leaves bits 1, 2, 3 at system status values.

x d; ao <CR> — Set/Reset mask bits directly. **a** is an AND mask and **o** is an OR mask, both being a hex nibble, written together as a hex byte.

Example: x d; E8 <CR> Clears bit 0; sets bit 3 on.

x e; ao <CR> — Set value for all bits when error is detected. **a** is AND mask and **o** is OR mask, both being a hex nibble, written together as a hex byte.

Example: x e; BB<CR>	Clears bit 2; sets bits 0, 1, 3 on.
x s; ao <CR>	Sets standby value, for no job in process. **a** is AND mask and **o** is OR mask, both being a hex nibble, written together as a hex byte.
Example: x s; 07<CR>	Clears bits 0,1,2,3; sets bits 0,1,2 on.

z - Slashed Zero Selection

The **z** command selects the style of zero to be printed in label text, whether it should appear with a slash (/), or not. This command can only effect internal bitmapped fonts, and will have no effect on internal outline or downloaded outline fonts.

Command Format:

z option <CR>

Where:

z = Slashed zero selection command.

option = Code the number 0 to specify slashed zero (Ø) printing; code the upper case letter O to specify unslashed zero (0) printing.

Example:

z 0 <CR> Selects zero printing as Ø
z O <CR> Selects zero printing as 0

ESC Sequence / Network Commands

The **ESC** sequences that follow affect the printer's status, and are immediately executed when received by the printer. When printers are networked, every network printer listens on the bus and waits for its' commands. These commands are most often used when a program (QBASIC, C, etc.) is issuing the commands to the printer. Some can also be used from a text file, but the text editor would need the capability of accepting ESC within the text. Otherwise, most of these functions are available through standard lower case Immediate commands for inclusion in a text file.

Note: See Appendix B for sample program on ESC command usage.

Command Formats / Usage:

ESC *	Activate all network printers.
ESC A - ESC Z	Activates an individual network printer using designations A through Z, and deselects all other printers not specified.
ESC ! ESC !	Performs a hard reset on the active printer(s*),* equivalent to turning the printer off then on.
ESC c	Cancels the current job or activity on the printer. This has the same effect as pressing the "Cancel" (CAN) key on the front of the printer.
ESC p0 **ESC p1**	Ends printer pause state. Sets printer to pause state.
ESC ?	Printer free-memory inquiry, will return a response of 0-9, where: 0= 0-9%, 1=10-19%, . . . 9 = 90-99%.

ESC Sequence / Network Commands

ESC s
XYNNNNNNZ

Printer status inquiry, will return a response as follows:

where:

X = On-line (Y=yes, N=no)

Y = Type of error:

- .....No error
a .....Applicator error – upper position
b .....Applicator error – lower position
c ..... Applicator error – vacuum plate empty
d ..... Applicator error – label not deposited
e ..... Applicator error – host stop/error
f ..... Applicator error – reflective sensor blocked
B ..... Protocol error
C ..... Memory card error
D .....Printhead open
E .....No label found
F .....End of ribbon
H ..... Illegal heating voltage
M ..... Cutter jammed
N .....Media too thick for cutter
O ..... Out of memory
P .....Out of paper
S .....Ribbon saver malfunction
V ..... Input buffer overflow
W .....Print head too hot
X ..... External I/O error
Z .....Printhead damaged.

NNNNNN = Number of labels left to print

Z = Job status (*Y/N*) *Y* means printer is working on a job. *N* means standby. The printer delivers Y immediately after starting a job and sets the flag back to N when the last label of that job is printed.

ESC t

Cancels the current job or activity and clears the input buffer of all data or jobs.

ESC ESC

Use as a replacement for single ESC in binary data while remaining transparent to networks.

A - Amount of Labels

The **A** command specifies the amount of labels to be printed. It is used to end a label definition. It may be sent as often as necessary to reprint the last label sent to the Apollo.

Usage:

A [NO] n | [?] | [?,R] | [$DBF] <CR>

Where:

A= Amount of labels command.

[NO] Used only when saving a label to the PCMCIA memory card. If the **[NO]** option is used the label will not print until the label is accessed from the memory card. If the **[NO]** option is not used, the label will print immediately when the label is sent to the printer. (See Appendix B for examples of using the A command with a memory card).

n or **[?]** or **[?,R] or [$DBF]**

n Number of labels to print. If omitted, an infinite number of labels will be printed.

[?] Prompts the user for the number of labels to be printed. This is the same as if the A command were omitted. It is intended for several jobs within the same program code, such as with multiple R commands separated by multiple A [?].

[?,R] Prompts the user for the number of labels to print. And when the job finishes printing, it prompts for the number to print again. Any incrementing number continues where it left off.

[$DBF] Prints the number of labels equal to the number of records in a database on the memory card. (See example on next page)

Example:

A 15 <CR> Prints 15 labels.

A [NO] <CR> Does NOT print the label when sent to the memory card.

A - Amount of Labels (Cont.)

This sample Apollo code illustrates the usefulness of the [$DBF] switch. The database "shipping" on the memory card has 450 records, sequentially numbered from 0001 to 0450. The text field *memdbfld* is assigned a sequentially incrementing number by [SER:0001]. The database field D1 loads the data from the specified record number defined by *memdbfld* and the A [$DBF] causes exactly 450 records to print.

```
m m
J
H 86,-10,T
O R
S l1;.0,.0,152.4,155.4,101.6
E dbf;shipping
T:memdbfld;37.8,99.0,.0,3,6.4;[SER:0001][I]
T:D1;17.1,11.3,.0,3,6.4;[DBF:RECNBR,memdbfld,RECNBR][I]
T:D2;16.8,20.4,.0,3,6.4;[DBF:RECNBR,memdbfld,NAME]
A [$DBF]
```

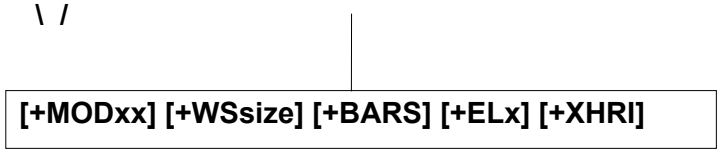
B - Bar Code Field Definition

The **B** command is used to place a bar code field in the current label format.

Currently, there are twenty-two (22) symbologies plus two "add-on" symbologies supported by Apollo. Depending on the symbology chosen, the parameters used within the command will vary. The bar code is normally printed in one of four (4) rotations with variable element widths and variable height. The corresponding text interpretation may be printed if desired.

Usage:

B [:name;] x, y, r, type, [+options,] size; data <CR>



Where:

B= Bar code field definition command.

[:name;] = Optional parameter. A unique name given to this bar code field. “name” must begin with a colon may be up to ten characters long, and may not contain any special characters. The field “name” may be used later to concatenate fields, in field data replacement or in field calculations and comparisons

***x*=** Coordinate "x" specifies the horizontal position of the bar code. This is the distance, in inches or millimeters, from the left edge of the printable area to the start position of the bar code. The printable area is defined by Label Size command “S". Start position of the bar code refers to the upper left corner of the bar code symbol. The unit of measurement, either inches or millimeters, is set by the Immediate Command “m".

y= Coordinate "y" specifies the vertical position of the bar code. This is the distance, in inches or millimeters, from the top of the label to the start position of the bar code. Start position of the bar code refers to the upper left corner of the bar code symbol. The unit of measurement, either inches or millimeters, is set by the Immediate command “m”.

r = Rotation of the bar code field. The rotation may be given in any one of 360 degrees, but for a scaleable bar code rotation should be given as 0, 90, 180 or 270 degrees. In determining rotation, consider the current orientation of the label, as specified by the Option command, Rotate parameter.

B - Bar Code Field Definition

type= The bar code symbology type selected. Symbology type is specified as either the full bar code name or a single letter short code as shown in the table below.

When using the full bar code name, the Apollo extracts only characters and digits from the named type, so UPC-A, UPC A and UPCA have the same result. Use of upper or lower case characters when naming the type of bar code controls printing of human readable interpretations, HRI, under the bar code. Type named in upper case causes HRI to be printed, and when named in lower case, causes no HRI printing. The Apollo HRI print size varies with the bar code height chosen.

Symbology types:

Bar Code Name	Short Code	Ratio Type *	Page
CODE 39	A	ratio	28
UPC-A	B	non-ratio	29
UPC-E	C	non-ratio	30
2 OF 5 INTERLEAVED	D	ratio	31
CODE 128	E	non-ratio	32
EAN-13	F	non-ratio	34
JAN-13	F	non-ratio	34
EAN-8	G	non-ratio	35
JAN-8	G	non-ratio	35
HIBC	H	ratio	36
CODABAR	I	ratio	37
MSI	K	ratio	38
ADD-ON 2	M	non-ratio	39
ADD-ON 5	N	non-ratio	40
CODE 93	O	non-ratio	41
POSTNET **	P **	non-ratio	42
UCC-128	Q	non-ratio	43
EAN-128	Q	non-ratio	43
FIM **	S **	non-ratio	45
MAXICODE **	U	non-ratio	46
DATAMATRIX **	W **	non-ratio	48
PLESSEY	X	ratio	50
UPC-E0	Y	non-ratio	51
PDF417 **	Z **	non-ratio	52
QRCODE ** #	N/A	non-ratio	54

* Ratio vs Non-Ratio bar codes will use different options for the size parameter.
** Upper case does not produce HRI on this symbology.
# Available on A-Series printers only

B - Bar Code Field Definition

size = Represents the bar height and width in a bar code. For Ratio bar codes, size is defined as **height, narrow element, ratio**. For Non-Ratio bar codes, size is defined as **height, narrow element** or as **SCx** for UPC/EAN bar codes. (See Symbology Type table with 'type' above for ratio/non-ratio designations)

height = Bar code height given in inches, or millimeters, limited only by label size.

narrow element= Width of narrow bar elements given in inches or millimeters. Apollo 1, Apollo 2, Apollo 3/300, Apollo 4/300 and A3/300 prints at 300 dots per inch, or 12 dots per millimeter, limiting actual narrow bar element size to multiples of .0033 inches, or .083 mm. Apollo 3, Apollo 4, A8 and A3/200 prints at 203 dots per inch, or 8 dots per millimeter, limiting actual narrow bar element size to multiples of .0049 inches, or .125 mm. Any size not a correct multiple will be rounded to the nearest multiple.

ratio = Relation of wide bar to narrow bar. Given as a ratio value (i.e. 5:2).

SCx = Prints a bar code based on UPC/EAN recommendations. Height of the bar code is 80% of the width. SC0 produces a bar code at 80% and SC1 provides a 100% bar code

data = String of ASCII characters to be encoded in the bar code symbol. Each bar code symbology has unique restrictions on the characters allowed, the length and the format of the string to be encoded. See individual symbology descriptions following for more specific information on the data allowed.

B - Bar Code Field Definition

+options = Options depend on the symbology being used. With the individual symbology descriptions on the following pages, appropriate options are shown with each. Options available include:

+MODxx= Allows addition of Check Digits to the symbology:
- **MOD10** Numeric data only,
- **MOD11** Numeric data only, typically used with MSI.
- **MOD43** Code 39 only.
- **MOD16** Codabar only.
- **MOD10GP** German Parcel Service, Interleaved 2 of 5 only

+WSsize = Prints quiet zone markers around the bar code to use as a design aid. A quiet zone is a clear area required around every bar code to ensure proper scanning. ***size*** is the length of the marker in inches or millimeters. These markers should be used only during design of a label, and should be removed before printing the final labels.

+**BARS=** Prints bearer bars above and below the symbology. Normally used with Interleaved 2 of 5 and Plessey bar codes.

+ELxx = Error Level is valid only with PDF417 and QRCODE. It defines the amount of redundancy included in the bar code.

+XHRI= Enhanced HRI is valid only with Code 39, Code 93, UPC-A and UPC-E. It provides additional human readable interpretation options.. For Code 39 it will print optional asterisks for start/stop. For Code 93 it will print optional boxes for start/stop. For UPC-A and UPC-E, it reduces the size of the product code and check digit symbols that are printed preceding and trailing the bar code.

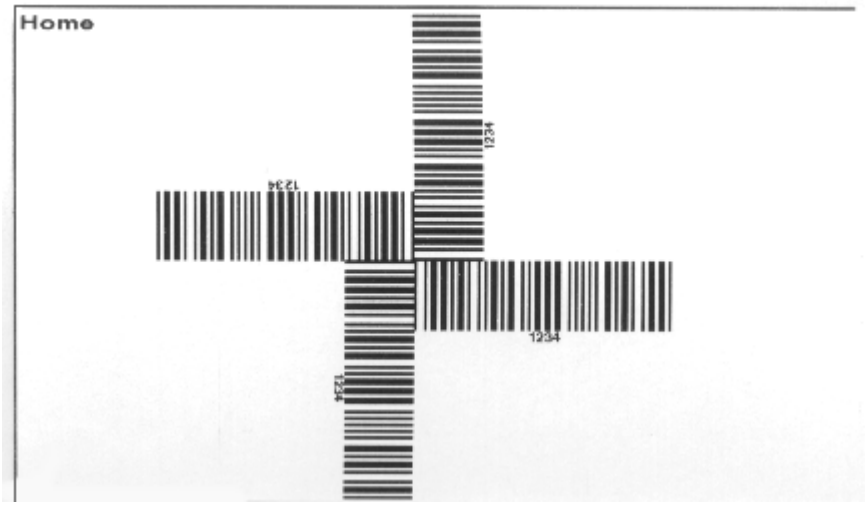
+NOCHECK = Disables the automatic check digit generation for variable weight barcodes (EAN 13 and UPC A only)

+RECT = DataMatrix can be printed as a square which is the default or as a rectangle which is the purpose of the +RECT command. (DataMatrix only)

B - Bar Code Field Definition

The following label shows the x and y coordinates in relation to the Home position when it is located in the top left corner of the label.

The bar codes on the following label are each positioned x=2.00 and y=1.30. The rotations shown are at 0, 90, 180 and 270.



B - Bar Code Field Definition

Bar Code Name: CODE 39 (Code 3 of 9)
Short Code: A

Command:

B [:name;] x, y, r, CODE39, [+WSize,] [+XHRI,] height,narrow,ratio; data <CR>

Description:

Code 3 of 9 is a variable-length alphanumeric code. Valid characters uppercase letters A-Z, digits 0-9 and the special characters - . $ / + % and space. Apollo inserts the required start/stop character automatically. If the +XHRI option is used, the start/stop characters will be printed as '*'. There is no check digit.

Examples:

```
m i
J
O R
S l1;.0,.0,2.5,2.5,4.26
B:CODE39;1.0,0.25,0,CODE39,0.4,.015,3:1;ABC123
B:CODE39;1.0,0.75,0,code39,0.4,.015,3:1;ABC123
B:CODE39;1.0,1.30,0,A+XHRI,0.4,.015,3:1;ABC123
A 1
```

B - Bar Code Field Definition

Bar Code Name: UPC-A
Short Code: B

Command:

B [:name;] x, y, r, UPCA, [+WSize,] [+XHRI] [+NOCHECK,] height,narrow; data <CR>
{ or SCx }

Description:

UPC-A is a fixed-length of 12, numeric-only bar code. The Apollo accepts the first eleven digits of data with the command, then automatically calculates and appends a Mod 10 check digit as the twelfth digit. If the +XHRI option is used, the print size of the product code and check digit, which precede and trail the bar code, will be reduced in size.

Examples:

```
m i
J
O R
S I1;.0,.0,2.5,2.5,4.26
B:UPCA1;0.75,0.25,0,UPCA,0.9,.013;01234554321
B:UPCA2;0.75,1.35,0,B+XHRI,SC1;01234554321
A 1
```

B - Bar Code Field Definition

Bar Code Name: UPC-E
Short Code: C

Command:

B [:name;] x, y, r, UPCE, [+WSize] [+BARS] [+XHRI] height,narrow; data <CR>
{ or SCx }

Description:

UPC-E is a fixed-length of eight, numeric-only bar code. The Apollo accepts seven digits with the command, where the first must be a zero, then automatically calculates and appends a check digit as the eighth. If the +XHRI option is used, the print size of the product code and check digit, which precede and trail the bar code, will be reduced in size.

Examples:

```
m i
J
O R
S I1;.0,.0,2.5,2.5,4.26
B:UPCE;1.0,0.22,0,UPCE,0.7,.013;0123456
B:UPCA;1.0,1.32,0,C+XHRI,SC1;0123456
A 1
```

B - Bar Code Field Definition

Bar Code Name: 2 OF 5 INTERLEAVED
Short Code: D

Command:

B [:name;] x, y, r, 2OF5INTERLEAVED, [+MODxx,] [+WSize,] [+BARS,] . . .
. . **height,narrow,ratio; data <CR>**

Description:
Interleaved 2 of 5 is a variable-length numeric-only bar code. This symbology encodes numbers in pairs, so if an odd number of numeric characters is sent, a leading zero will be added to the data encoded in the bar code. Adding the optional parameter +MODxx to the symbology name attaches a check digit to the end of the bar code. In many cases, Interleaved 2 of 5 should be printed with bearer bars at the top and bottom to prevent partial scans of the bar code. The +BARS option will produce the bearer bars when desired.

Examples:

```
m i
J
O R
S I1;.0,.0,2.5,2.5,4.26
B:I2of5;1.0,0.40,0,D,0.5,.015,3:1;1234567890
B:BAR2 ;1.0,1.00,0,2of5interleaved+BARS,0.5,.015,3:1;1234567890
B:BAR3 ;1.0,1.60,0,2 OF 5 INTERLEAVED+MOD10,0.5,.015,3:1;1234567890
A 1
```

B - Bar Code Field Definition

Bar Code Name: CODE 128
Short Code: E

Command:

B [:name;] x, y, r, CODE128, [+MODxx,] [+WSize,] [+BARS] . . .

. . **height,narrow; [U:codeset character] data <CR>**

Description:

Code 128 is a variable-length code that includes all of the 128 standard ASCII characters. The Apollo automatically calculates and appends the mandatory mod 103 check digit. If an additional check digit is desired, it may be added with the +MOD option.

Available Codesets:

The default Subset for Code 128 is Subset B. Depending on the data in the Code 128 bar code, the Apollo printer switches to the correct Subset automatically. If you want to force a particular Subset and prevent shifting to another Subset, use the [U:CODEx] option.

Code 128 Subset A includes all of the standard uppercase alphanumeric keyboard characters plus the control and special characters. To manually select Code 128 Subset A, place [U:CODEA] before the data to be encoded.

Code 128 Subset B includes all of the standard uppercase alphanumeric keyboard characters plus the lowercase alphabetic and special characters. To manually select Code 128 Subset B, place [U:CODEB] before the data to be encoded.

Code 128 Subset C includes the set of 100 digit pairs from 00 through 99 inclusive, as well as special characters. Code 128 Subset C is used for double density encoding of numeric data. To manually select Code 128 Subset C, place [U:CODEC] before the data to be encoded. Subset C can only encode numeric data.

B - Bar Code Field Definition

Bar Code Name: CODE 128
Short Code: E

Examples:

```
m i
J
O R
S I1;.0,.0,2.5,2.5,4.26
B:bar1;1.00,0.30,0,CODE128,.6,.015;[U:CODEA]ABC123
B:bar2;1.00,1.00,0,CODE128,.6,.015;[U:CODEB]ABCxyz123
B:bar2;1.00,1.70,0,CODE128+MOD10,.6,.015;[U:CODEC]123456
A 1
```

B - Bar Code Field Definition

Bar Code Name: EAN-13/JAN-13 (European/Japanese Article Numbering)
Short Code: F

Command:

B [:name;] x, y, r, EAN13, [+WSize,] [+BARS,] [+NOCHECK,] height,narrow; data <CR> { or SCx }

Description:

EAN/JAN 13 is a fixed-length of 13, numeric-only bar code. The Apollo accepts 12 data characters with the command, then automatically calculates and appends the check digit as the thirteenth. JAN-13 (Japanese Article Numbering system) is a special application of EAN-13 where the first two values entered must be “49”, designating Japan.

Examples:

```
m i
J
O R
S I1;.0,.0,2.5,2.5,4.26
B:EAN13;1.65,0.11,0,EAN13,SC1;402345607891
B:EX2;1.65,1.1,0,EAN13,.6,.013;402345607891
B:JAN13;1.65,1.8,0,JAN13,.6,.013;490005607891
A 1
```

B - Bar Code Field Definition

Bar Code Name: EAN-8, JAN-8 (European/Japanese Article Numbering)
Short Code: G

Command:

B [:name;] x, y, r, EAN8, [+WSize,] [+BARS,] height,narrow; data <CR>
{ or SCx }

Description:

EAN/JAN 8 is a fixed-length of 8, numeric-only bar code. The Apollo accepts 7 data characters with the command, then automatically calculates and appends the check digit as the eighth. JAN-8 (Japanese Article Numbering system) is a special application of EAN-8 where the first two values entered must be “49”, designating Japan.

Examples:

```
m i
J
O R
S l1;.0,.0,2.5,2.5,4.26
B:EAN8;1.66,0.11,0,EAN8,SC1;4023456
B:EX2;1.66,1.00,0,G,.6,.013;4023456
B:JAN8;1.66,1.70,0,EAN8,.6,.013;4900056
A 1
```

B - Bar Code Field Definition

Bar Code Name: HIBC (Health Industry Bar Code)
Short Code: H

Command:

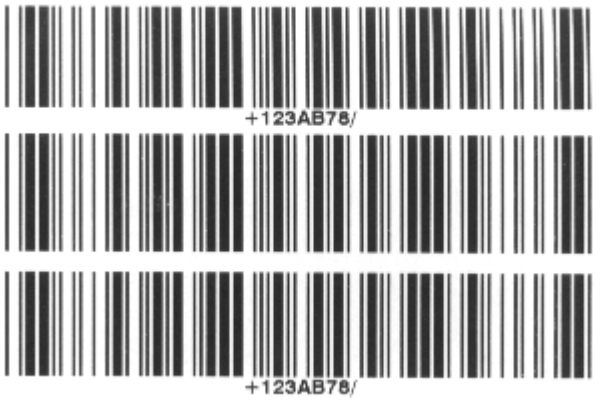
B [:name;] x, y, r, HIBC, [+WSize,] [+BARS,] height,narrow,ratio; data <CR>

Description:

The **Health Industry Bar Code** is a variable-length alphanumeric bar code. It is a variation of Code 3 of 9 with mod 43 check digit. The characters allowed are the uppercase letters A-Z, digits 0-9 and the special characters - . $ / + % and space. The Apollo automatically calculates and appends the mod 43 check digit and adds the start and stop characters. Any required leading '+' characters must be sent in the data string.

Examples:

```
m i
J
O R
S I1;.0,.0,2.5,2.5,4.26
B:CODE39;.5,.35,0,HIBC,0.6,.015,3:1;+123AB78
B:39CODE;.5,1.00,0,hibc,0.6,.015,3:1;+123AB78
B:CO39DE;.5,1.70,0,H,0.6,.015,3:1;+123AB78
A 1
```



B - Bar Code Field Definition

Bar Code Name: CODABAR
Short Code: I

Command:

B [:name;] x, y, r, CODABAR, [+MOD16,] [+WSize,] [+BARS,] . . .

. . . **height,narrow,ratio; data <CR>**

Description:

Codabar is a variable-length bar code that can encode 16 different characters, including digits 0-9 and the special characters - . $ + : / . In addition, it requires an A B C or D as start/stop characters. A MOD16 check digit may be added by using the +MOD option.

Examples:

```
m i
J
O R
S I1;.0,.0,2.5,2.5,4.26
B:F0O1;1.0,0.1,0,CODABAR,.60,.013,3:1;A12345678A
B:F0O2;1.0,0.9,0,I,.60,.013,3:1;A23456789C
B:F0O3;1.11,1.70,0,CODABAR+MOD16,.60,.013,3:1;A13572468C
A 1
```

B - Bar Code Field Definition

Bar Code Name: MSI (MSI Plessey)
Short Code: K

Command:

B [:name;] x, y, r, MSI, [+MODxx,] [+WSize,] [+BARS,] . . .

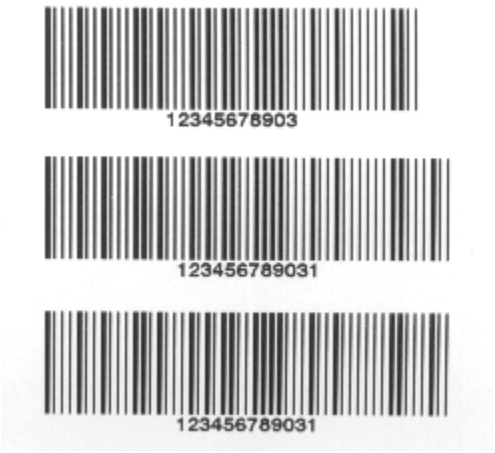
. . . height,narrow,ratio; data <CR>

Description:

MSI bar code is a variant of the Plessey bar code, which is a variable-length numeric only bar code. The Apollo automatically calculates and appends the correct Mod 10 check digit. If desired, additional check digits may be added with +MODxx.

Examples:

```
m i
J
O R
S l1;.0,.0,2.5,2.5,4.26
B:MSI  ;0.80,0.20,0,K,0.6,.013,2:1;1234567890
B:msi10;0.80,0.95,0,MSI+MOD10,0.6,.013,2:1;1234567890
B:MSI11;0.80,1.73,0,MSI+MOD11,0.6,.013,2:1;1234567890
A 1
```



B - Bar Code Field Definition

Bar Code Name: ADD-ON2 (UPC/EAN Addendum 2)
Short Code: M

Command:

B [:name;] x, y, r, ADDON2, [+BARS,] height,narrow; data <CR>
{ or SCx }

Description:

UPC/EAN Addendum 2 is a fixed-length numeric-only bar code addendum. This bar code is normally used in conjunction with symbologies UPC-A, UPC-E, EAN-8, EAN-13 and frequently represents the two-digit month of a periodical publication (i.e. 03 for March).

Because the addendum is an add-on to a separate bar code, it should be consistent with that base bar code. Therefore, the size specified, whether as height and narrow element, or as SCx, must match the size of the UPC-A, EAN-8 or EAN-13 base bar code. For positioning, the UPC/EAN Addendum must be placed a minimum of 9 times the width of the narrow element to the right of the main bar code. If an interpretation is printed, it will appear above the bar code.

Examples:

```
m i
J
H 5.2,0,T,R0
O R
S I1;.0,.0,2.5,2.6,4.26
B:BAR1;1.100,0.75,0,UPCA,SC2;01234567890
B:ADD4;2.519,0.75,0,ADDON2,SC2;09
A 1
```

B - Bar Code Field Definition

Bar Code Name: ADD-ON5 (UPC/EAN Addendum 5)
Short Code: N

Command:

B [:name;] x, y, r, ADDON5 [+BARS,] height,narrow ; data <CR>
{ or SCx }

Description:

UPC/EAN Addendum 5 is a fixed-length numeric-only bar code addendum. This bar code is normally used in conjunction with symbologies UPC-A, UPC-E, EAN-8, EAN-13 and frequently represents the price of a publication (i.e. 00399 for $3.99).

Because the addendum is an add-on to a separate bar code, it should be consistent with that base bar code. Therefore, the size specified, whether as height and narrow element, or as SCx, must match the size of the UPC-A, UPC-E, EAN-8 or EAN-13 base bar code.. For positioning, the UPC/EAN Addendum must be placed a minimum of 9 times the width of the narrow element to the right of the main bar code. If an interpretation is printed, it will appear above the bar code.

Examples:

```
m i
J
H 5.2,0,T,R0
O R
S I1;.0,.0,2.5,2.6,4.26
B:BAR1;1.1,0.75,0,EAN13,SC2;019876543210
B:ADD5;2.519,0.75,0,ADDON5,SC2;00399
A 1
```

B - Bar Code Field Definition

Bar Code Name: CODE 93
Short Code: O

Command**:**

B [:name;] x, y, r,CODE93,[+WSize,][+BARS,][+XHRI,] height,narrow; data <CR>

Description:

Code 93 is a variable-length alphanumeric bar code, which can encode all 128 ASCII characters including lower case and control characters. Special two-character sequences are used to designate some characters. The Apollo automatically calculates and appends the check digit. It also inserts the correct start and stop characters. The +XHRI option can be used to print the start and stop characters as a box (□) in the HRI below the bar code. The +BARS option can be used to place bearer bars at the top and bottom of the bar code.

Examples

```
m i
J
O R
S I1;.0,.0,2.5,2.5,4.26
B:F0O1;1.75,0.20,0,CODE93+XHRI,0.7,.011;ABC123
B:F0O2;1.75,1.00,0,code93,0.6,.011;ABC123
B:F0O3;1.75,1.70,0,O+BARS,0.7,.011;ABC123
A 1
```

B - Bar Code Field Definition

Bar Code Name: POSTNET (U.S. Postal Service)
Short Code: P

Command:

B [:name;] x, y, r, POSTNET, [+Wsize,]; data <CR>

Description:

Postnet is used by the United States Post Service to encode zip codes. Any length string of data can be used, although it is usually a nine digit zip code (zip + 4) followed by the check digit value. Sometimes a two digit post office code will be attached to the zip code. HRI (human readable interpretation) will not be printed with this bar code.

Individual bars, which make up the Postnet bar code, should be printed so that the height of the tall bars is 0.125 inch ± 0.010 inch, and the height of the short bars is 0.050 inch ± 0.010 inch.

Examples:

```
m i
J
O R
S l1;.0,.0,2.5,2.5,4.26
B:POSTNET1;1.0,.75,0,postnet,.5,.05;442120798
B:POSTNET2;1.0,1.5,0,P,.5,.05;441361234
A 1
```

B - Bar Code Field Definition

Bar Code Name: UCC128/EAN128
Short Code: Q

Command:

B [:name;] x, y, r, UCC128, [+WSize, +BARS,] height,narrow; data <CR>
{ or SCx }

Description:

UCC/EAN128 is a unique version of Code 128 used to encode serialized shipping container information. Each bar code has an application identifier embedded, and all applications share a similar bar code structure. Each UCC/EAN-128 bar code is composed of five (5) structural elements:

1. The Start Code. Start (B or C), FNC 1. The Apollo printer will generate a start character B or C FNC1 automatically. Start C will be used when the data begins with 4 or more numeric characters.

2. AI (Application Identifier) and Data. The number of data characters allowed in the bar code is determined by the Application Identifier chosen. See Appendix A for list of Application Identifiers. The AI is surrounded by parenthesis in the data string. The parenthesis will not be encoded in the bar code symbol, but will be printed in the HRI. If the correct number of digits are not supplied, the barcode will print with three question (?) marks, indicating an error.

3. The mandatory Mod 103 Symbol Check Character. (If a Mod 10 check character is needed, it must be calculated and included in the Data. The Apollo will not calculate this character; do not use +MOD10.)

4. The Stop Character.

5 The Quite Zones.

B - Bar Code Field Definition

Bar Code Name: UCC128/EAN128
Short Code: Q

Examples:

```
m i
J
O R
S l1;.0,.0,2.5,2.5,4.26
B:UCCEAN128;1.0,0.15,0,EAN128,0.6,0.013;(00)345678901234567890
B:UCC2;1.0,0.85,0,UCC128,0.6,0.013;(00)345678901234567890
B:UCC3;1.0,1.55,0,Q,0.6,0.013;(00)345678901234567890
A 1
```

B - Bar Code Field Definition

Bar Code Name: FIM (Facing Identification Mark)
Short Code: S

Command:

B [:name;] x, y, r, FIM, [+WSize,] [+BARS,] height,narrow; data <CR>

Description:

Facing Identification Mark is a 9 position bar/no-bar pattern. The FIM patterns are used by business mailers on preprinted mailing pieces for compatibility with various United States Postal Service automatic sorting systems. There are four FIM patterns (A, B, C and D) that can be printed on the Apollo printer, these are:

- A FIM-A is used on Courtesy Reply Mail, with preprinted Postnet symbology in the address.
- B FIM-B is used on Business Reply, Penalty, and Franked mail with no preprinted Postnet symbology.
- C FIM-C is used on Business Reply, Penalty, and Franked mail with preprinted Postnet symbology.
- D FIM-D is for OCR readable mail with no Postnet symbology

Examples:

```
m i
J
H 5.2,0,T,R0
O R
S I1;.0,.0,2.5,2.6,4.26
B:UCCODE;2.00,0.10,0,FIM,0.60,0.0133;A
B:UCCODE;2.00,0.90,0,FIM,0.60,0.0133;B
B:UCCODE;2.00,1.70,0,S,0.60,0.0133;C
A 1
```

B - Bar Code Field Definition

Bar Code Name: MAXICODE
Short Code: U

Command:

B [:name;] x, y, r, MAXICODE [+MODE]; [ZIP], [COUNTRY], [SERVICE],
. . . . . [MESSAGE] <CR>

Description:

Maxicode is a fixed-size matrix symbology which is made up of offset rows of hexagonal modules arranged around a unique finder pattern. The symbologies dimension is fixed at one inch by one inch. The symbology is use by the United Parcel Service for the tracking of packages. Maxicode has modes which are used to define the structuring of the data and error correction with a symbol. The available modes are as follows:

MODE2 Designed for the transport industry, Mode 2 is used in the US and encodes the zip code as numeric data.

MODE3 Designed for the transport industry, Mode 3 is used internationally and encodes the zip code as alphanumeric data.

MODE4 Mode 4 encodes a text message of 93 characters

MODE6 Mode 6 encodes a text message of 93 characters. This mode is used for reader programming since no data is transmitted.

ZIP 9 digit zip code for Mode 2, 6 characters for Mode 3.

COUNTRY 3 digit UPS country code

SERVICE 3 digit UPS service code

MESSAGE 84 character message for Mode's 2 and 3, 93 characters for Modes 4 and 6.

MaxiCode Special Content Fields

[U:ANSI_TM] Embeds the ANSI message header [)>$^{R}_{S}$01$^{G}_{S}$) into datastream. The header is only valid with Mode 2 and Mode 3. The data following the header is a two digit year, 5 or 9 digit zip code, 3 digit country code, 3 digit service code and a message.

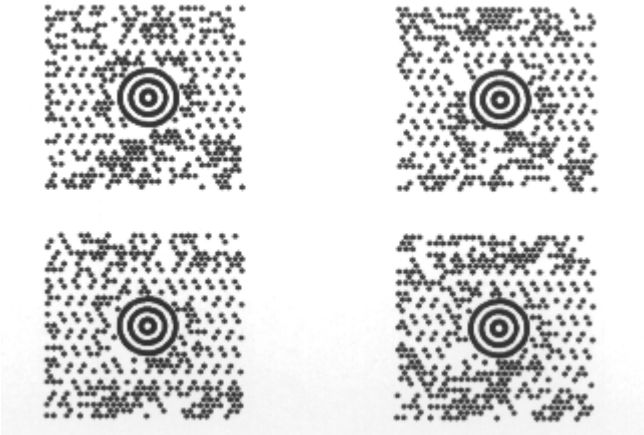
Example:
[U:ANSI_TM]98442120798,840,024,Tharo Systems, Inc.

B - Bar Code Field Definition

Bar Code Name: MAXICODE
Short Code: U

Examples:

m i
J
O R
S l1;.0,.0,2.5,2.5,4.26
H 5.2,0,T
B:MAXICODE;4.00,2.30,0,MAXICODE+MODE2;442120798,840,123,Parcel for Tharo Systems, Inc.
B:MAXICODE;4.00,1.20,0,MAXICODE+MODE3;ABC123,840,123,Parcel for International Zip Code
B:MAXICODE;1.50,2.30,0,MAXICODE+MODE4;Parcel for Tharo Systems, Inc.
B:MAXICODE;1.50,1.20,0,MAXICODE+MODE6;Parcel for Tharo Systems, Inc.(Reader Only)
A 1



B - Bar Code Field Definition

Bar Code Name: DATAMATRIX
Short Code: W

Command:

B [:name;] x, y, r, DATAMATRIX [+RECT],Height; data <CR>

Description:

DataMatrix is a variable size two-dimensional bar code symbology capable of encoding a number of different character sets, including all 128 ASCII characters. Every DataMatrix symbol consists of an array of data cells within a distinct perimeter pattern. DataMatrix can encode up to 2000 characters per symbol. DataMatrix can be printed in two forms, either as a square or as a rectangle.

For coding DataMatrix symbols, the following unique details apply:

- If data cannot be coded, an empty symbol of size 8 x 8 will be printed.
- Height parameter is the dimension of the cell size.

DataMatrix Special Content Fields

The following Special Content fields are available for DataMatrix only. Special Content fields are acronyms that allow special characters to be embedded by using standard keyboard characters.

[U:ANSI_AI] Embed ANSI Application Identifier [)>$^{R}_{S}$05$^{G}_{S}$ header and $^{R}_{S}{}^{EO}_{T}$ trailer.

[U:ANSI_DI] Embed ANSI Data Identifier [)>$^{R}_{S}$06$^{G}_{S}$ header and $^{R}_{S}{}^{EO}_{T}$ trailer.

[U:PROG] A Reader Programming character indicates that the symbol encodes a message used to program the reader system.

[ECE:x] Extended Channel Interpretation protocol allows the output data stream to have interpretations different from the default character set. Four broad types of interpretations are supported in DataMatrix:

1. International character sets (or code pages)
2. Encryption and compaction
3. User defined for closed systems.
4. Control information for structured append in unbuffered mode.

See DataMatrix specification for information on using Extended Channel Interpretations.

B - Bar Code Field Definition

Bar Code Name: DATAMATRIX
Short Code: W

DataMatrix Special Content Fields (cont.)

[APPEND:m,n,id1,id2] Up to 16 symbols may be appended in a structured format. This allows files of data to be represented in up to 16 DataMatrix symbols. The original data can be correctly reconstructed regardless of the order in which the symbols are scanned.

Where:

m = The number of the actual symbol. Value range is 1 to 16.

n = The number of all symbols together. Value range is 1 to 16.

id1 and id2 = A value between 1 and 254 which must be identical on all symbols.

Examples:

```
m i
J
O R
S I1;.0,.0,2.5,2.5,4.26
B:BAR1;1.50,0.50,0,DATAMATRIX,.04;Tharo Systems, Inc.
B:BAR2;1.50,1.75,0,W+RECT,.04;Tharo Systems, Inc.
A 1
```

B - Bar Code Field Definition

Bar Code Name: PLESSEY
Short Code: X

Command:

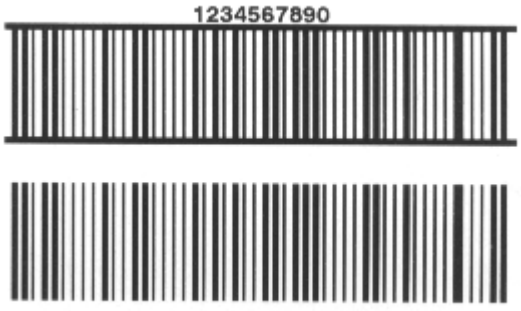
B [:name;] x, y, r, PLESSEY, [+WSize,] [+BARS,] height,narrow,ratio; data <CR>

Description:

Plessey bar code is a variable length non-self checking symbology that can encode the digits 0-9 and letters A-F. This symbology is not in widespread use, and is usually seen only in older library and grocery applications. When bearer bars are desired at the top and bottom of the bar code, the +BARS option should be used.

Examples:

```
m i
J
O R
S l1;.0,.0,2.5,2.5,4.26
B:PLESSEY;0.75,0.50,0,PLESSEY+BARS,0.7,.015,2:1;1234567890
B:plesse2;0.75,1.30,0,plessey,0.6,.015,2:1;1234567890
A 1
```



B - Bar Code Field Definition

Bar Code Name: UPC-E0
Short Code: Y

Command:

B [:name;] x, y, r, UPCE0, [+WSize] [+BARS] [+XHRI] height,narrow; data <CR>
{ or SCx }

Description:

UPC-E0 is a fixed-length of eight, numeric-only bar code. The Apollo accepts 11 or 12 digits with the command, where the first digit must be a zero. Leaving the leading zero in position 1, the remaining 10 digits are compressed with zero suppression to six digits, filling positions 2-7 in the final number. Based on these six digits, Apollo automatically calculates and appends a check digit as the eighth.

Examples:

```
m i
J
O R
S I1;.0,.0,2.5,2.5,4.26
B:UPCE0A;1.0,0.22,0,UPCE0,0.7,.013;03210000678
B:UPCE0B;1.0,1.32,0,Y,SC1;01230000088
A 1
```

B - Bar Code Field Definition

Bar Code Name: PDF417
Short Code: Z

Command:

B [:name;] x, y, r, PDF417, [+WSize,] [+ELx,] height,narrow,dimensional ratio; data <CR>

Description:

PDF417 is a two-dimensional bar code symbol capable of encoding information at about one thousand bytes per symbol. Every PDF417 symbol is composed of a stack of rows. Each row consists of several code words. Each code word is constructed of 17 modules arranged into four bars and four spaces. The first and last code words in each row are row indicators. There are two checksums for a PDF417 symbol. For coding PDF417 symbols, the following unique details apply:

- Error levels of 00 to 08 may be specified with +ELxx on the Apollo printer. The higher the error level, the more redundancy that will be included in the symbol, and the larger the symbol will appear.
- Carriage return/line feed can be inserted into the text by using Unicode [U:13][U:10] or given as HEX values [U:$0D] [U:$0A].

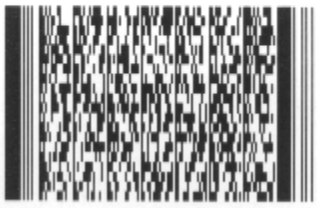
Due to a restriction in the Apollo printer, when printing a PDF417 bar code, the row height specification must be at least three times the narrow element width specification.

Example: If a bar code is created with a narrow element width of .34mm, and a row height of less than 1mm, the printer will force the total row height to 1mm. (.34mm x 3 = 1.02).

B - Bar Code Field Definition

Bar Code Name: PDF417
Short Code: Z

Examples:
m i
J
O R
S l1;.0,.0,2.5,2.5,4.26
B:BAR1; 0.75,0.30,0,PDF417+EL0,0.05,.011,.26;THARO SYSTEMS INC[U:13][U:10]2866 NATIONWIDE PKWY[U:13][U:10]BRUNSWICK OH 44212
B:PDF417;0.75,1.00,0,Z+EL5,0.05,.011,.66;THARO SYSTEMS INC[U:CR][U:LF]2866 NATIONWIDE PKWY[U:CR][U:LF]BRUNSWICK OH 44212
A 1



B - Bar Code Field Definition

Bar Code Name: QRCODE
Short Code: N/A
Note: This command is supported by A-Series printers ONLY!

Command:

B [:name;] x, y, r, QRCODE [+ELx][+MODELx][+WSize], size; data <CR>

Description:

Denso of Japan developed QR Code (Quick Response Code) as a 2D matrix symbology having position detection patterns on its three corners. QR Code features mass data capacity, high data density and error correction ability, ultra high speed, omni-directional reading and Japanese Kana-Kanji representation. QR Code is in the public domain and it is read by using CCD array cameras and image-processing technology because of the layout of the finder pattern.

QR Code symbols are square in shape and can easily be identified by their finder pattern of nested alternating dark and light squares at three corners of the symbol. Maximum symbol size is 177 modules square, capable of encoding 7366 numeric characters, or 4464 alphanumeric characters. One important feature of the symbology is its ability to encode directly Japanese Kanji and Kana characters. QR Code is designed for rapid reading using CCD array cameras and image processing technology because of the layout of the finder pattern.

- Error levels of 1 to 4, L, M, Q or H with +ELx on the printer. The default value is 1.
- The MODEL can be 1 for the original version, or 2 for the enhanced version with +MODELx on the printer. The default value is 1.

Example:
```
m i
J
O R
S l1;.0,.0,2.5,2.5,4.00
B:BAR1; 0.75,0.75,0,QRCODE+ELL+MODEL1,0.04;THARO SYSTEMS INC[U:13][U:10]2866
NATIONWIDE PKWY[U:13][U:10]BRUNSWICK OH 44212
A 1
```

C - Cutter Parameters

The **C** command sets the cutter parameters. Cutting can be set specifically by counting labels, with or without additional displacements, or the 'end' of the current job can be selected.

Usage:

C cnt [,disp1 [,disp2]] <CR>
C e <CR>

Where:

'**cnt**' = A number of labels after which the cutter should cut.

'**disp1**' = The displacement from the end of the label for cutting in inches or millimeters. This specifies where, within the gap between labels, the cut should be made.

'**disp2**' = An offset from the first cut for a second cut in inches or millimeters. This second cut could be used to cut out reflective marking on continuous (endless) media.

e = Sets the cutting to be performed once at the **e**nd of the current job. (End of job is determined by the label count in the A command).

Example:

C 5,0,.20 <CR> Sets cutting at every five labels, cutting after an offset of 2/10th extra inches past the fifth label's trailing edge.

D - Global Object Offset

The **D** command moves the origin of all objects on a label the specified values. This command is useful if you change from one label size to another, and need to center the data on the new label without changing the values of each object. However, if you exceed the limits of your label size, your data will be cut off. The preferred method for moving objects is using the S command - Set Label Size which will not cut off your data if you exceed the limits of your label.

Usage:

D x,y <CR>

Where:

x = Specifies the horizontal amount of offset. The unit of measurement, either inches or millimeters, is set by the Immediate Command “m”

y = Specifies the vertical amount of offset. The unit of measurement, either inches or millimeters, is set by the Immediate Command “m”

Example:

D 0.5, 0.5 <CR> Sets the Global Offset at 1/2 inch horizontally and 1/2 inch vertically. Whatever values are defined for all other fields on the label, will be offset by this amount.

E - Define Files

The **E** command defines database, serial and log files for use on the PCMCIA memory card.

Note: See Appendix B for sample programs and text files demonstrating E commands.

E DBF;name <CR>

This command must be added to formats that access fields from a database.

Where:

name = Specifies the name of the database on the PCMCIA memory card. The database must be present on the memory card.

Example:

E DBF;ZIPCODE <CR> Defines the database ZIPCODE.DBF for use on the PCMCIA memory card.

E TMP;name <CR>

Where:

name = Specifies the name of the serial file on the PCMCIA memory card.

Example:

E TMP;SERIAL <CR> Defines the serial file SERIAL.TMP for use on the PCMCIA memory card.

E LOG;name <CR>

Where:

name = Specifies the name of the log file on the PCMCIA memory card.

Example:

E LOG;TRACK <CR> Defines the log file TRACK.LOG for use on the PCMCIA memory card.

F - Font Number

The **F** command allows assignment of an optional, alternate number for a font. This alternate font number can simplify use, make formats easier to read, and provide a simple way to replace the font throughout an entire format. With this command, the font identifier would be changed on this single command, instead of in each individual text format command.

On TrueType fonts, the number found in the typeface file is used as the default.

Usage:

F number;name <CR>

Where:

F = Font command

number = Alternate number to assign to the font.

name = Name of the font selected. For a downloaded font, this will be the name assigned at download. For internal fonts, the font number shown in the printer self-test should be used to identify the font.

All fonts on Apollo have a default number which is encrypted in the font header. This number can be found on the Apollo's status printout.

For example, a font "Aurora" might have a number such as '1137'. This number, 1137, would be used in each text field definition command, every time the user wants to use the Aurora typeface. To simplify font selection, the user may create their own alternate number. The user could code 'F 5;Aurora' (for a downloaded font), or 'F5;1137' (for an internal font) early in the label command set. Then, in each text field definition command, the '5' could be used to specify font.

If the user wishes to change the Aurora typeface to Arial at some later time, they could accomplish this by changing the 'F 5; Aurora' to 'F 5;Arial'. This would eliminate the need for changing each individual text field definition command.

Example:

F 4; Swiss 721 <CR> Assigns user's chosen alternate number 4 to Swiss™ 721 font.

G - Graphic Field Definition

The **G** command specifies a graphic field definition statement to add a line, rectangle or ellipse field to the current label format.
Usage

G [:name;] x, y, r; type:type options [,shade options] [,outline options] <CR>

Where:

G = Graphic field definition command.

[:name;] = Optional parameter. A unique name given to this graphic field. “name” must begin with a colon may be up to ten characters long, and may not contain any special characters. The field “name” may be used with the R - Replace Command. See the R - Replace Command for instructions on using this command.

x = Coordinate "x" specifies the horizontal position of the graphic field. This is the distance, in inches or millimeters, from the left edge of the printable area to the start position of the graphic field. Start position of the graphic varies by graphic type; see individual type descriptions. The printable area is defined by the Label Size Command “S". The unit of measurement, either inches or millimeters, is set by the Immediate Command “m".

y= Coordinate "y" specifies the vertical position of the graphic field. This is the distance from the top of the label to the start position of the graphic field. Start position of the graphic varies by graphic type; see individual type descriptions. The unit of measurement, either inches or millimeters is set by the Immediate Command “m".

r= Rotation of the graphic field. Rotation may be given in any one of 360 degrees. In determining rotation, consider the current orientation of the label, as specified by the Option command, Rotate parameter.

type = Type identifies the shape of the graphic, where:
C = Circle
L = Line
R = Rectangle
See Type and Type Options definitions on the following pages.

type options = The graphic’s type options specify the size of the object in terms appropriate for the object’s shape. See Type and Type Options definitions on the following pages.

G - Graphic Field Definition

Graphic Type: C - Circle

Command:

G [:name;] x, y, r, **C: radius1 [,radius2 [,width]]** [,shade options] [,outline options]

Type Options:

radius1 = Radius of circle in inches or millimeters, or horizontal radius (in relation to rotation 0) if ellipse.

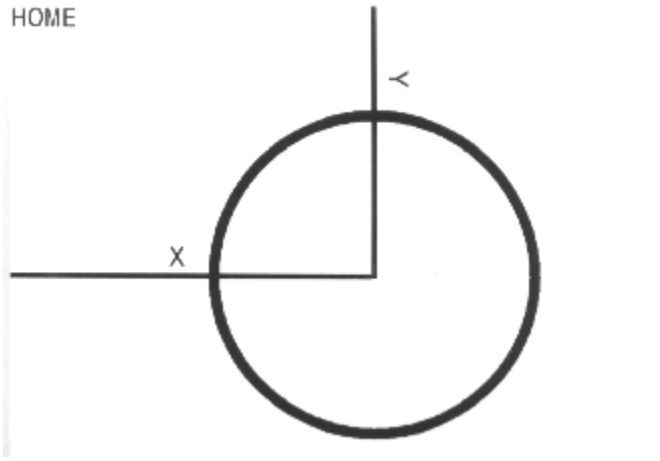
radius2 = Radius2 is the vertical radius (in relation to rotation 0) of an ellipse in inches or millimeters.

width = Width of line in inches or millimeters.

Note: x, and y, coordinates point to the **center** of the circle or ellipse. See example below.

Example:

```
m i
J
O R
S l1;.0,.0,2.5,2.5,4.26
G 1.83,1.35,0;C:0.83,0.83,0.05
G 0.00,1.35,0;L:1.82,0.03
G 1.83,0.00,270;L:1.32,0.03
T:TEXT1;0.01,0.11,0,596,pt10;HOME
T:TEXT2;0.80,1.30,0,596,pt10;X
T:TEXT3;1.90,0.33,270,596,pt10;Y
A 1
```



G - Graphic Field Definition

Graphic Type: L - Line

Command:

G [:name;] x, y, r, **L: length, width [,start [,end]]** [,shade options] [,outline options]

Type Options:

length = Length of the line in inches or millimeters.

width = Width of the line in inches or millimeters.

start = Type of line start **(s**=squared (default), **r**=rounded, **a**=arrowed).

end = Type of line end (**s**=squared (default), **r**=rounded, **a**=arrowed).

Note: x, and y, coordinates point to the **center** of the starting point of the line. See example below.

Examples:

```
m i
J
H 5.2,0,T,R0
O R
S l1;.0,.0,2.5,2.5,4.26
T:Text1;0.10,0.10,0,596,.10,h.10;HOME
G .01,1.15,0;L:1.2,.015
G 1.23,1.15,90;L:1.32,.015
G 1.23,1.15,0;L:2.45,.22
T:XTEXT;.615,1.13,0,596,.10,h.10;X
T:YTEXT;1.25,.575,270,596,.10,h.10;Y
A 1
```

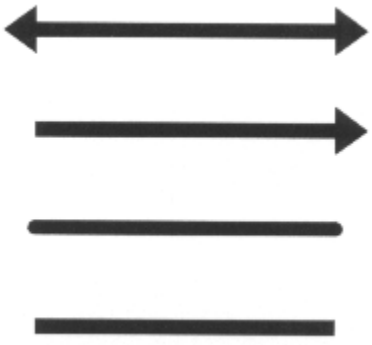
G - Graphic Field Definition

Graphic Type: L - Line

Examples:

The following example shows the different line ends available.

```
m i
J
OR
S l1;.0,.0,2.5,2.5,4.26
G 1.40,0.50,.0;L:1.50,0.08,a,a
G 1.40,1.00,.0;L:1.50,0.08,s,a
G 1.40,1.50,.0;L:1.50,0.08,r,r
G 1.40,2.00,.0;L:1.50,0.08
A 1
```



G - Graphic Field Definition

Graphic Type: R - Rectangle

Command:

G [:name;] x, y, r, **R: hor,ver [,horw [,verw]]** [,fill option] [,shade options] [,outline]

Type Options:

hor = Horizontal length of the rectangle in inches or millimeters.

ver = Vertical length of the rectangle in inches or millimeters.

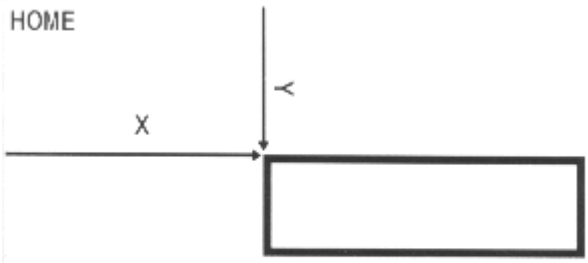
horw = Width of horizontal lines in inches or millimeters.

verw= Width of vertical lines in inches or millimeters.

Note: x, and y, coordinates point to the **top right, outside corner** of the rectangle. See example below.

Example:

```
m i
J
O R
S I1;.0,.0,2.5,2.5,4.26
G 1.30,0.75,0;R:1.60,0.50,0.04,0.04
G 0,0.75,0;L:1.25,0.02,s,a
G 1.30,0,270;L:0.69,0.02,s,a
T:X;0.65,0.65,0,596,pt10;X
T:Y;1.35,0.375,270,596,pt10;Y
T:TEXT1;0.03,0.12,0,596,pt10;HOME
A 1
```



G - Graphic Field Definition

Graphic Option: Fill

Fill provides for the filling of all graphic objects with a specified pattern or dot density.

Command:

G [:name;] x, y, r, type:type options**, [F:name]** [shade] [outline]

Fill Parameters:

F: = Fill parameter.

name = Name of the fill pattern desired from the following:
0%, 6%, 12%, 25%, 38%, 50%, 100% (for dot density)
left, right, grid, diamond, dots (predefined patterns)
user1, user2, user3, user4 (downloaded images)

Note: Downloaded user images should be 32 by 32 dots.

Note: The fill option with its parameter must be coded in brackets [ ].

Example:

```
m i
J
O R
S l1;.0,.0,2.5,2.5,4.26
H 4,0,T,R0
G :square1;0.50,0.75,0;R:0.75,0.80,0.40,0.40[F:grid][O]
G :circle1;2.20,1.22,0;C:0.37,0.37,0.37[F:38%][O]
G :square2;3.20,0.75,0;R:0.75,0.80,0.40,0.40[F:left][O]
A 1
```

G - Graphic Field Definition

Graphic Option: Shade

Shade provides for the gradient filling of all graphic objects.

Command:

G [:name;] x, y, r, type:type options**, [S:percent1[,percent2[,direction]]]** [outline

Shade Parameters:

S: = Shade parameter.

percent1 = Beginning value of darkness, as a percent of black.

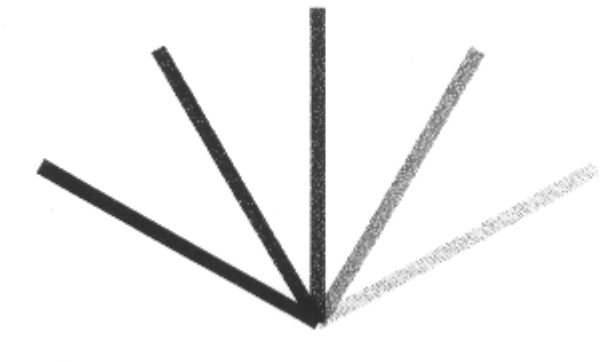
percent2 = Ending value of darkness, as a percent of black. Using this parameter causes a gradient shading, rather than a constant shading.

direction = Angle of shading, if other than horizontal gradient is needed.

Note: The shade option with its parameters must be coded in brackets [ ].

Example:

```
m i
J
O R
S l1;.0,.0,2.5,2.5,4.26
G 2.30,1.90,150.5;L:1.60,0.08
G 2.30,1.90,120.0;L:1.60,0.08[S:60]
G 2.30,1.90,090.6;L:1.60,0.08[S:45]
G 2.30,1.90,060.8;L:1.60,0.08[S:30]
G 2.30,1.90,030.2;L:1.60,0.08[S:15]
A 1
```



G - Graphic Field Definition

Graphic Option: Outline

Outline prints a one dot outline around any filled graphic object**.**

Command:

G [:name;] x, y, r, type:type options [shade options] **[O]**

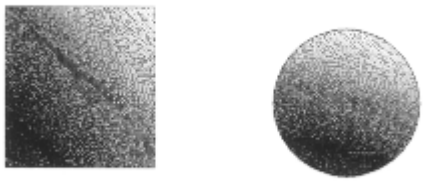
Parameter:

[O] = Outline is to be applied to this filled object.

Note: The outline parameter must be coded in brackets [ ].

Example:

```
m i
J
O R
S l1;.0,.0,2.5,2.5,4.26
G 0.75,0.75,0;R:0.75,0.80,0.40,0.40[S:60,10,45][O]
G 2.45,1.22,0;C:0.37,0.37,0.37[S:60,10,75][O]
A 1
```



H - Heat, Speed, Method of Printing

The **H** command defines the speed, heat and method of printing to be used for the current label.

Usage:

H speed [,heat] [,method] [,ribbon] <CR>

Where:

speed = Specified in inches or millimeters per second (depending on current measuring unit). For example, with inches 5.2 is 5.2 inches/second, and with millimeters 100 is 100 mm/second. Standard speeds per second available:

Apollo 1 and Apollo 2:
2.6", 3.9", 5.2", 6.3", 7.9" and 66mm, 100mm, 133mm, 160mm, 200mm.

Apollo 3, Apollo 4 and A8:
2.0", 3.0", 3.9", 4.9" and 50mm, 75mm, 100mm, 125mm.

Apollo 3/300 and Apollo 4/300:
1.3", 2.0", 2.5", 3.0", 3.4" and 33mm, 50mm, 63mm, 75mm, 87mm.

A3/200:
2.0", 3.0", 3.9", 4.9", 5.9", 6.9", 7.9" and 50mm, 75mm, 100mm, 125mm, 150mm, 175mm, 200mm

A3/300:
2.0", 3.0", 3.9", 4.9", 5.9" and 50mm, 75mm, 100mm, 125mm, 150mm

Any speed coded that is not one of the standard speeds will be rounded to the next existing speed. If this parameter is not used, the default for Apollo 1 and 2 is 160 mm/sec. The default for Apollo 3 and 4 is 100mm/sec. The default for Apollo 3/300 and 4/300 is 75mm/sec.

heat = Specified as a value which ranges from -10 to +10. Default is 0.

method= Specify as **T** for transfer printing and **D** for direct thermal printing. Default is set on front panel.

ribbon = Controls ribbon saving when using **T**, transfer printing method. Specify **R0** for ribbon saving off, **R1** for ribbon saving on. Default is set on front panel. Ribbon Saver is only available on Apollo 1.

Example:

H 5.2,0,D <CR>	Sets speed to 5.2 in/sec, nominal heat and direct thermal printing.
H 3.9,T,R1 <CR>	Sets speed to 3.9 in/sec, heat to +1, thermal transfer printing, with ribbon saving on.

I - Image Field Definition

The **I** command is used to place a graphic image field in the current label format. The graphic image must have been previously downloaded (see Immediate Command “d” and Appendix B for information on downloading graphics).

Usage

I [:name;] x, y, r [,mx,my] ;imagename <CR>

Where:

I = Image field definition command.

[:name;] = Optional parameter. A unique name given to this image field. “name” must begin with a colon may be up to ten characters long, and may not contain any special characters. The field “name” may be used with the R - Replace Command. See the R - Replace Command for instructions on using this command.

x = Coordinate "x" specifies the horizontal position of the image field. This is the distance, in inches or millimeters, from the left edge of the printable area to the start position of the image field. The printable area is defined by the Label Size Command “S". The unit of measurement, either inches or millimeters, is set by the Immediate Command “m".

y= Coordinate "y" specifies the vertical position of the image field. This is the distance from the top of the label to the start position of the image field. The unit of measurement, either inches or millimeters, is set by the Immediate Command “m".

r= Rotation of the image field. Rotation may be given in any one of 360 degrees. In determining rotation, consider the current orientation of the label, as specified by the Option command, Rotate parameter.

mx,my = Optional parameters. These are magnification factors, 1 to 10, to be applied for horizontal and vertical expansion of the image. “mx” is horizontal magnification and “my” is vertical magnification.

imagename = The name of the previously downloaded graphic image to be used. This name was stored in the printer’s memory by the Immediate Command “d”. (See Appendix B for download example)

Example:

I:PIC1;.01,.05,0,2,4;LOGO Print the image named LOGO, magnified by 2 horizontally and by 4 vertically.

J - Job Start

The **J** command starts a new label definition job by setting all parameters to their default values.

Usage:

J [comment] <CR>

Where:

J= Job start command.

comment = Optional parameter. Text which describes the label job. This text may be printed on the printer's LCD while selecting a label from memory card, giving a better idea of the label than 8 byte DOS-type filenames. The comment can be up to 16 characters in length.

Example:

J Test Label — Sets the job start command and names the format 'Test Label'. When the label is recalled for printing from the memory card, 'Test Label' will be displayed on the printer's LCD.

M - Memory Card Access

The **M** commands provide access to the memory card for listing its' contents, and for deleting, formatting, loading, and storing data.

Note: See Appendix B for sample programs and text files demonstrating M commands.

Command Formats:

For the following Memory commands, where **type** is shown, specify the file type as:

FMT -	Media format definition (Label size, heat, speed, method, etc. can be specified separately from the format definition if desired)
FNT -	Font file (any font file, including SPD, TTF, etc.)
	IMG - Image file (any image file, including BMP,GIF, IMG, MAC, PCX, TIF, etc.)
LBL -	Label format definition (any label command set)
DBF -	Database definition

M c <CR> contents list (directory) of the card.

Produces a list similar to the following, with date shown as dd:mm:yy :

```
Directory of 'TESTCARD  '
TESTFONT  FNT    338  14:03:95 15:10
TESTLABL  LBL   3530  14:03:95 15:45
1024512 bytes free
```

M d type; name<CR> **d**eletes a file from the card.

where:

type = As defined above.

name = The datafile name following DOS conventions, up to 8 characters.

Example: M d FNT;MYFONT Deletes FNT named MYFONT from card.

M - Memory Card Access (Cont.)

Command Formats:

M f; name<CR> **f**ormats a memory card with the correct file system. RAM cards are formatted using a FAT file system. Flash cards will be formatted using the Flash File 2.0 system.

where:

name = The name to be assigned to the memory card, up to 8 characters.

Example: M f;myfiles Formats the card and names it 'myfiles'.

M l type; name<CR> **l**oad a datafile into the printer from the memory card.

where:

type = As specified above.

name= The datafile name following DOS conventions, i.e., 8 character name.

Example: M l IMG;MYIMAGE Loads IMG named MYIMAGE from the card.

M - Memory Card Access (Cont.)

Command Formats:

M s type; name<CR> **s**tores data on the memory card. Command is used to start and end store operation.
where:

type = As specified above.

name The datafile name following DOS conventions, i.e., 8 character name. When storing a label, this command is also used at the end of the label data stream to stop the store process.

Note: Images and Fonts to be stored must be downloaded prior to issuing this command. Label and Media format statements follow the store command as shown below.

Example:	M s LBL;MYLABEL	Begins label storing
	. . . Label commands are sent next . . .	Label data here
	M s LBL	Ends label storing

M u type; name<CR> **U**pload a datafile from the memory card to the host.

where:

type = As specified above.

name= The datafile name following DOS conventions, i.e., 8 character name.

Example:	M u LBL;MYLABEL	Uploads label named MYLABEL from the card. During a Hyper-Terminal session, this text can be copied with a <Ctrl>C and Pasted with a <Ctrl>V into a file opened in text mode, such as Notepad.

Attention: When uploading other types of files, such as IMG, the data is in sent in it's raw binary form. The only exception is the <ESC> conversion to a double, <ESC><ESC>, that's also used in the d (download) command. See Appendix B for an example Basic program showing this conversion.

O - Set Print Options

The **O** command provides a variety of options to be used while printing a label.

Usage:

O [M,][R,][S,][T,][N,][U,][p] <CR>

Where:

O = Print Options command.

M= M activates the **M**irrored image effect in printing.

R = **R** causes the Home position for the label to be **R**otated by 180 degrees. Without this command, Home is at the bottom right of the label, as viewed from the front of the printer. With this command, Home positioned at the top left of the label, as viewed from the front of the printer. (See Command Overview section for more information)

S= **S** switches the printer to **S**ingle Page Buffering. The next format is not imaged until the current label is finished printing.

T = **T** places the printer in **T**ear-off mode. This is used to advance the label forward to the tear plate, then backfeed and print the next label.

N = N causes **N**egative image printing, which results in white characters on a black background.

U = **U** provides for Sec**u**rity by eliminating label re-prints. Pressing the PSE key or the F2/F3 on an external keyboard will not re-print the label.

p = Enables the printer to overlap the operations of starting to print a new label, while the old one is being cut or **p**eeled off without backfeeding the label. This feature can also be set as a default in the printer's setup using the 'Backfeed' option. Using **P** activates this feature and is the same as selecting 'smart' in the printer's setup. Using **D** the printer will backfeed before printing the next label. This is the same as selecting 'always' in the printer's setup. We recommend using the **D** option to avoid lines or imperfections on your fields. These commands function identically to the 'always' and 'smart' options available under 'Backfeed' from the printer's setup.

Examples

O R,T <CR> Rotates orientation to place Home position at top left of label and places printer in Tear-off Mode.

P - Set Peel-Off Mode

The **P** command sets the printer in peel-off mode, which causes it to pause after each label is printed, allowing the labels to be peeled off individually. An error will occur if a present sensor is not attached to the printer.

Usage:

P [disp] <CR>

Where:

P = Set Peel-Off command.

disp = Optional parameter. A displacement in inches or millimeters (dependent on the m - set measuring unit command) relative to the normal peel-off position of the label set from the front panel setup. The displacement can be a negative or positive value and has no range limitation. After advancing the label the specified amount, the printer will stop to let you peel off the label, or enable it to be grabbed for handling by an applicator. Proper adjustment with this command will enable proper applicator grasping of various size labels or successful peeling of labels.

Example:

P <CR> Sets peel-off mode

P .25 <CR> Sets peel-off mode and adjusts the placement of the label edge .25 inches past the default peel position.

P -.25 <CR> Sets peel-off mode and adjusts the placement of the label edge .25 inches before the default peel position.

R - Replace Field Contents

The **R** command provides for replacing the contents of a field, without recoding an entire label command set to include the new value. This option would be very useful where the user did not want to retransmit the entire label, only the fields that change. This option does work with graphics, but with limitations. You cannot change the position or rotation of the graphic image. If the graphic that is replacing the first graphic is larger, the graphic field could overlap or even cover other fields on the label. The graphic that is replacing the original graphic must be previously downloaded using the ‘d - Download Data’ command.

Usage:

R name; newcontent<CR>

Where:

R = Replace command.

name; = The name of the text data field or bar code data field.

newcontent = The new value of the field, which will replace it’s former value. This field may be any length, and does not have to match the former length.

Example:

R article;01234512345 <CR> Sets new content for field named 'article'.

R PCX;FISH <CR> Sets new image for field named 'PCX'. The image must have been downloaded previously using the ‘d - Download Data’ command.

S - Set Label Size

The **S** command provides for definition of the label size in length and width, characteristics and repetition.

Usage:

S [type;] xo, yo, length, dy, wide [,dx ,col] [;name] <CR>

Where:

S = Set label size command.

Type; = Optional parameter. Specifies the type of labels from:
- **e** for continuous (endless) media with no die cuts.
- **l0** (lowercase L, zero) for labels with reflective marker on the upper side of media. The **l0** option is only available on the Apollo 1 and Apollo 2.
- **l1** (lowercase L, 1) for die cut labels with gap sensor detection (default).
- **l2** (lowercase L, 2) for labels with reflective marker on the lower side of media.

xo= The horizontal displacement, in inches or millimeters, to move the print on the label, shifting the starting point of print to the side - the left margin.

yo = The distance, in inches or millimeters, from the top of the form to the starting point of print - the top margin.

length = The length of the label, in inches or millimeters.

dy = The distance from the starting point of the one label to the starting point of the next label, in inches or millimeters.

wide = The width of one label, in inches or millimeters.

dx = The horizontal distance between labels, from the start of one label to the start of the next label, in inches or millimeters.

col = The number of labels across the label roll.

name = Optional parameter. The media name, which may be displayed on the LCD to show the user which media he has to insert.

Example:

S l1; 0,0, 6.00, 6.10, 4.00 <CR> Sets size of the label to six inches by four inches, with no horizontal or vertical displacement.

T - Text Field Definition

The **T** command is used for text field definition**,** to add a text field to the current label format. The text may be printed in any rotation, but most often text is printed in one of four rotations: 0, 90, 180 or 270 degrees. Bitmap fonts can only print in rotations of 0, 90, 180 or 270 degrees. The Apollo printer has eight internal character sets and the ability to generate character bit maps from downloaded TrueType fonts.

Usage:

T [:name;] x, y, r, font,size [, effects]; data <CR>

Where:

T = Text field definition command.

:name; = Optional parameter. A unique name given to this text field. May be used later to concatenate fields, in field data replacement or in field calculations and comparisons “name” may be up to ten characters long, no comma, colon or semicolon allowed.

x= Coordinate "x" specifies the horizontal position of the text field. This is the distance, in inches or millimeters, from the left edge of the printable area to the start position of the text field. The printable area is defined by Label Size command “S". Start position refers to the upper left corner of the text field area. The unit of measurement, either inches or millimeters is set by the Immediate Command “m".

y= Coordinate "y" specifies the vertical position of the text field. This is the distance, in inches or millimeters, from the top of the label to the baseline of the text field. Baseline refers to the bottom corner of the text field area. The diagram below illustrates the position of the baseline. The unit of measurement, either inches or millimeters is set by the Immediate command “m”.

r = Rotation of the text field. The rotation may be given in any one of 360 degrees, but is most often given as 0, 90, 180 or 270 degrees. Bitmap fonts can only print in rotations of 0, 90, 180 or 270 degrees. In determining rotation, consider the current orientation of the label, as specified by the Option command, Rotate parameter.

T - Text Field Definition

font = The number of the font used, which provides selection of the style for printing the text string. Either one of the printer's own internal character sets or a downloaded font can be used. To obtain the information contained in the Font file, perform a printer self test (see Immediate Command "t"). The internal character sets available are:

Font No	Name	Type	Description
-1	_DEF1	Bitmap	Default Size 12x12 dots
-2	_DEF2	Bitmap	Default Size 16x16 dots
-3	_DEF3	Bitmap	Default Size 16x32 dots
-4	OCR_A_I	Bitmap	OCR-A Size I
-5	OCR_B	Bitmap	OCR-B
3	BX000003	Scaleable	Swiss 721(TM)
5	BX000005	Scaleable	Swiss 721 Bold(TM)
596	BX000596	Scaleable	Monospace 821

size = Defines the size of the characters.

- With Scaleable fonts, size may be set by point size "**pt** x", in hundredths of an inch, or in millimeters.

- For bitmapped fonts, a magnification factor may be used for vertical and horizontal expansion of the characters "**x** mx, **y** my", where mx and my are expansion values 1 to 10 (see text samples on the following pages).

T - Text Field Definition

effects = Optional parameters. Describes special effects to be applied to the characters. Depending on the font not all effects may be available.

The following effects are available for most fonts:

b = bold
s = slanted
i = italic
n = negative (reverse print)
u = underlined
l = light
z = slanted left
k = kerning is auto-adjusting of the space between printed characters.
v = print text in vertical alignment.
q*n* = squeeze characters, where *n* is a percentage (10 -1000 only, default percentage is 100)
h*n* = width of upper case “H” , with *n* in inches or millimeters.
m*n* = horizontal spacing of text, with *n* in inches or millimeters.

The following effects are available only with internal bitmapped fonts:

o = outlined (not for OCR font)
g = gray (not for OCR font)
x*n* = horizontal expansion factor, where *n* = 1 to 10
y*n* = vertical expansion factor, where *n* = 1 to 10

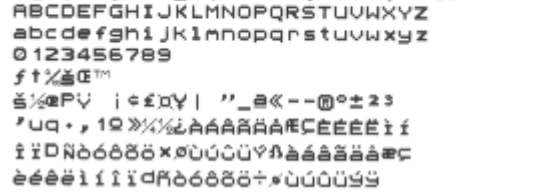
data = string of ASCII, EBCDIC, CODEPAGE 850, CODEPAGE 852, MACINTOSH, ISO 8859-8, WINDOWS 1250, WINDOWS 1252 or ISO 8859-1 characters to be printed as text on the label format. The string of characters representing 'data' may also consist of special content fields, including calculations (see Special Content Fields section). Note: Selecting a codepage is done during front panel setup. The option is labeled ‘Character Set’.

T - Text Field Definition

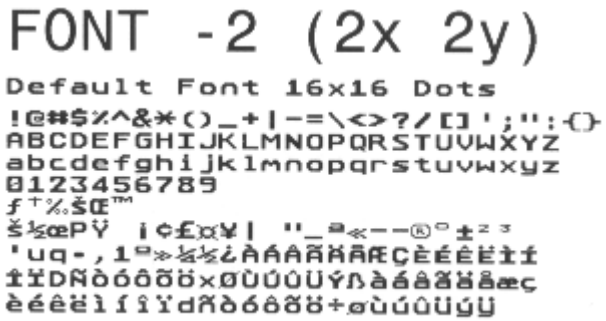
Internal Fonts

The following are examples of the Apollo's internal bitmapped fonts. Character sets labeled 2x 2y are shown doubled in size for better readability.

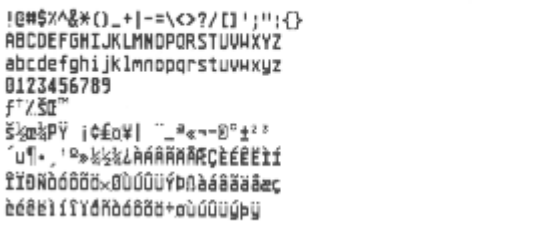
FONT -1 (2x 2y)



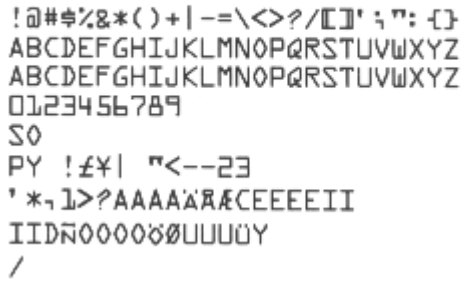
FONT -2 (2x 2y)



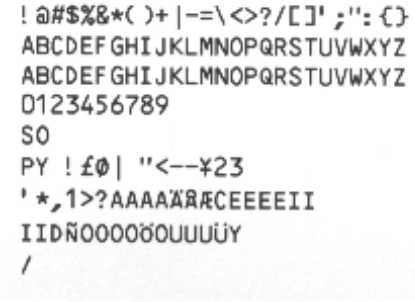
FONT -3 (1x 1y)



FONT -4



FONT -5



T - Text Field Definition

Internal Fonts

The following are examples of the Apollo's internal smooth (scaleable) fonts.

FONT 3
SWISS 721 (TM)
!@#$% ^ &*()_+|-=\<>?/[]';":{}
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqurstuvwxyz
0123456789
□š□¶□·□□ÒÓÔØ×Þ□□□''â™ãêëY™
š½œ¾PŸµÕàéŸ¥¦ ¨ª«¬-®¯°±23
´µ¶· ¹º»¼½¾¿ÀÁÂÃĀÅÆÇÈÉÊËÌÍ
ÎÏÐÑÒÓÔÕÖ×ØÙÚÛÜÝÞßàáâãäåæç
èéêëìíîïðñòóôõö÷øùúûüýþÿ

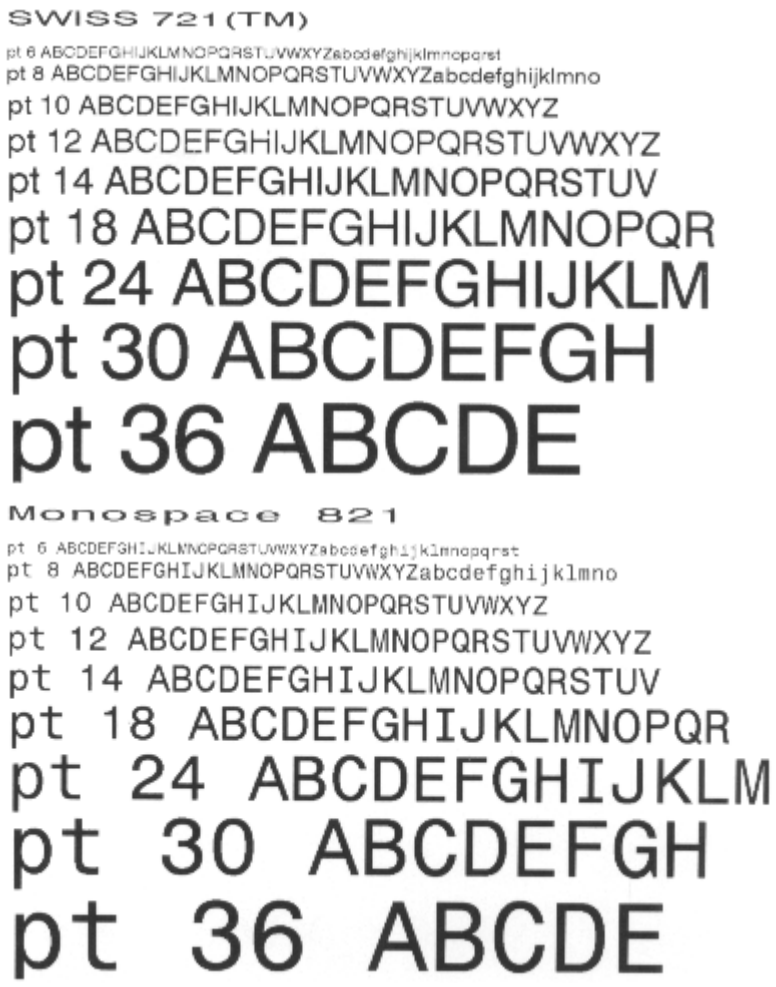
FONT 5
SWISS 721 BOLD(TM)
!@#$% ^ &*()_+|-=\<>?/[]';":{}
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqurstuvwxyz
0123456789
□š□¶□·□□ÒÓÔØ×Þ□□□''â™ãêëY™
š½œ¾PŸµÕàéŸ¥¦ ¨ª«¬-®¯°±23
´µ¶· ¹º»¼½¾¿ÀÁÂÃĀÅÆÇÈÉÊËÌÍ
ÎÏÐÑÒÓÔÕÖ×ØÙÚÛÜÝÞßàáâãäåæç
èéêëìíîïðñòóôõö÷øùúûüýþÿ

FONT 596
MONOSPACE 821
!@#$%^&*()_+|-=\<>?/[]';":{}
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqurstuvwxyz
0123456789
□š□¶□·□□ÒÓÔØ×Þ□□□''â™ãêëY™
š½œ¾PŸµÕàéŸ¥¦ ¨ª«¬-®¯°±23
´µ¶· ¹º»¼½¾¿ÀÁÂÃĀÅÆÇÈÉÊËÌÍ
ÎÏÐÑÒÓÔÕÖ×ØÙÚÛÜÝÞßàáâãäåæç
èéêëìíîïðñòóôõö÷øùúûüýþÿ

T - Text Field Definition

Internal Fonts

The following shows two of Apollo's scaleable fonts in point sizes from 6 to 36.



T - Text Field Definition

Examples:

This example shows four different bitmapped fonts with text rotated in four different orientations.

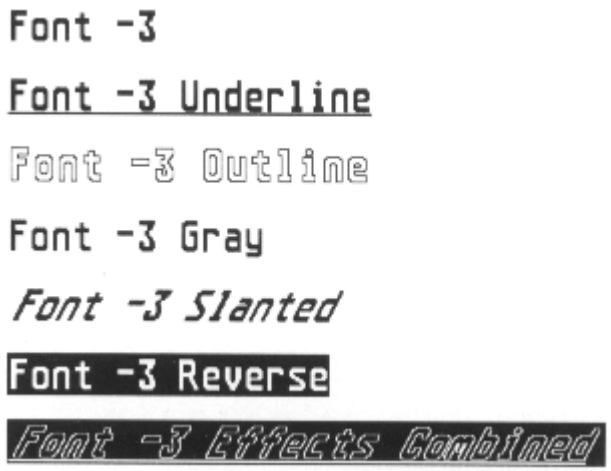
```
m i
J
H 5.2,0,T,R0
O R
S I1;.0,.0,2.5,2.6,4.26
T:PEAR;.45,.50,.0,-3,x6,y6;PEAR
T:BANANA;.22,2.43,90.0,-2,x6,y6;BANANA
T:ORANGE;1.98,.05,270.0,596,.60,h.4;ORANGE
T:APPLE;1.73,2.3,180.0,-1,x6,y6;APPLE
A 1
```

T - Text Field Definition

Examples:

The following example demonstrates some of the special effects available with the Apollo's internal bitmapped fonts:

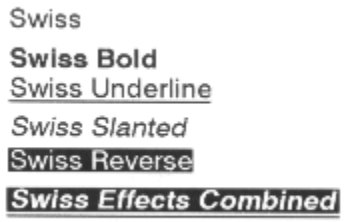
```
m i
J
O R
S l1;.0,.0,2.5,2.5,4.26
T:F0O5;0.5,0.20,0.0,-3,x2,y2;Font -3
T:F0O4;0.5,0.55,0.0,-3,x2,y2,u;Font -3 Underline
T:F0O1;0.5,0.90,0.0,-3,x2,y2,o;Font -3 Outline
T:F0O3;0.5,1.25,0.0,-3,x2,y2,g;Font -3 Gray
T:F0O2;0.5,1.60,0.0,-3,x2,y2,s;Font -3 Slanted
T:F0O7;0.5,1.95,0.0,-3,x2,y2,n;Font -3 Reverse
T:F0O6;0.5,2.30,0.0,-3,x2,y2,s,u,o,n;Font -3 Effects Combined
A 1
```



T - Text Field Definition

The following demonstrates some of the special effects available with the Apollo's internal scaleable fonts.

```
m i
J
O R
S l1;.0,.0,2.5,2.5,4.26
T:F0O1;.50,0.30,.0,3,PT 10;Swiss
T:F0O2;.50,0.50,.0,5,PT 10;Swiss Bold
T:F0O3;.50,0.65,.0,3,PT 10,u;Swiss Underline
T:F0O4;.50,0.85,.0,3,PT 10,s;Swiss Slanted
T:F0O5;.50,1.02,.0,3,PT 10,n;Swiss Reverse
T:F0O6;.50,1.22,.0,5,PT 10,s,u,n;Swiss Effects Combined
A 1
```



X - Synchronous Peripheral Signal Settings

The **X** command provides for synchronous setting of signals to the peripheral connector on the front of the printer. These signals may be used to switch simple mechanical devices or enable bar code verifiers. This is similar to the Immediate Command “x”, which sets signals, with the addition of control over exactly when the signal is set, which is the synchronous component of this command. (see x command)

Note: The list of positions (all signal settings) is cleared when starting a new job.

Usage:

X y[;ao] <CR>

Where:

X = Synchronous Peripheral Signal Setting Command

y = Coordinate of printing when signal change should be applied. Specify as the distance from start of print, in inches or millimeters, for example as .5 inches after start of print on the label.

ao= This is a combination of two values. The a-value is an AND-mask and the o-value is an OR-mask, both being a hex nibble, written together as a hex byte. These values will be used to set or reset the peripheral signal. If the ao operand is omitted entirely, the item is cleared from the internal list.

Example:

X .5;E0 <CR>	Clears bit 0 when the printhead reaches .5 inch from the top of the label.

Special Content Fields

Special Content Fields provide predefined values which are used to insert data into label format commands. They also provide special data manipulation functions useful in label format commands. When Special Content Fields are coded, each field is enclosed in brackets, [ ]. The brackets enclosing a name listed in the table below will not be printed on the label. If brackets enclose a value that is not a valid name from the list below, the brackets, [ ], will be printed on the label.

In this section only, optional parameters are shown enclosed in { } brackets.

Comprehensive examples showing many of the following operands are shown in the Appendix B Samples.

[SER:start{incr,{freq}}]
Insert Serial Number field. **start** represents the initial value of the incrementing field.
The **incr** field is the amount added to the start value. The **freq** field is the number of labels to print before each increment. The fields **incr** and **freq** default to 1 if no value is specified. Serial Number fields do not work with operator prompted fields.

Example: [SER:1000,1,2]
Start with a value of 1000. Print two labels with each value before incrementing by 1.

[DBF:keyfield,keyvalue,entryfield]
A special text to be used in text and barcodes to access data from a database. The **keyfield** field is the name of the field in a database which is checked for the value of the **keyvalue** field. If this is located, it reads the value of **entryfield** and uses this as the text or barcode data.

Example: [DBF:ZIP,ZIPCODE,CITY]
Search the database using the keyfield of ZIP, attempting to find and match a record with the value of the ZIPCODE field. Once this match is found, the value of CITY will be printed in the text or barcode field

[S:name]
Set the script style for numeric values. The **name** field can be either LATIN or ARABIC. Latin is the default without this parameter. The Arabic selection is valid only with font -3 and some special Arabic download fonts.

Example: [SER:1000,1] [S:ARABIC]
Print serial number in Arabic script style.

Special Content Fields

[?:{Headline},{Default},{Frequency}{,D}{,J}{,Lx}{,Mx}{,R}]
This is a special content field to place a prompt on the LCD panel for either keyboard or keypad entry.

Headline: The text appearing on the LCD display
Default: The default value assigned to this field
Frequency: The number of label(s) each entered value is to be used on before prompting for a new value. For example a value of '5' here will print 5 labels before prompting for a new value.
D: **D** deletes previously entered input from the display.
J: **J** will cause the prompt to repeat whenever the printer prompts for a new number of labels. This is especially useful when used with the "A [?,R]" command because it causes the printer to continuously loop on this prompt and the number to print. See example in Appendix B.
L: Length of input **L** is followed by a decimal value between 1 and 200.
M: The input mask **M** is followed by a value for each input digit. When the **M** is followed by a '!' blanks are not allowed as input. The following values are valid:

1 Numeric
2 Lower case letters
3 Alphanumeric Lower
4 Upper case letters
5 Alphanumeric Upper
6 Upper and Lower case letters
7 Alphanumeric Upper and Lower
8 All characters
0 Sign and decimal point

R: **R** will repeat the prompt if an error occurs such as if a record is not found in a database.

Example:
[?:Part Number,123456,,L6,M!111111]
Display "Part Number" on the LCD panel with a default value of 123456, entering 6 numeric digits with no blanks and a length of 6.

Example:
[?:Part Number,123456,,J,L6,M!111111]
A [?,R]
Display "Part Number" on the LCD panel with a default value of 123456, entering 6 numeric digits with no blanks and a length of 6. After printing the number of labels prompted to print, printer will re-prompt Part Number and the process will start over again. Incrementing numbers will not reset, but continue where they left off.

[C:fill**{,base}]**
Set replacement character fill for leading zero digits. Optional **base** is counting system for serialized fields (default: 10). Values for base that are supported include from base 2 through base 36. The character fill can be used with [DATE] and [ODATE] to allow a leading zero for single digit months. (Example January would be 01). See the date options in the Special Contents Field.

Example: [C:]
Replaces leading zeros with spaces. If number is 0010, it will print as 10.

Example: [C:0,4]
Using a serial number field starting with 000, and sending a quantity of 10 labels to the printer would print 000, 001, 002, 003, 010, 011, 012, 013, 020 and 021.

[DATE]
Prints actual date in the format of the defined country (see I
command). The [C:0] option can be used to have a leading zero for single digit months.

Example: [DATE]
Produces 10-15-1998 or Oct 15, 1998, if country is US.

[DAY02], [WEEK02], [MONTH02], [YY], [YYYY]
Note: [WEEK02] is supported by A-Series printers ONLY!
Prints numeric values for Day (01-31), Week (01-53), Month (01-12) and Year (00-99 or as 1998 . . .)

Example: [MONTH02][DAY02][YY]
Produces 101598 for October 15, 1998.

Example: [MONTH02] - [YYYY]
Produces 10-1998 for October 1998.

Special Content Fields

[WEEK], [DOFY], [WDAY]
Prints numeric values for week of the year (1-53), day of the year (1-366) and day of the week , weekday (1-7).

Example: [DOFY]
Produces 034 for February 3

Example: [WDAY]
Produces 4 for Thursday

[ODATE:+ddd{,+mm}{,+yy}]
Prints the date with an offset of dd-days, and/or mm-months, and/or yy-years (i.e. 'use before. . .'). The parameters are positional, and should be specified as 0 when not used to hold the position. The [C:0] can be used to have a leading zero for single digit months.

Example: [ODATE:+015]
Produces 10-20-1998 by adding 15 days, if today is Oct 5, 1998

Example: [ODATE:+0,+3][C:0]
Produces 01-05-1999 by adding 3 months if today is Oct 5, 1998.

[OWEEK:+ww]
Prints the week (01-52) with an offset. Used to calculate a ‘Use Before:’ or ‘Best By:’ date.

Example: [OWEEK:+4]
Produces 06 if today is Jan 10 (wk 02)

[wday], [wday2], [wday3]
Prints the weekday name, as a complete name, or as a 2 or 3 character abbreviation.

Example: [wday]
Produces Monday or Tuesday and so on

Example: [wday3]
Produces Mon or Tue and so on

[month], [mon]
Prints the name of the month or its 3 character abbreviation.

Example: [month]
Produces January or February and so on

Example: [mon]
Produces Jan or Feb and so on

Special Content Fields

[TIME]
Prints actual time in the format of the defined country (see 'I - Change Language' command).

Example: [TIME]
Produces 14:30:00 pm for 2:30 pm.

[H24], [H12], or [H024], [H012], [MIN], [SEC], [XM]
Prints actual time in 12 or 24 hour format, including minutes, seconds and am/pm as specified (using XM parameter).

Example: [H24] [MIN] [SEC]
Produces 153010 for 3:30:10 pm

Example: [H12]:[MIN]:[XM]
Produces 3:30:pm for 3:30 pm

Example: [H012]:[MIN]:[XM]
Produces 03:30:pm for 3:30 pm

[I]
Makes text field invisible. Most often used to hide results of calculations or to hide parts of a string that are concatenated for printing later.

[UPPER:x] and [LOWER:x]
Note: This command is supported by A-Series printers ONLY!
Convert the value of x to either uppercase or lowercase.

Example: [UPPER:uppercase text]
Produces: UPPERCASE TEXT

The value of x can also be the value from a [name] field.

Example: T:FIELD1;0,0,0,3,pt12;[month][I]
T 10,10,0,3,pt12;[UPPER:FIELD1]
Produces: AUGUST

Special Content Fields

[U:x]
Inserts Unicode character x into text or barcodes. x may be any of the following:

Decimal value
Hex value (indicated by a '$')
ASCII control code symbolic name
Code 128 control code name

Any standard Unicode characters may be inserted with this command. For a list of standard ASCII Control Codes and Code 128 Control Code names, see Appendix A.

Example: [U:13][U:10]
Produces characters needed for Carriage Return/Line Feed

Example: [U:$C]
Produces a Formfeed character.

Example: [U:CR]
Produces Carriage Return character.

Example: [U:FNC1]
Produces Code 128 Function 1 character. Use the following example to embed a Function 1 character between 123 and 456 in the Code 128 bar code symbology:

B:FUNCTION1;1.00,1.50,0,CODE128,.6,.015;123[U:FNC1]456
(See the Bar Code Field Definition Section for further examples of Code 128).

[name]
Inserts the contents of a previously defined field "name" into text (see the :name; parameter in Label Format commands **B** and **T**). With this field, specific text can be coded once and used multiple times on the same label. A common use would be to concatenate results of multiple calculations into one text field.

Example:
T:TESTFLD;0.20,0.30,0,596,PT 10;Apollo
T:TESTLINE;0.20,0.60,0,596,PT 14;Printer is *[TESTFLD]*

Produces the following, where contents of TESTFLD (Apollo) have been placed in the TESTLINE text string:

Apollo
Printer is *Apollo*

Special Content Fields

[name,m{,n}]
Inserts a substring from previously defined field "name" into text (see the :name; parameter in Label Format commands B and T). **m** represents the first character position to be copied. n is the number of characters to be copied, if not all remaining characters. With this field, parts of previously defined fields can be used again without recoding the values. Often used in connection with a serialized field, to obtain and use part of the number.

Example:
T:TESTFLD;0.20,0.30,0,596,PT 10;Apollo
T:TESTLINE;0.20,0.60,0,596,PT 14;Printer is *[TESTFLD,2,3]*

Produces the following, where contents of TESTFLD (Apollo), starting with second character for a total of 3 characters, have been placed in the TESTLINE text string:

Apollo
Printer is *pol*

[RTMP] or [RTMP:x]
Reads the value from the serial file and stores the value in the fieldname. In the example below, the value from the 'sample' file is stored in the XVALUE fieldname. The 'x' determines how many times this value will repeat.

[WTMP]
Writes the value specified to the serial file on PCMCIA memory card. In the example below, the value of XVALUE is written to the PCMCIA memory card after the field has incremented.

Example:
E TMP;sample
T:XVALUE;0,0,0,3,PT 16;[RTMP,2][I]
T:SERIAL;0,0,0,3,PT 16;[+:XVALUE,1][C:0][I][WTMP]
T:TESTFIELD;1.00,1.00,0,3,PT 16;Serial number is: [SERIAL]

[WLOG]
Writes data to a log file on a PCMCIA memory card. This log file can be used as an audit trail of labels that have been printed. In the example below, a transaction containing the incremented number along with the Date and Time, will be recorded on the 'sample' log file for each incremented value.

Example:
E LOG;sample
T:XVALUE;1.00,1.00,0,3,PT 16;[SER:0001]
T:TESTFIELD;0,0,0,3,PT 16;Label #[XVALUE] processed at [DATE] on [TIME].[WLOG][I]

Field Calculations and Comparisons

Text fields that have previously been assigned a name can be further manipulated using calculations, comparisons and concatenation. In most cases, the operations shown below can use a combination of previously defined fields and literal values for their operands.

Simple **calculations** may be done with field contents by placing arithmetic symbols before field names or values (see [:name;] in Label Format commands **B** and **T**). Operations available include: add, subtract, multiply, divide, and modulo operation. The figure resulting from the calculation will be inserted as the value of the field. For all calculations, a default of 2 decimal positions is used. The result of the calculation is placed in the text field defined by :name;.

The following arithmetic operations are available:

[+: oper1,oper2. . ,operx] Operand1 plus operand2.
[-: oper1,oper2] Operand1 minus operand2.
[*: oper1,oper2. . ,operx] Operand1 multiplied times operand2.
[/ : oper1,oper2] Operand1 divided by operand2.
[%: oper1,oper2] Operand1 minus operand2, remainder Modulo

Example: [*: weight, pricepound]
Multiplies weight by price per pound resulting in the item's total price, which is inserted where indicated by this field.

A **comparison** between field contents is accomplished by placing the compare symbol before the field names or values (see [:name;] in Label Format commands **B** and **T**). Comparisons available include: less than, equal to, greater than, logical-and and logical-or. The result of a comparison can be: 1=True, 0=False. The result of the comparison is placed in the text field defined by :name;.

[<: oper1,oper2] Is operand1 less than operand2?
[>: oper1,oper2] Is operand1 greater than operand2?
[=: oper1,oper2] Is operand1 equal to operand2?
[&: oper1,oper2] Logical And operand1 with operand2, all on?
[|: oper1,oper2] Logical Or operand1 with operand2, any on?

Example: [=:netwt,grosswt]
Compares field netweight to field grossweight, testing for an equal condition. Result is set to 1 if equal, 0 if not equal.

Field Calculations and Comparisons

The appearance of the result from a calculation may be additionally affected using the following options:

[D:m,n]

Set number of digits to print. **m** = digits (default is infinite) and **n** = fractional digits (default of 2).

Example: [/:10,3] [D:4,3]
Produces 3.333

[P:name,mn{o}]

Prints the result in a price format for field 'name'. **m** is the thousands separator, **n** is the character used as decimal point, **o** is an optional string inserted when the result has no fractional part to show after the decimal.

Example: $[P:price,.,-]
For 1000.00 prints $1.000, -- For 6543.21 prints $6.543,21

[R:x]

Rounding for math operations, where **x** is: **u=** rounding up, **d=** rounding down, **m=** round mathematically, to next, **n=** do not round (default)

Example: [R:d]
Round the result down.

[J:ml]

Sets justification within a fixed length string. **m** is the orientation of the string where: **l=**left**, c=**centered and **r=**right**. l** is the length of area for the string in inches or millimeters (see m command).

Example: [J:r3.00]APOLLO
Right justifies the text within a 3.00 inch area, producing:
APOLLO

Appendix A - Tables and Lists

UCC/EAN Application Identifiers

The three AIs in italics are not part of the American National Standard ANSI/UCC4—1995: UCC/EAN-128 Application Identifier Standard, June 1995 because these AIs had not been approved by the UCC at the time this standard was submitted to ANSI. This information will be incorporated in the ANS in future revisions of the standard.

AI	Content	Format
00	SSCC-18	n2+n18
01	SCC-14	n2+n14
02	*Item Number of Goods Contained Within Another Unit (Must Use with AI 37)*	*n14*
10	Batch or Lot Number	n2+an..20
11 (*)	Production Date (YYMMDD)	n2+n6
13 (*)	Packaging Date (YYMMDD)	n2+n6
15 (*)	Sell By Date (Quality) (YYMMDD)	n2+n6
17 (*)	Expiration Date (Safety) (YYMMDD)	n2+n6
20	Product Variant	n2+n2
21	Serial Number	n2+an..20
22	HIBCC - Quantity, Date, Batch and Link	n2+an..29
23 (**)	Lot Number (Transitional Use)	n3+n..19
240	Additional Product Identification assigned by the Manufacturer	n3+an..30
250	Secondary Serial Number	n3+an..30
30	Quantity	n2+n..8
310 (***)	Net Weight, Kilograms	n4+n6
311 (***)	Length or 1st Dimension, Meters	n4+n6
312 (***)	Width, Diameter or 2nd Dimension, Meters	n4+n6
313 (***)	Depth, Thickness, Height or 3rd Dimension, Meters	n4+n6
314 (***)	Area, Square Meters	n4+n6
315 (***)	Volume, Liters	n4+n6
316 (***)	Volume, Cubic Meters	n4+n6
320 (***)	Net Weight, Pounds	n4+n6
321 (***)	Length or 1st Dimension, Inches	n4+n6
322 (***)	Length or 1st Dimension, Feet	n4+n6
323 (***)	Length or 1st Dimension, Yards	n4+n6
324 (***)	Width, Diameter, or 2nd Dimension, Inches	n4+n6
325 (***)	Width, Diameter, or 2nd Dimension, Feet	n4+n6
326 (***)	Width, Diameter, or 2nd Dimension, Yards	n4+n6
327 (***)	Depth, Thickness, Height or 3rd Dimension, Inches	n4+n6
328 (***)	Depth, Thickness, Height or 3rd Dimension, Feet	n4+n6
329 (***)	Depth, Thickness, Height or 3rd Dimension, Yards	n4+n6
330 (***)	Gross Weight, Kilograms	n4+n6
331 (***)	Length or 1st Dimension, Meters, Logistics	n4+n6
332 (***)	Width, Diameter or 2nd Dimension, Meters, Logistics	n4+n6
333 (***)	Depth, Thickness, Height or 3rd Dimension, Meters, Logistics	n4+n6
334 (***)	Area, Square Meters, Logistics	n4+n6
335 (***)	Gross Volume, Liters	n4+n6
336 (***)	Gross Volume, Cubic Meters	n4+n6
340 (***)	Gross Weight, Pounds	n4+n6
341 (***)	Length or 1st Dimension, Inches, Logistics	n4+n6
342 (***)	Length or 1st Dimension, Feet, Logistics	n4+n6
343 (***)	Length or 1st Dimension, Yards, Logistics	n4+n6
344 (***)	Width, Diameter or 2nd Dimension, Inches, Logistics	n4+n6
345 (***)	Width, Diameter or 2nd Dimension, Feet, Logistics	n4+n6
346 (***)	Width, Diameter or 2nd Dimension, Yards, Logistics	n4+n6
347 (***)	Depth, Thickness, Height or 3rd Dimension, Inches, Logistics	n4+n6
348 (***)	Depth, Thickness, Height or 3rd Dimension, Feet, Logistics	n4+n6
349 (***)	Depth, Thickness, Height or 3rd Dimension, Yards, Logistics	n4+n6

UCC/EAN Application Identifiers (cont.)

350 (***)	Area, Square Inches	n4+n6
351 (***)	Area, Square Feet	n4+n6
352 (***)	Area, Square Yards	n4+n6
353 (***)	Area, Square Inches, Logistics	n4+n6
354 (***)	Area, Square Feet, Logistics	n4+n6
355 (***)	Area, Square Yards, Logistics	n4+n6
356 (***)	Net Weight, Troy Ounce	n4+n6
360 (***)	Volume, Quarts	n4+n6
361 (***)	Volume, Gallons	n4+n6
362 (***)	Gross Volume, Quarts	n4+n6
363 (***)	Gross Volume, Gallons	n4+n6
364 (***)	Volume, Cubic Inches	n4+n6
365 (***)	Volume, Cubic Feet	n4+n6
366 (***)	Volume, Cubic Yards	n4+n6
367 (***)	Gross Volume, Cubic Inches	n4+n6
368 (***)	Gross Volume, Cubic Feet	n4+n6
369 (***)	Gross Volume, Cubic Yards	n4+n6
37	*Quantity of Units Contained (For Use With AI 02 Only)*	*n..8*
400	Customer's Purchase Order Number	n3+an..30
410	Ship To (Deliver To) Location Code Using EAN-13	n3+n13
411	Bill To (Invoice To) Location Code Using EAN-13	n3+n13
412	Purchase From (Location Code of Party from Whom Goods are Purchased)	n3+n13
414	EAN Location Code for Physical Identification	n3+n13
420	Ship To (Deliver To) Postal Code Within a Single Postal Authority	n3+an..9
421	Ship To (Deliver To) Postal Code With 3-Digit ISO Country Code Prefix	n3+n3+an..9
8001	Roll products - Width, Length, Core Diameter, Direction and Splices	n4+n14
8002	Electronic Serial Number for Cellular Mobile Telephones	n4+an..20
8003	UPC/EAN Number and Serial Number of Returnable Asset	n4+n14+an..16
8004	*UCC/EAN Serial Identification*	*an..30*
8005	*Identifies the Price Per Unit of Measure*	*n6*
8100	Coupon Extended Code - Number System Character and Offer	n4+n1+n5
8101	Coupon Extended Code - Number System Character, Offer, and End of Offer	n4+n1+n5+n4
8102	Coupon Extended Code - Number System Character preceded by zero	n4+n1+n1
90	Mutually Agreed, Between Trading Partners or FACT DIs	n2+an..30
91	Intra-Company (Internal)	n2+an..30
92	Intra-Company (Internal)	n2+an..30
93	Intra-Company (Internal)	n2+an..30
94	Intra-Company (Internal)	n2+an..30
95	Internal-Carriers	n2+an..30
96	Internal-Carriers	n2+an..30
97	Intra-Company (Internal)	n2+an..30
98	Intra-Company (Internal)	n2+an..30
99	Internal	n2+an..30

(*) : To indicate only year and month, DD must be filled with "00"
(**) : Plus one digit for length indication
(***) : Plus one digit for decimal point indication

Data Value Representation:

a	alphabetic characters	an3	3 alphanumeric characters, fixed length
n	numeric characters	a..3	up to 3 alphabetic characters
an	alphanumeric characters	n..3	up to 3 numeric characters
a3	3 alphabetic characters, fixed length	an..3	up to 3 alphanumeric
n3	3 numeric characters, fixed length		

Appendix A - Tables and Lists

[U:] Command - Common Control Codes

Decimal Value	Hexadecimal Value	ASCII Control Code Equivalent
0	0	NUL
1	1	SOH
2	2	STX
3	3	ETX
4	4	EOT
5	5	ENQ
6	6	ACK
7	7	BEL
8	8	BS
9	9	HT
10	A	LF
11	B	VT
12	C	FF
13	D	CR
14	E	SO
15	F	SI
16	10	DLE
17	11	DC1
18	12	DC2
19	13	DC3
20	14	DC4
21	15	NAK
22	16	SYN
23	17	ETB
24	18	CAN
25	19	EM
26	1A	SUB
27	1B	ESC
28	1C	FS
29	1D	GS
30	1E	RS
31	1F	US

Code 128 Control Code Names

FNC1	FNC2	FNC3	FNC4
CODEA	CODEB	CODEC	

Appendix B - Coding Examples

Text File Label Coding Example

The following sample code is in the form of a text file and can be created with any text editor. It produces the label shown on the following page. *This code is available on the sample diskette provided with this manual.*

```
; *** TEXTCODE.TXT ***
; Set unit of measurement of inches
m i
; Start label job
J
; Set print speed to 5.2, heat to 5, using thermal transfer printing and ribbon saver off
H 5.2,5,T,R0
; Set Print Options to rotated
O R
; Define labels as die-cut, label size as height of 6.59 and width of 4.0 with a .10 gap between
; each label
S l1;.0,.0,6.59,6.69,4.0
; Define the serial text field rotated 270 degrees using the Swiss 721 scalable font with a font
; height of .10 inches.
T:SERIAL      ;0.80,0.05,270,3,0.10,h0.10;SERIAL
; Define the description text field rotated 270 degrees using the Swiss 721 scalable font with a
; font height of .10 inches.
T:DESC        ;2.70,3.15,270,3,0.10,h0.10;DESCRIPTION
; Define the quantity text field rotated 270 degrees using the Swiss 721 scalable font with a
; font height of .10 inches.
T:QUANT2      ;2.70,0.05,270,3,0.10,h0.10;QUANTITY
; Define the supplier text field rotated 270 degrees using the Swiss 721 scalable font with a font
; height of .10 inches.
T:SUP2        ;1.60,0.05,270,3,0.10,h0.10;SUPPLIER
; Define the part number text field rotated 270 degrees using the Swiss 721 scalable font with a
; font height of .10 inches.
T:PART2       ;3.85,0.05,270,3,0.10,h0.10;PART NO.
; Define the Steering Column text field rotated 270 degrees using the Swiss 721 scalable font
; with a font height of .20 inches.
T:DESC2       ;2.60,4.10,270,3,0.20,h0.12;STEERING COLUMN
; Define the (S) text field rotated 270 degrees using the Swiss 721 scalable font with a font
; height of .10 inches.
T:SER2    ;0.69,0.15,270,3,0.10,h0.10;(S)
; Define the (V) text field rotated 270 degrees using the Swiss 721 scalable font with a font
; height of .10 inches.
T:V       ;1.48,0.15,270,3,0.10,h0.10;(V)
```

Continued on the next page

Appendix B - Coding Examples

Text File Label Coding Example

; Define the (Q) text field rotated 270 degrees using the Swiss 721 scalable font with a font
; height of .10 inches.
T:Q ;2.60,0.15,270,3,0.10,h0.10;(Q)
; Define the (P) text field rotated 270 degrees using the Swiss 721 scalable font with a font
; height of .10 inches.
T:P ;3.75,0.15,270,3,0.10,h0.10;(P)
; Define the company text field rotated 270 degrees using the Swiss 721 scalable font with a
; font height of .08 inches.
T:COMPANY;0.02,0.40,270,3,0.08,h0.08;THARO SYSTEMS, INC. BRUNSWICK, OH 44212 330-273-4408
; Define the 2983104E text field rotated 270 degrees using the Swiss 721 scalable font with a
; font height of .25 inches. Will be printed above the barcode as interpretation
T:PARTNO ;3.55,0.85,270,3,0.50,h0.25;2983104E
; Define the 144 text field rotated 270 degrees using the Swiss 721 scalable font with a font
; height of .25 inches. Will be printed above the barcode as interpretation
T:QUANT ;2.45,0.85,270,3,0.50,h0.25;144
; Define the 43563 text field rotated 270 degrees using the Swiss 721 scalable font with a font
; height of .20 inches. Will be printed above the barcode as interpretation
T:SUPPLIER ;1.50,0.85,270,3,0.25,h0.20;43563
; Define the serial text field rotated 270 degrees using the Swiss 721 scalable font with a font
; height of .20 inches. The field will be a serial field printing two labels with the same serial
; number before the number increments by 1. Will be printed below the barcode as
; Interpretation
T:SERIALNO ;0.68,0.85,270,3,0.25,h0.20;[SER:100000000,1,2]
; Define the serial number barcode field using Code 3 of 9 with no interpretation. The barcode
; will be .50 inches high and will be a serial field printing two labels with the same serial
; number before the number increments by 1.
B:S2 ;0.60,0.50,270,code39,0.50,0.013,3:1;[SER:S100000000,1,2]
; Define the vendor number barcode field using Code 3 of 9 with no interpretation with a
; barcode height of .50 inches.
B:V2 ;1.45,0.50,270,code39,0.50,0.013,3:1;V43563
; Define the quantity barcode field using Code 3 of 9 with no interpretation with a barcode
; height of .50 inches.
B:Q2 ;2.40,0.50,270,code39,0.50,0.013,3:1;Q144
; Define the part number barcode field using Code 3 of 9 with no interpretation with a barcode
; height of .50 inches.
B:PART ;3.40,0.50,270,code39,0.50,0.013,3:1;P2983104E
Continued on the next page

Appendix B - Coding Examples

Text File Label Coding Example

; Define a line with a length of 4.00 inches and a thickness of .02 inches
G 0.90,0.01,270;L:4.00,0.02
; Define a line with a length of 4.00 inches and a thickness of .02 inches
G 1.70,0.01,270;L:4.00,0.02
; Define a line with a length of 6.59 inches and a thickness of .02 inches
G 2.80,0.01,270;L:6.59,0.02
; Define a line with a length of 1.70 inches and a thickness of .02 inches
G 0.01,4.0, 0;L:1.70,0.02
; Define a line with a length of 1.10 inches and a thickness of .02 inches
G 1.70,3.1, 0;L:1.10,0.02
; Define a line with a length of 3.51 inches and a thickness of .02 inches
G 2.40,3.1, 270;L:3.51,0.02
; Print 1 label
A 1

Appendix B - Coding Examples

QBASIC Program Label Coding Example

The following QBASIC program shows how to download a graphic image with the "d" command, then use the graphic image with the 'l' command to create a label. The label printed from this example is on the next page. *This code is available on the sample diskette.*

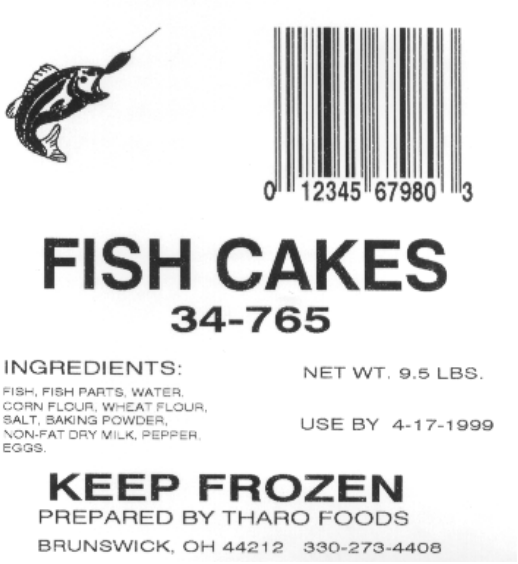
```
' *** BASICODE.BAS ***
ESC$ = CHR$(27)              'Define ESC, ASCII character 27 as
                             '  a string character.
DOT$ = CHR$(46)                    'Define period, ASCII character 46
                           '  as a string character.
CLS                      '  Clears the screen.
CLOSE                    '  Close any open files.
OPEN "COM1:9600,N,8,1,bin" FOR RANDOM AS #1
                         ' Open port 1 for communications at
                           '  9600 baud, no parity, 8 data bits,
                       '  1 stop bit, allow for full 8 bit data
                           '  transfer by use of the binary flag
                           '  (bin).  Communications are open
                          '  as random to allow DOS to buffer
                           '  character in both directions.
PRINT #1, "e PCX;*"                'Make space in printer's memory
PRINT #1, "r"                    'Reset values to default settings
PRINT #1, "d PCX;FISH"           'Places image header in printer memory
                           '  and names it FISH.
CLOSE #2                 '  Close channel 2
OPEN "A:FISH.PCX" FOR BINARY AS #2
                                 'Open the file "FISH.PCX" to be read in
                                 ' as file #2.  The use of BINARY stops
                                 ' DOS and BASIC from terminating input
                                 ' in the event the image file contains a Ctrl-Z
PRINT #1, ESC$; DOT$;            'ESC period, denotes the beginning of the
                                 '  graphic information.
'****** The following While statement filters the input, changing ESC to  ******'
'******   ESCESC as required by the Apollo for a graphic file download.  ******
WHILE NOT EOF(2)                 'While there is still data to read
      A$ = INPUT$(1, #2)     '  pass the data to A$
           IF A$ = ESC$ THEN PRINT #1, ESC$;       'If an ESC is encountered
                                 ' double the character.
      PRINT #1, A$;              ' Transfers data in A$ to printer.
WEND                             ' Stop when file is empty.
PRINT #1, ESC$; DOT$       ' ESC period, denotes the end of the graphic
                                 ' information.
'*** Print of DOWNLOADED GRAPHICS ***
PRINT #1, "m i"                  'Measure in inches
```

Continued on next page

Appendix B - Coding Examples

QBASIC Program Label Coding Example

```
PRINT #1, "J FISHFOOD"          'Begin fishfood label
PRINT #1, "H 5.2,0,T,R0"        'Print speed 5.2 in/sec., temp nominal
                                '  thermal transfer, ribbon saver off.
PRINT #1, "O R"                 'Reverse orientation.
PRINT #1, "Sl1;0,0,4.0,4.1,4.00;4x4 format"
                                'Use the see-through gap sensor with a
                                ' 4 by 4 inch label size.
PRINT #1, "I:PIC1;.20,.20,0,1,1;FISH"      'Print the image named FISH'
'*** The following commands format the text and bar code for the label, then print the label.***
PRINT #1, "T:TEXT1;0.45,2.00,0,5,0.50,h0.35;FISH CAKES"
PRINT #1, "T:TEXT2;1.35,2.29,0,5,0.26,h0.26;34-765"
PRINT #1, "T:TEXT3;0.20,2.58,0,3,0.13,h0.13;INGREDIENTS:"
PRINT #1, "T:TEXT4;0.20,2.73,0,3,0.08,h0.07;FISH, FISH PARTS, WATER,"
PRINT #1, "T:TEXT5;0.20,2.83,0,3,0.08,h0.07;CORN FLOUR, WHEAT FLOUR,"
PRINT #1, "T:TEXT6;0.20,2.92,0,3,0.08,h0.07;SALT, BAKING POWDER,"
PRINT #1, "T:TEXT7;0.20,3.02,0,3,0.08,h0.07;NON-FAT DRY MILK, PEPPER,"
PRINT #1, "T:TEXT8;0.20,3.11,0,3,0.08,h0.07;EGGS."
PRINT #1, "T:TEXT9;0.44,3.79,0,3,0.11,h0.11;BRUNSWICK, OH 44212   330-273-4408"
PRINT #1, "T:TEXT10;0.50,3.44,0,5,0.26,h0.26;KEEP FROZEN"
PRINT #1, "T:TEXT11;2.23,2.95,0,3,0.11,h0.11;USE BY [ODATE:+180]"
PRINT #1, "T:TEXT12;2.25,2.60,0,3,0.11,h0.11;NET WT. 9.5 LBS."
PRINT #1, "T:TEXT13;0.44,3.60,0,3,0.13,h0.13;PREPARED BY THARO FOODS"
PRINT #1, "B:BAR;2.07,0.20,0,UPCA,SC3;01234567980"
PRINT #1, "A 3"
CLOSE
END
```



Appendix B - Coding Examples

ESC Command Demonstration - QBASIC Program

The following code is available on the sample diskette provided with this manual.

```
'*** ESCDEMO.BAS ***
'*** Demonstrate ESC commands ***
' The following program demonstrates two of the ESC commands, the free memory command
' (ESC?)  and the printer status command (ESCs).  Some ESC commands can be coded in text    '
files.  But, those receiving a return response should be coded in a program, with code 'to accept    '
the response and pause so it can be viewed or tested.

DECLARE SUB LOOK ()                    'Declaration of subroutine
ESC$ = CHR$(27)                'Defines ESC, ASCII character 27
                          '  as a string character.
QST$ = CHR$(63)                        'Defines question mark (?), ASCII
                          '  char 63 as a string character.
LTRS$ = CHR$(115)                      'Defines letter s, ASCII character
                          '  115 as string character.

CLS : CLOSE                            'Clear screen & Close all open files

OPEN "COM1:9600,N,8,1" FOR RANDOM AS #1
          'Open port 1 at 9600 baud, no parity, 8 data bits, 1 stop bit.

PRINT "***Request free memory percent***"
PRINT #1, ESC$; QST$;          'ESC? requests free memory percent download information.
CALL LOOK                              'Pause to see response

PRINT "***Request printer status***"
PRINT #1, ESC$; LTRS$;         'ESCs requests printer status
CALL LOOK                              'Pause to see response

END

SUB LOOK
  PRINT "VIEW RESPONSE, THEN PRESS ANY KEY TO STOP LOOKING"
    DO
      IF LOC(1) > 0 THEN                       'If the file pointer for
        a$ = INPUT$(1, #1)           '  file #1 contains a number
        PRINT a$;                    '  read one byte at a time...
      END IF                         '  and print it on the screen.
    LOOP WHILE INKEY$ = ""           'Otherwise, loop until a
END SUB                                '  key is pressed.
```

Appendix B - Coding Examples

Memory Card - QBASIC Programs and Text Files - Summary

The following pages contain a series of examples demonstrating commands associated with PCMCIA Memory Card use. *These coding examples, along with all others included in this Appendix, are available on the sample diskette shipped with this manual.*

Using a memory card provides a way to permanently store fonts, images, databases and label format commands at the printer level rather than on a PC hard drive or diskette. This provides the means for the printer to function independently of a PC, by recalling formats using the front panel instead of commands sent to the printer from a PC. See the description of Front Panel Access on the following page.

The examples on the following pages include:

1) Format, Display Info/Directory - This program is useful before, during and after other programs access the memory card. It contains:
   - Command to initially format a memory card. This is commented out, as it is usually executed only once.
   - Command to delete unwanted files from memory card. Commented out - use only when needed.
   - Command (q) to request info about the card.
   - Command to request list of contents (directory) of card.

2) Store Font - This program stores the desired font to the memory card. This font will be referenced by a label format.

3) Store Image - This program stores an image (i.e. logo or symbol) to the memory card. This image will be referenced by a label format.

4) Store Database - This program stores a dBase database to the memory card. This database will be accessed by a label format.

5) Store Label - This text file stores the label format commands on the card. It can be retrieved later with a Load command, or through the front panel.

6) Store Label 2 - Store an additional label on the card.

7) Load Label Formats and Print - This text file demonstrates one method for retrieving labels stored on the card.

8) Store Label - Store a label on the memory card that accesses the database for variable input.

9) Load Label Formats and Print from Database - This text file demonstrates one method for retrieving labels stored on the card.

10) Store Label - Store a label on the memory card that has operator prompt fields for user input.

11) Load Label Formats with Operator Prompt Fields - This text file demonstrates one method for retrieving labels stored on the card.

Appendix B - Coding Examples

Memory Card - Front Panel Access

A label command set that has previously been stored on a memory card can be accessed or recalled using the following process:

1) Press ONL to take the printer Off-line.

2) Press CAN to display start of Label List. The front panel display shows:
Label from card
16-Char Label Description

3) Using CAN to move Up and FF to move Down, move through the list until the desired label is found. Press PSE for Enter to select the label.

4) If the label command set includes number of labels to print, (the A command), labels will begin printing automatically.

5) If the label command set does not include number of labels to print, it will be requested. The front panel display will show:
Number of labels
00001

6) Select the number of labels to print as follows:

- For the blinking digit (the left-most digit), use the FF for Down and CAN for Up to adjust the number to the desired value, then press PSE for Enter.
- The next digit will then blink. Repeat the FF or CAN then PSE process to select the digit's value.
- Select the desired value for each digit from the left-most (ten thousand) to the right-most (ones), using the same process.
- When the value for the right-most digit has been selected, the printer will begin printing the number of labels selected.
- **Note**: If an error is made, you cannot go back and correct it. Instead, you must cancel the entire selection by pressing ONL to put the printer on-line, then start from step 1 again.

7) When labels are finished printing, the printer is returned to its' normal on-line state.

Note: At any point in selecting a label, the entire process may be canceled by pressing ONL to place the printer back On-line. This is the only way to correct an error in selection.

Appendix B - Coding Examples

Memory Card - Format, Display Info/Directory - QBASIC Program

```
'*** MEMDIR.BAS ***
'***Memory Card commands - Format Card and/or Display Info and Directory***

DECLARE SUB LOOK ()         ' Declaration of subroutine
CLS : CLOSE                 ' Clear screen & Close all open files

OPEN "COM1:9600,N,8,1" FOR RANDOM AS #1   'Open port 1 at 9600 baud, no parity,
                            '8 data bits, 1 stop bit.

'---> ---> ****** Comment out FORMAT for most runs ****** <--- <---
'PRINT "***Formatting Memory Card***"
'PRINT #1, "M f;apollo"          ' Format card as apollo
'CALL LOOK                      ' Pause

'---> ---> ****** Comment out DELETE for most runs ****** <--- <---
'PRINT "***Deleting specified file***"
'PRINT #1, "M d LBL;GRAPHICS"          'Delete old file
'CALL LOOK                      ' Pause

PRINT "***Query Memory Card Info***"
PRINT #1, "q m"                 ' Query memory card info
CALL LOOK                       ' Pause to see response

PRINT "***Directory of Memory Card***"
PRINT #1, "M c"                 ' Request directory
CALL LOOK                       ' Pause to see response

END

SUB LOOK
  PRINT "VIEW RESPONSE, THEN PRESS ANY KEY TO STOP LOOKING"
    DO
      IF LOC(1) > 0 THEN        ' If the file pointer for
        a$ = INPUT$(1, #1)      ' file #1 contains a number
        PRINT a$;               ' read one byte at a time
      END IF                    ' and print it on the screen.
    LOOP WHILE INKEY$ = ""      ' Otherwise, loop until a
END SUB                         ' key is pressed.
```

Appendix B - Coding Examples

Memory Card - Store Font - QBASIC Program

```
'*** MEMFONT.BAS ***
'***Store Font on Memory Card***
'The following program downloads a font from the Windows directory to
'the Apollo's internal memory, then stores the downloaded font on the
'optional Memory Card currently inserted in the printer's PCMCIA slot.

ESC$ = CHR$(27)       ' Defines ESC, ASCII character 27 as
                      ' a string character.
DOT$ = CHR$(46)       ' Defines period, ASCII character 46
                      ' as a string character.
CLS                   ' Clear the screen.
CLOSE                 ' Close any open files.
OPEN "COM1:9600,N,8,1,bin" FOR RANDOM AS #1
                      ' Open port 1 for communications at
                      ' 9600 baud, no parity, 8 data bits,
                      ' 1 stop bit, allow for full 8 bit
                      ' data transfer by use of the binary
                      ' flag (bin). Communications are open
                      ' as random to allow DOS to buffer
                      ' characters in both directions.
CLOSE #2              ' Close channel 2
PRINT #1, "d TTF;ARIAL[SAVE]"    'Initiate download to Apollo internal
                      ' memory, a TrueType Font named Arial.
                      ' The [SAVE] option will copy the font
                      ' to the memory card as well.
OPEN "A:\ARIAL.TTF" FOR BINARY AS #2   'Open the file "ARIAL.TTF" to be read in
                      ' as file #2. The use of BINARY stops DOS and BASIC from
                      ' terminating input in the event that the file contains a Ctrl-Z
PRINT #1, ESC$; DOT$;          'ESC period, denotes the beginning of the
                      ' download information.
WHILE NOT EOF(2)      ' While there is still data to read
 A$ = INPUT$(1, #2)   ' pass the data to A$
  IF A$ = ESC$ THEN PRINT #1, ESC$; 'If an ESC is encountered
                      ' double the character.
 PRINT #1, A$;        ' Transfers data in A$ to printer.
WEND                  ' Stop when file is empty.
PRINT #1, ESC$; DOT$          'ESC period, denotes the end of the download information.
CLOSE
END
```

Appendix B - Coding Examples

Memory Card - Store Image - QBASIC Program

```
'*** MEMIMAGE.BAS ***
'***Store Image on Memory Card***
'The following program downloads an image file from a diskette to
'the Apollo's internal  memory, then stores the downloaded image on the
'optional Memory Card currently inserted in the printer's PCMCIA slot.

ESC$ = CHR$(27)       '  Defines ESC, ASCII character 27 as
                      '  a string character.
DOT$ = CHR$(46)       '  Defines period, ASCII character 46
                      '  as a string character.
CLS                   '  Clear the screen.
CLOSE                 '  Close any open files.
OPEN "COM1:9600,N,8,1,bin" FOR RANDOM AS #1
                      'Open port 1 for communications at
                      '  9600 baud, no parity, 8 data bits,
                      '  1 stop bit, allow for full 8 bit
                      '  data transfer by use of the binary
                      '  flag (bin).  Communications are open
                      '  as random to allow DOS to buffer
                      '  characters in both directions.
CLOSE #2              'Close channel 2
PRINT #1, "e PCX;*"    'Make space in printer's memory
PRINT #1, "r"         'Reset values to default settings
PRINT #1, "d PCX;FIRE[SAVE]"       'Initiate download to Apollo internal
                      '  memory, a PCX Image named Fire
OPEN "A:\FIRE.PCX" FOR BINARY AS #2
                      'Open the file "FIRE.PCX" to be read in
                      '  as file #2.  The use of BINARY stops
                      '  DOS and BASIC from terminating input
                      '  in the event that the file contains
                      '  a Ctrl-Z
PRINT #1, ESC$; DOT$;          'ESC period, denotes the beginning of the
                      '  download information.
WHILE NOT EOF(2)              'While there is still data to read
  A$ = INPUT$(1, #2)          '  pass the data to A$
    IF A$ = ESC$ THEN PRINT #1, ESC$;  'If an ESC is encountered
                              '  double the character.
  PRINT #1, A$;               'Transfers data in A$ to printer.
WEND                          'Stop when file is empty.
PRINT #1, ESC$; DOT$          'ESC period, denotes the end of the
                      '  download information.
CLOSE
END
```

Appendix B - Coding Examples

Memory Card - Store Label - Database File'

```
***MEMDBF.BAS***
'***Store Database on Memory Card***
'The following program downloads a dBASE database file from the sample
'diskette to the Apollo printer and saving a copy on the optional Memory
'Card currently inserted in the printer's PCMCIA slot.

ESC$ = CHR$(27)       '  Defines ESC, ASCII character 27 as
                      '  a string character.
DOT$ = CHR$(46)       '  Defines period, ASCII character 46
                      '  as a string character.
CLS                   'Clear the screen.
CLOSE                       'Close any open files.
OPEN "COM1:9600,N,8,1,bin" FOR RANDOM AS #1
                      'Open port 1 for communications at
                      '  9600 baud, no parity, 8 data bits,
                      '  1 stop bit, allow for full 8 bit
                      '  data transfer by use of the binary
                      '  flag (bin).  Communications are open
                      '  as random to allow DOS to buffer
                      '  characters in both directions.
CLOSE #2                    'Close channel 2
PRINT #1, "d DBF;UPC[SAVE]"       'Initiate download to Apollo internal
                      '   memory, a DBF Database named UPC. The
                        '  [SAVE] option will save a copy to the memory
                        '  card.
OPEN "A:\UPC.DBF" FOR BINARY AS #2
                      'Open the file "UPC.DBF" to be read in
                      '  as file #2.  The use of BINARY stops
                      '  DOS and BASIC from terminating input
                      '  in the event that the file contains
                      '  a Ctrl-Z
PRINT #1, ESC$; DOT$;         'ESC period, denotes the begining of the
                      '  download information.
WHILE NOT EOF(2)              'While there is still data to read
  A$ = INPUT$(1, #2)          '  pass the data to A$
    IF A$ = ESC$ THEN PRINT #1, ESC$;       'If an ESC is encountered
                          '  double the character.
  PRINT #1, A$;               'Transfers data in A$ to printer.
WEND                          'Stop when file is empty.
PRINT #1, ESC$; DOT$          'ESC period, denotes the end of the
                      '  download information.
CLOSE
END
```

Appendix B - Coding Examples

Memory Card - Store Label - Serial File'

```
***MEMTMP.BAS***
'***Store Serial File on Memory Card***
'The following program downloads an ASCII serial file from the sample
'diskette to the Apollo printer and saving a copy on the optional Memory
'Card currently inserted in the printer's PCMCIA slot.

ESC$ = CHR$(27)                 'Defines ESC, ASCII character 27 as
                        '  a string character.
DOT$ = CHR$(46)                 'Defines period, ASCII character 46
                        '  as a string character.
CLS                     'Clear the screen.
CLOSE                       'Close any open files.
OPEN "COM1:9600,N,8,1,bin" FOR RANDOM AS #1
                        'Open port 1 for communications at
                        '  9600 baud, no parity, 8 data bits,
                        '  1 stop bit, allow for full 8 bit
                        '  data transfer by use of the binary
                        '  flag (bin).  Communications are open
                        '  as random to allow DOS to buffer
                        '  characters in both directions.
CLOSE #2                    'Close channel 2
PRINT #1, "d TMP;SERIAL[SAVE]"      ' Initiate download to Apollo internal memory, an ASCII
                        ‘serial file named SERIAL. The [SAVE] option will save a copy to
                        ‘ the memory card.
OPEN "A:\SERIAL.TMP" FOR BINARY AS #2
                        'Open the file "SERIAL.TMP" to be read in
                        '  as file #2.  The use of BINARY stops
                        '  DOS and BASIC from terminating input
                        '  in the event that the file contains
                        '  a Ctrl-Z
PRINT #1, ESC$; DOT$;           'ESC period, denotes the beginning of the
                        '  download information.
WHILE NOT EOF(2)                'While there is still data to read
A$ = INPUT$(1, #2)          '  pass the data to A$
IF A$ = ESC$ THEN PRINT #1, ESC$;      'If an ESC is encountered double the character.
PRINT #1, A$;                   'Transfers data in A$ to printer.
WEND                        'Stop when file is empty.
PRINT #1, ESC$; DOT$            'ESC period, denotes the end of the
                        '  download information.
CLOSE
END
```

Appendix B - Coding Examples

Memory Card - Store Label - Text File

; ***MEMLABEL.TXT***
; The following label command set will be stored on the Memory Card.
;
; Store the format on the memory card as a LBL saved as 'FIREXTNG'
M s LBL;FIREXTNG
; Set the unit of measurement to inches
m i
; Start the label job and set the comment to 'Fire Extinguisher'. This will appear of the printers
; LCD as 'Fire Extinguisher'
J Fire Extinguisher
; Set print options to rotated
O R
; Define the label as die-cut with a height of 2.5 inches and a width of 4.26 inches
S l1;.0,.0,2.5,2.5,4.26
; Set the print speed to 5.2" per second, heat setting to 0, thermal transfer printing with ribbon
; saver off
H 5.2,0,T,R0
; Load the image FIRE from the memory card to the printer's memory
M l IMG;FIRE
; Load the font ARIAL from the memory card to the printer's memory
M l FNT;ARIAL
; Set the font number for ARIAL to 99
F 99;Arial
; Define the image FIRE with a magnification factor of 2 for horizontal and vertical
I:FIREPCX;.10, .10,0, 2,2;FIRE
; Define the Kitchen text field using Arial with a point size of 15 to print bold
T:TEXT1; 2.70, .40,0, 99,pt 15,b;Kitchen
; Define the Fire text field using Arial with a point size of 15 to print bold
T:TEXT2; 2.70, .70,0, 99,pt 15,b;Fire
; Define the Extinguisher text field using Arial with a point size of 15 to print bold
T:TEXT3; 2.70,1.00,0, 99,pt 15,b;Extinguisher
; Define the barcode field using UPC A. Based on UPC/EAN recommendations, the height is
; 80% of the width
B:UPCA; 2.70,1.30,0, UPCA,SC0;01234567890
; Define the Recharge after: text field using Arial with a point size of 12.
T:TEXT4; .30,1.50,0, 99,pt 12;Recharge after:
; Define the Date text field using Arial with a point size of 12 adding 2 years to today's date
T:TEXT5; .50,1.80,0, 99,pt 12;[ODATE:+0,+0,+2]
; Define the company text field using Arial with a point size of 11.
T:TEXT6; .30,2.10,0, 99,pt 11;Eveready Kitchen Helpers, Inc.
; Save the format to the memory card
M s LBL
; Define the amount of labels as a prompted field to be filled at time of printing.
A [NO]

Appendix B - Coding Examples

Memory Card - Store Label 2 - Text File

```
; ***MEMLBL2.TXT***
; ***Store Label on Memory Card - Example 2***
; The following label command set will be stored on the Memory Card
;
;  Save format to memory card as a LBL call GRAPHICS
M s LBL;GRAPHICS
; Set the unit of measurement to inches
m i
; Start the label job and set the comment to 'Graphic Samples'
J Graphic Samples
; Set print options to rotated
O R
; Define labels as die-cut, 2.5 inches in height and 4.26 inches in width
S l1;.0,.0,2.5,2.5,4.26
; Set print speed to 4 inches per second, heat setting to 0, thermal transfer printing and ribbon
; saver off.
H 4,0,T,R0
; Define a rectangle with a horizontal length of .75 inches, a vertical length of .80 inches and
; width of horizontal and vertical lines of .40 inches.  The fill option with place a grid pattern in
; the rectangle.
G :square1;0.50,0.75,0;R:0.75,0.80,0.40,0.40[F:grid][O]
; Define a circle with a first radius of  .37 inches, a second radius of  .37 inches and a width of
; .37 inches.  Using the fill option and selecting a shading of 38%, this will make a complete
; circle with the center shaded at 38%.
G :circle1;2.20,1.22,0;C:0.37,0.37,0.37[F:38%][O]
; Define a rectangle with a horizontal length of .75 inches, a vertical length of .80 inches and
; width of horizontal and vertical lines of .40 inches.  The fill option with place a single line grid
; pattern in the rectangle with all lines originating from the left side.
G :square2;3.20,0.75,0;R:0.75,0.80,0.40,0.40[F:left][O]
; Save the number of labels to be printed with a predetermined amount of 2.
A [NO] 2
; Save the format to the memory card.
M s LBL
```

Appendix B - Coding Examples

Memory Card - Load Label Formats and Print - Text File

```
; *** MEMLOAD1.TXT ***
; ***Load Formats from Memory Card and Print Labels***
;
; Load the firextng label into the printer's memory
M l lbl;firextng
; Print 2 labels
A 2
; Load the graphics label into the printer's memory.  The quantity is not necessary since the
; amount of labels was already saved to the format on the memory card.
M l lbl;graphics
```

The following labels are produced from the example above:

Appendix B - Coding Examples

Memory Card - Print from Database - Text File

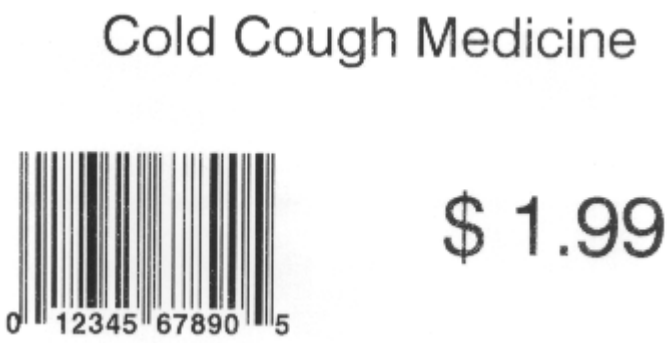
```
;*** DBLABEL.TXT ***
;*** Store label on Memory Card accessing the UPC.DBF database
; The following label will be stored on the Memory Card as a LBL file named DATABASE.
; The label uses the UPC.DBF database earlier stored on the Memory Card.
;
; Store the format on the memory card as a label called DATABASE
M s LBL;DATABASE
; Set the unit of measurement to inches
m i
; Start the label job
J
; Set the print options to rotated
O R
  ; Set the print speed to 5.2 inches per second, heat setting to 0, thermal transfer printing and the
  ; ribbon saver off.
H 5.2,0,T,R0
; Define the labels as die-cut, 2.5 inches in height, 4.26 inches in width
S l1;.0,.0,2.5,2.5,4.26
; Define the database file UPC for use on the memory card
E DBF;UPC
; Define a prompted field as “Enter UPC”, with a default value of 01234567890 and length of
; 11 characters.  This field is used for the barcode and the database search so the field is set to
; invisible. The “Enter UPC” will prompt for new input prior to each time the new Amount of
; Labels is requested.
T:UPCENTERED;0,0,0,3,pt20;[?:Enter UPC,01234567890,,J,L11][I]
; Define the barcode UPC A with a height of .9 inches and a narrow element of .013 inches.
; The UPC A barcode will print the input from the UPCENTERED line.
B:UPCA;0.5,1.0,0,UPCA,0.9,.013;[UPCENTERED]
; Define a text field to print the from the database.  Using the Special Content field DBF, a
; search will be performed on the database looking for a match between UPCENTERED and the
; search key from the database called UPCNUMBER.  Once a match has been found, the DESC
; field will print on the label.
T:DESC;0.9,0.5,0,3,pt20;[DBF:UPCNUMBER,UPCENTERED,DESC]
; Define a text field to print the from the database.  Using the Special Content field DBF, a
; search will be performed on the database looking for a match between UPCENTERED and the
; search key from the database called UPCNUMBER.  Once a match has been found, the PRICE
; field will print on the label.
T:PRICE;2.6,1.5,0,3,pt30;$ [DBF:UPCNUMBER,UPCENTERED,PRICE]
; Prompt continuously for Amount of labels.  Any prompted entry field marked for re-prompt, J,
; will continuously prompt also.
A [?,R]
; Store the label to the memory card
M s LBL
```

Appendix B - Coding Examples

Memory Card - Load Label Formats and Print from Database - Text File

```
; ***MEMLOAD2.TXT***
; ***Load Formats from Memory Card and Print Labels***
; This example loads the label DATABASE from the memory card, then
;   will ask the user, what UPC to enter.  If a keyboard is not attached, you can
;   scroll through the numbers for the upc number, or  use the default value.
;   The valid upc numbers for the database UPC.DBF are:
;     01234567890
;     09876543210
;     01111122222
;     02222233333
;
; Load the DATABASE label into the printer's memory.  You will then be prompted to enter a
;UPC Number.  Enter one of the valid UPC numbers listed above or accept the default value.
M l lbl;DATABASE
; Print 1 label
A 1
```

The following label is produced from the example above if the UPC number is 01234567890:



Appendix B - Coding Examples

Memory Card - Operator Prompt Example - Text File

;*** WPLABEL.TXT ***
;*** Store label on Memory Card with prompted fields for
; label information. It is best to use the optional keyboard with this example.
;
; Save the label to the memory card as WPLABEL
M s LBL;WPLABEL
; Set unit of measurement to inches
m i
; Start the label job
J
; Set print options to rotated
O R
; Define labels as die-cut, 2.5 inches in height, 4.26 inches in width
S l1;0,0,2.5,2.5,4.26
; Set print speed to 3.9 inches per second, heat setting to 0, thermal transfer printing
; and ribbon saver off.
H 3.9,0,T,R0
; Define a text field to accept user entry for an item number. The text will use the Swiss 721
; Bold typeface with a point size of 20 and will be centered on the label.
T:PART;0,0.5,0,5,pt20;Item Number: [?:Item Number:][J:c4.26]
; Prompt the user for Price per LBS and store the value in PPLBS. This field will not print on
; the label since the invisible option is used.
T:PPLBS;0,0,0,3,pt12;[?:Price per LBS:][I]
; Prompt the user for the weight and store the value in WEIGHT. This field will not print on
; the label since the invisible option is used.
T:WEIGHT;0,0,0,3,pt12;[?:Weight:,,1.0,][I]
; Define the PRICE field by multiplying the value of WEIGHT and PPLBS together. This field
; will not print on the label since the invisible option is used.
T:PRICE;0,0,0,3,pt12;[*:WEIGHT,PPLBS][I]
; Multiply the PRICE field by 100, set the number of digits to print as 5 and if the field is less
; than 5, use the fill option to place leading 0's in front of the amount. Store this value in the
; BCPRICE field. This field will not print on the label since the invisible option is used.
T:BCPRICE;0,0,0,3,pt12;[*:100,PRICE][D:5,0][C:0][I]
; Define a text field to print 'Fresh Deli Sandwich' using Swiss 721 typeface with a point size of
; 18. This field will be centered on the label.
T 0,1.0,0,3,pt18;Fresh Deli Sandwich[J:c4.26]
; Define a text field to print 'Packed' and today's date as February 19, 1997 using Swiss 721
; typeface with a point size of 10.
T 2.2,1.4,0,3,pt10;Packed: [month] [DAY], [YYYY]
; Define a text field to print 'Best Before' and today's date plus 10 days using Swiss 721
; typeface with a point size of 10.
T 2.2,1.6,0,3,pt10;Best before: [ODATE:10]

Continued on the next page

Appendix B - Coding Examples

Memory Card - Operator Prompt Example - Text File

; Define a text field to print 'Content' and the value of the WEIGHT field, followed by 'lbs'
; using Swiss 721 typeface with a point size of 10.
T 2.2,1.8,0,3,pt10;Content: [WEIGHT] lbs.
; Define a text field to print 'Price' and the value of the PPLBS field using the price format
; followed by 'per LBS'. The text field uses Swiss 721 typeface with a point size of 10.
T 2.2,2.0,0,3,pt10;Price: [P:PPLBS,,.] per LBS.
; Define a text field to print 'Price: $' and the value of the PRICE field using the price format.
; The text field uses Swiss 721 typeface with a point size of 14.
T 2.2,2.2,0,3,pt14;Price: $ [P:PRICE,,.]
; Define a barcode field UPC A to print '212345' and the value of the BCPRICE field. The
; barcode will be .8 inches in height and a narrow element of .013 inches.
B .5,1.5,0,UPCA,.8,.013;212345[BCPRICE]
; Save label to the memory card
M s LBL
; Define the amount of labels as a prompted field to be filled at time of printing.
A [NO]

Appendix B - Coding Examples

Memory Card - Load Label Format with Operator Prompt Fields - Text File

```
; ***MEMLOAD3.TXT***
; ***Load Formats from Memory Card and Print Labels***
; This example loads the label WPLABEL from the memory card, then
;   will display three prompts to the user and then print a label.  The example also
;   displays some of Apollo's Special Content Fields for calculation and User Prompt's
;
; Load the label into the printer's memory
M l LBL;WPLABEL
; Print 1 label
A 1
```

The following label is produced from the example above if the Item Number is 1526, the Price per Pound is 1.99, and the Weight is 2.5 pounds:

Appendix B - Coding Examples

Memory Card - Load Label and Replace Existing Data Text File

; *** REPLACE.TXT ***
; *** Replace text on previously stored format ***
; The following example will replace text on a previously stored
; format. Using the example label created from MEMLABEL.TXT, we
; change the text fields and the barcode field to new information.
; By using the R command it is unnecessary to resend all information to the memory card.
;
; Load the label FIREXTNG into the printer's memory
M l LBL;FIREXTNG
; Replace the UPCA field with 09876543210
R UPCA;09876543210
; Replace the TEXT1 field with Tharo
R TEXT1;Tharo
; Replace the TEXT2 field with Blue Tip
R TEXT2;Blue Tip
; Replace the TEXT3 field with Matches
R TEXT3;Matches
; Replace the TEXT4 field with Good until:
R TEXT4;Good until:
; Replace the TEXT5 field with today's date plus 3 years.
R TEXT5;[ODATE:+0,+0,+3]
; Replace the TEXT6 field with Tharo Systems, Inc.
R TEXT6;Tharo Systems, Inc.
; Print 2 labels.
A 2

The following label is produced from the example above:

Appendix B - Coding Examples

Memory Card - Incrementing Number from Operator Prompt - Text File

```
; *** SERIAL.TXT ***
; This example demonstrates incrementing an operator
; prompted line.  The format is saved to the PCMCIA memory card.
;
; Save the format to the memory card as serial.
M s LBL;serial
; Set the unit of measurement to inches.
m i
; Start the label job.
J
; Set print options to rotated and advance to tear-off.
O R,T
; Define the labels as die-cut, with a height of 2.5 inches and a width
; of 4.0 inches.
S l1;0,0,2.5,2.6,4.0
; Set print speed to 5.2 inches per second, heat setting to 0, thermal
; transfer printing with ribbon saver off.
H 5.2,0,T,R0
; Define a text field that is a title for our label.  The field will use
; the Swiss font and a point size of 10.  The field will also be centered.
T:title;0,1.00,0,3,pt10;Incrementing Serial Number from Operator Prompt[J:c4.0]
; Prompt the user for the starting number and store the value in the start
; field.  The default value will be 1000.  This field will not be printed
; on the label since the invisible option is used.
T:start;0,0,0,3,pt12;[?:Start Value?,1000][I]
; Define the offset field as an incrementing field. This field will not
; be printed on the label since the invisible option is used.
T:offset;0,0,0,3,pt12;[SER:0000][I]
; Define a text field that adds the values of start and offset together.
; This value will print a 4 digit number with leading 0's if the field is
; less than 4 digits.  The field will also be centered.
T:serial; 0.00,2.00,0,3,pt36;[+:start,offset][C:0][D:4,0][J:c4.0]
; Save label to the memory card
M s LBL
; Define the amount of labels as a prompted field to be filled in
; at time of printing.
A [NO]
```

Appendix B - Coding Examples

Memory Card - Increment Serial Number from Serial File - Text File

```
; *** SERFILE.TXT ***
; This example demonstrates reading a value from a serial file,
; incrementing the number and saving the value back to the serial file
; The format is saved to the PCMCIA memory card.
;
; Save the format to the memory card as serial.
M s LBL;serfile
; Set the unit of measurement to inches.
m i
; Start the label job.
J
; Set print options to rotated.  Advance label to tear-off bar.
O R,T
; Define the labels as die-cut, with a height of 2.5 inches and a width
; of 4.0 inches.
S l1;0,0,2.5,2.6,4.0
; Set print speed to 5.2 inches per second, heat setting to 0, thermal
; transfer printing with ribbon saver off.
H 5.2,0,T,R0
; Define the serial file.  The serial file name is SERIAL.
E TMP;SERIAL
; Define a text field that is a title for our label.  The field will use
; the Swiss font and a point size of 10.  The field will also be centered.
T:TITLE;0,1.00,0,3,pt10;Increment Number from Serial File[J:c4.0]
; Read the value from the serial file and store the value in the field
; READVALUE. This field will not print on the label using the [I] option.
T:READVALUE;0,0,0,3,pt12;[RTMP][I]
; This line has several things happening.  We will add 1 to READVALUE to increment
; the value stored in READVALUE.  This value will be stored in SNUMBER.  This value
; will also update the serial file with a format option of 10 spaces and no decimal
; places ([D:10,0]). This is necessary since the default is 2 decimal places. The
; field will not print on the label since we use the [I] option.
T:SNUMBER;0,0,0,3,pt12;[+:READVALUE,1][D:10,0][I][WTMP]
; Define the text field that will print the serial number.  The text field will use
; Swiss as the font and a point size of 14.
T:NUMBER;0.50,2.00,0,3,pt14;The Serial Number is: [SNUMBER]
; Save label to the memory card
M s LBL
; Define the amount of labels as a prompted field to be filled in
; at time of printing.
A [NO]
```

Appendix B - Coding Examples

RS485 Network Card

The program for checking status and printing a test label with Apollo printers and the RS485 network card is included on the program diskette. Due to the length and complexity of the program, it is not listed in the programming manual. See the actual program for documentation.

This page intentionally left blank

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

Z