Pyramid-ESC/POS Documentation

Release 2.0.0

Pyramid Technologies

May 20, 2022

Contents

Pyramid Technologies provides thermal printing solutions for self-service kiosk and custom OEM applications. We have printers that support 58mm to 80mm paper widths, multiple operating systems, and many standard protocols. Our dedicated team of engineers are ready to help make your project successful. Ask us about our Phoenix and Reliance thermal printers today.

This document provides a detailed list of all the ESC/POS commands that are supported by the PTI Reliance and Phoenix Thermal Printers. This document will provide descriptions, explanations, use cases, and examples of how to use the ESC/POS command protocol.

Please refer to our operations manuals for detailed cleaning and usage instructions:

Reliance - Reliance English Manual -

Phoenix - Phoenix English Manual

Tip: Thermal Talk ESC/POS API automates a lot of these features for you. Continue reading this document if your specific requirements are not yet covered by our ESC/POS API.

CHAPTER 1

ESC/POS Documentation for Pyramid Printers

Pyramid Technologies provides thermal printing solutions for self-service kiosk and custom OEM applications. We have printers that support 58mm to 80mm paper widths, multiple operating systems, and many standard protocols. Our dedicated team of engineers are ready to help make your project successful. Ask us about our Phoenix and Reliance thermal printers today.

This document provides a detailed list of all the ESC/POS commands that are supported by the PTI Reliance and Phoenix Thermal Printers. This document will provide descriptions, explanations, use cases, and examples of how to use the ESC/POS command protocol.

Please refer to our operations manuals for detailed cleaning and usage instructions:

Reliance - Reliance English Manual -

Phoenix - Phoenix English Manual

Tip: Thermal Talk ESC/POS API automates a lot of these features for you. Continue reading this document if your specific requirements are not yet covered by our ESC/POS API.

CHAPTER 2

Command Table Layout

Each command will be described in the format as shown below.

Name The name of the command

Format Hex, ASCII and Decimal values of the command

Range Range of acceptable values for a command and its parameters

Default A description of the command and what settings it affects

Notes Other notes about the command and how it operates

Related Lists any command that relates to the command

Example Displays and example of the command if it is available

Format Order

Commands are represented in hexadecimal, ASCII, and decimal, in that order. We use the following notation to indicate what form is being used.

Format `$Hex 1st`

`ASCII 2nd`

`Dec 3rd`

This section describes the terminology used in this document to describe the command and their functions.

Data Buffer Buffer for storing ESC/POS commands

Print Buffer Buffer for storing ticket image before printing

New Line A new line (line empty state) is a state that satisfies the following conditions:

1. There is currently no print data in the buffer
2. There is no skipped portion using the HT (horizontal tab) command
3. A print position has not been specified using *Absolute* or *Relative* print position commands

Dots vs Pixels vs Bytes In the context of Reliance printing, we prefer to speak in terms of dots. A dot is a literal dot on the thermal printer burn head. This is essentially the same concept as a pixel on your monitor. Each of our dots are 0.12499975mm (0.00492125") in diameter. This means we achieve a dot-density of 203.2, or more commonly stated as 203 DPI. Some of our commands are simplified to use bytes, or clusters of 8 dots. This reduces errors due to incorrect Two-byte calculations and provides slightly faster command parsing.

A good rule of thumb is that 1 byte == 8 dots == 1mm. Keep this in mind when you are printing images and dealing with margins. Another important number is 640. That is the absolute maximum number of dots we support on our absolute widest paper roll (80mm).

Two-byte Number Definitions Many ESC/POS commands use two-byte number definitions to represent large numbers in two data bytes. In order to represent numbers greater than 255 in this way, we perform an integer division and a modulo division to obtain the high and low bytes, respectively.

The common terms, nH and nL are used throughout this document and refer to the high and low bytes, respectively.

Note: ProTip: If the target value is less than 256, set nH to 0 and nL to the target value.

Example 1 To load the value 456 into two bytes, you first must solve for the quotient.

$nH = Quotient = \frac{456}{256} = 1$

Then solve for the modulo, where the value *b* is the result from above.

$nL = Modulo = 456 - (256 * (b)) = 200$

The resulting byte order transmitted to the printer would then be [01, 200] where transmission is from left to right.

Example 2 The *Left Margin* command requires a two-byte parameter for horizontal motion units. To get a left margin of 549 motion units

$nH = Quotient = \frac{549}{256} = 2$

$nL = Modulo = 549 - (2 * 256) = 37$

$\therefore nH = 2 and nL = 37$

$Verify: 37 + (2 * 256) = 549$

Example 3 To represent a negative number, use the identity

$(nL + (nH * 256)) = 65536 - (value)$.

If we needed to represent the value -324 we would do the following:

$65536 - 324 = 65212$

$nH = Quotient = \frac{65212}{256} = 254$

$nL = Modulo = 65212 - (254 * 256) = 188$

$\therefore nH = 254 and nL = 188$

$Verify: 65536 - (188 + (254 * 256)) = 324$

CHAPTER 3

Pseudo Command Syntax

Throughout this document, sample functions will be used to express actions such as writing data to the printer, calling print and viewing the results. These are not meant to represent low-level implementations but are simply abstractions for the purpose of providing examples.

Command	Description
write(data)	Writes the specified data to the printer. The data may be hex, ascii, mixed, etc.
print()	Request the printer to print its buffer
>>>	Display result or the printer's response

These examples will reside in code blocks with the important lines highlighted yellow.

Like This

```
write('pseudo write command send data to printer')
print()
>>> Some sort of response
```

CHAPTER 4

Printer Information

Commands that provide information about the printer's identity are provided here

4.1 Printer ID - `$1D $49`

Short description

Format `Hex $1D $49 n`

`ASCII GS I n`

`Decimal 29 73 n`

Notes

- This command responds when the data buffer is processed. Therefore, a time delay between when the command is received and when the printer responds can occur. This time delay depends on the data buffer status and printer status.
- Refer to this table for valid values of `n`:
- Phoenix only supports the `n=3` parameter.

n	Printer ID	Description
1,29	Printer Model ID	[Model, Reserved, Reserved]
2,50	Type ID	RESERVED (Reports $02)
3,31	Firmware Revision	4 character revision, e.g. “1.12”

Range Reliance: `1 n 3, 49 n 51` Phoenix: `n = 3`

Default `N/A`

Related `None`

Example Model

```
write('\x1d\x49\x01')
>>> $5D $95 $59              # Reliance model code followed by 2␣
↪reserved bytes
```

Example Firmware Revision (PHX and REL)

```
write('\x1d\x49\x03')
>>> $31 $2e $31 $32          # 1.12 in ASCII
```

4.2 Transmit Status - `$1D $72`

Transmits the paper sensor status based on the value of `n`.

Format `Hex $1D $72 n`

`ASCII GS r n`

`Decimal 29 114 n`

Notes

- This is **not** a real time status command.
- Commands will be processed in order of reception, therefore a time delay may be present between receiving the command transmitting the Paper Sensor Status.

- Refer to this table for response codes:

Ejector State Byte Table				
BIT	OFF/ON	HEX	DECIMAL	DESCRIPTION
0,1	Off	00	0	Paper Roll Present With Abundance
	On	03	3	Near Paper Roll End
2,3	Off	00	0	Paper Present
	On	0C	12	Paper Not Present
4				Reserved
5				Undefined
6				Undefined
7				Reserved

Range `n=1,49`

Default `N/A`

Related `None`

Example Paper Status

```
write('\x1d\x72\x01')
>>> \x03                # Roll is present but near end
```

4.3 Transmit paper sensor status - $1B $76

Transmits the status of paper sensor as 1 byte of data.

Format `Hex $1B $76`

`ASCII ESC v`

`Decimal 27 118`

Notes

- This is **not** a real time status command.
- Commands will be processed in order of reception, therefore a time delay may be present between receiving the command transmitting the Paper Sensor Status.
- Refer to this table for response codes:

Ejector State Byte Table				
BIT	OFF/ON	HEX	DECIMAL	DESCRIPTION
0,1	Off	00	0	Paper Roll Present With Abundance
	On	03	3	Near Paper Roll End
2,3	Off	00	0	Paper Present
	On	0C	12	Paper Not Present
4				Reserved
5				Undefined
6				Undefined
7				Reserved

Range `N/A`

Default `N/A`

Related `None`

Example `None`

CHAPTER 5

Font Controlling Commands

This section describes all commands that affect how and which font is rendered.

5.1 Initialize - `$1B $40`

Clears the data in the print buffer and resets the printer modes to the modes that were in effect when the power was turned on.

Format `Hex $1B $40`

`ASCII ESC @`

`Decimal 27 64`

Range `None`

Default `None`

Notes

- Print buffer is cleared
- Data buffer contents are preserved
- NV graphics (NV bit image) information is maintained.
- User NV memory data is maintained.

Related `None`

Example `None`

5.2 Select Print Mode - `$1B $21`

Quick-select a variety of print control options such as font type and effects.

Format `Hex $1B $21 n`

`ASCII ESC ! n`

`Decimal 27 33 n`

Range `0 n 255`

Default `None`

Notes

- See table for the appropriate value of n.
- The baseline for characters of different vertical scalars will be the same

Underline exceptions

- Does not underline 90°/270° rotation
- Does not underline horizontal tabs
- Underline thickness is specified by *Underline Mode*

This command resets the left and right margins

- Left margin set by *Left Margin*
- Right margin set by *Print Area Width*

For each of the underline, italic, bold modes:

- These can be issued by their respective ESC commands or this command
- The last received command is the effective command.

BIT	State	HEX	DECIMAL	Function
0	Disabled	00	0	Select character font A
	Enabled	01	1	Select character font B
1	–	–	–	Reserved
2	–	–	–	Reserved
3	Disabled	00	0	Disable emphasis (bold) mode
	Enabled	08	8	Enabled emphasis (bold) mode
4	Disabled	00	0	Disable double-height mode
	Enabled	10	16	Enable double-height mode
5	Disabled	00	0	Disable double-width mode
	Enabled	20	32	Enable double-width mode
6	Disabled	00	0	Disable italic mode
	Enabled	40	64	Enable italic mode
7	Disabled	00	0	Disable underline mode
	Enabled	80	128	Enable underline mode

Related `None`

Example

```
write("\x1b\x21\x01")          # Select Font B
write("\x1b\x21\x11")          # Select double-height mode
write("This is font B, double-height")
print()
>>> This is font B, double-height
write("\x1b\x21\x00")          # Select Font A and disable double-
↪height
write("This is font A")
print()
>>> This is font A"
```

5.3 Underline Mode - `$1B $2D`

Turns underline mode on or off, based on the following values of n:

- n = 0, 48 Turns off underline mode
- n = 1, 49 Turns on underline mode (1-dot thick)
- n = 2, 50 Turns on underline mode (2-dot thick)

Format `Hex $1B $2D n`

`ASCII ESC - n`

`Decimal 27 45 n`

Range `0 n 2, 48 n 50`

Default `0`

Notes

- Invalid `n` values will be ignored. The existing underline and underline thickness settings will be maintained.
- *90° Rotation* characters will not be underlined
- *Black/White Reverse* characters will not be underlined.
- Tab characters are not underlined when this mode is enabled.
- Disabled or enabling this mode takes is applied immediately. The following data will be underlined.
- Default underline thickness is 1 dot.
- Character size does not affect underline thickness.
- Thickness moves downward from the natural top of the character.
- *Select Print Mode* Can also be used for this setting. The last received command is the effective one.

Related `None`

Example

```
write("\x1b\x2d\x01")          # Enable underline
write("This is underlined")
print()
>>> __This text is underlined__
write("\x1b\x2d\x00")          # Disable underline
write("This is not underlined")
print()
>>> This is not underlined
```

5.4 Italics Mode - `$1B $34`

Turns *italics* mode on or off, based on the following values of n:

- n = 0, 48 Turns off *italics* mode
- n = 1, 49 Turns on *italics* mode

Format `Hex $1B $34 n`

`ASCII ESC 4 n`

`Decimal 27 52 n`

Range `0 n 1, 48 n 49`

Default `n=0, n is base 10`

Notes

- This effect is applied immediately
- *Select Print Mode* can also be used for these settings. The last received command is the effective one.

Related `None`

Example

```
write("\x1b\x34\x01")              # Enable italics
write("This is italic")
print()
>>> This is italic
write("\x1b\x34\x00")              # Disable italics
write("This is not italic")
print()
>>> This is not italic
```

5.5 Emphasis Mode - `$1B $45`

Turns **emphasis** mode on or off, based on the LSB of `n`:

- n = 0, Turns off *emphasis* mode
- n = 1, Turns on *emphasis* mode

Format `Hex $1B $45 n`

`ASCII ESC E n`

`Decimal 27 69 n`

Range `0  n  255`

Default `n=0, n is base 10`

Notes

- This effect is applied immediately
- Only the LSB of `n` is inspected
- *Select Print Mode* can also be used for this settings. The last received command is the effective one.

Related `None`

Example

```
write("\x1b\x45\x01")              # Enable emphasis
write("This is bold")
print()
>>> This is bold
write("\x1b\x45\x00")              # Disable itemphasisalics
write("This is not bold")
print()
>>> This is not bold
```

5.6 Select Character Font - `$1B $4D`

Selects character font based on n.

Set	n	DESCRIPTION
A	0,48	Select Font A
B	1,49	Select Font B

Format `Hex $1B $4D n`

`ASCII ESC M n`

`Decimal 27 77 n`

Range `n = 0, 1, 48, 49`

Default `n=0, n is base 10`

Notes

- **Toggles between Font A and Font B**
  - Font A (12w 24h)
  - Font B (9w 17h)
- **Scales the current font depending on *CPI Mode***
  - Mode 1: Font A width * 0.33, Font B width * 2
  - Mode 2: Font A width * 2 , Font B width * 4
  - Mode 3: Font A width * 4 , Font B width * 2

Related `None`

Example `None`

5.7 Select Font A - `$1B $50`

Selects Font A Stored in the printer.

Format `Hex $1B $50`

`ASCII ESC @`

`Decimal 27 80`

Range `None`

Default `None`

Notes

- Set character font to type A. (12w 24h)

Related `None`

Example `None`

5.8 Select Font C - `$1B $54`

Selects Font C Stored in the printer.

Format `Hex $1B $54`

`ASCII ESC P`

`Decimal 27 84`

Range `None`

Default `None`

Notes

- Set character font to type C. (24w 48h)

Related `None`

Example `None`

5.9 Select Font D - `$1B $55`

Selects Font D Stored in the printer.

Format `Hex $1B $55`

`ASCII ESC T`

`Decimal 27 85`

Range `None`

Default `None`

Notes

- Set character font to type D. (16w 24h)

Related `None`

Example `None`

5.10 90° Rotation - $1B $56

Turns 90° rotation on or off, based on n

- n = 0, 48 Turns off rotation
- n = 1, 49 Turns on rotation

Format `Hex $1B $56 n`

`ASCII ESC V n`

`Decimal 27 86 n`

Range `n = 0, 1, 48, 49`

Default `n=0, n is base 10`

Notes

- Invalid `n` values will be ignored
- *90° Rotation* characters will be underlined in the vertical direction.

Tip: *Double-width* and *Double-height* commands in *90° Rotation* will enlarge the opposite dimension when this mode is enabled. i.e. width and height scalars are swapped

---

Related `None`

Example `None`

5.11 Select Character Code Page - $1B $74

Select character code page based on `n`

Format `Hex $1B $74 n`

`ASCII ESC t n`

`Decimal 27 116 n`

Range `See n Table Below`

Default `n=0, n is base 10`

Notes

- Reserved values of `n` should not be used in your application
- Names with an asterisk (*) may require updated firmware

n	Font Code Page
0	**Default** USA ASCII + Cyrillic
2	Reserved
3	CP437 (Spanish)
4	Reserved
5	Reserved
17	CP808 (Cyrillic)
18	Georgian Mkhedruli*
19	Reserved
255	Reserved

Related `None`

Example `None`

5.12 Upside-down Mode - `$1B $7B`

Turn upside-down print mode on/off

- When the LSB of n is 0, upside-down print mode is turned off.
- When the LSB of n is 1, upside-down print mode is turned on.

Format `Hex $1B $7B n`

`ASCII ESC { n`

`Decimal 27 123 n`

Range `0  n  255`

Default `n=0, n is base 10`

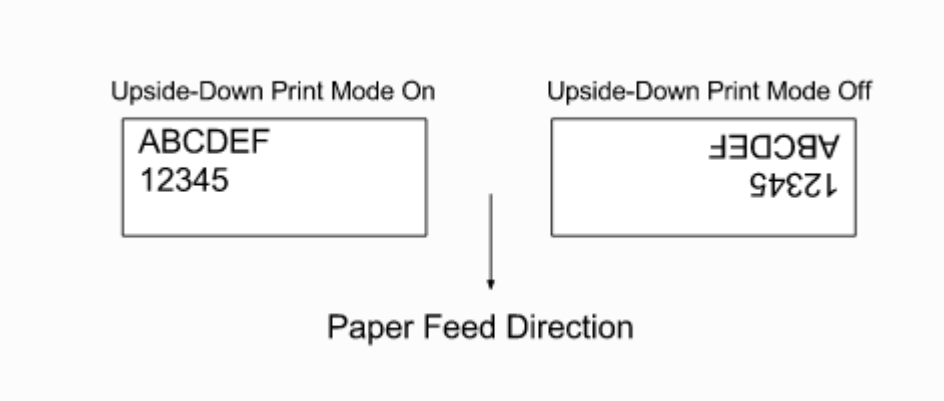
Notes

- This command is enabled only when processed at the beginning of the line.
- When upside-down print mode is turned on, the printer prints all characters rotated 180° from right to left.
- Upside-down print mode is effective for all data except for the following:
  - Raster bit image from *Raster Image*
- Upside-down print mode is effective until any of the following occur:
  - It is explicitly disabled by settings LSB of `n` to 0
  - *Initialize* is executed
  - Printer is reset
  - Power is turned off

Tip: The line printing order is not reversed. Therefore, care should be taken when considering the order of the data transmitted.

Related `None`

Example



5.13 Set CPI Mode - `$1B $C1`

Selects the active CPI mode.

n	CPI Mode	
0,48	Font A = 11 cpi	Font B = 15 cpi
1,49	Font A = 15 cpi	Font B = 20 cpi
2,50	Font A = 20 cpi	Font B = 15 cpi

Format `Hex $1B $C1 n`

`ASCII ESC Á n`

`Decimal 27 193 n`

Range `n= 0, 1, 2, 48, 49, 50`

Default `n=0, n is base 10`

Notes

- CPI is characters per inch
- The higher the CPI, the smaller the font

Related *Select Print Mode*

Example `None`

5.14 Select Codepage - `$1C $7D $26`

Used to select any installed codepage as the active codepage. Using Two Byte Number Definitions, send the integer number of the codepage. For example, if codepage 437 is desired, then send the integer 437.

Format `Hex $1C $7D $26 xL xH`

`ASCII FS } & xL xH`

`Decimal 28 125 36 xL xH`

Range `0  xL + (xH * 256)  65535`

Default Default codepage is set with PC tools

Notes

- Codepage = (xL + (xH * 256))
- If the codepage sent to the printer is not installed, the currently active codepage will not change.
- See *Two Byte Numbers* section for more information on two byte number definitions.

Related *Select Codepage*

Example Select Codapge 437

```
write('\x1c\x7d\x26\xb5\x01')   #  Select codepage 437
```

Example Select Codapge 1252

```
write('\x1c\x7d\x26\xe4\x04')   #  Select codepage 1252
```

5.15 Select Character Size - `$1D $21`

Select character width and height according to the bits of n.

- Bits 0 to 3 : select character height (see table 2)
- Bits 4 to 7 : select character width (see table 1)

Table 1 - Width

HEX	DECIMAL	Width
0	0	1 (normal)
10	16	2 (2x width)
20	32	3 (3x width)
30	48	4 (4x width)
40	64	5 (5x width)
50	80	6 (6x width)
60	96	7 (7x width)
70	112	8 (8x width)

Table 2 - Height

HEX	DECIMAL	Height
0	0	1 (normal)
1	1	2 (2x height)
2	2	3 (3x height)
3	3	4 (4x height)
4	4	5 (5x height)
5	5	6 (6x height)
6	6	7 (7x height)
7	7	8 (8x height)

Format `Hex $1D $21 n`

`ASCII GS ! n`

`Decimal 29 33 n`

Range `0 n 255`

Default `n=0, n is base 10`

Notes

- Invalid `n` values are ignored, the current character size is maintained.
- Characters on the same line sized to different heights will be aligned to the topline.
- Width is expanded to the right.
- In standard mode, the character is enlarged in the paper feed direction when double-height mode is selected, and it is enlarged perpendicular to the paper feed direction when double-width mode is selected. However, when character orientation changes in 90° clockwise rotation mode, the relationship between double-height and double-width is reversed.
- *Select Print Mode* Can also be used for this setting. The last received command is the effective one.

Related *Select Print Mode*

Example `None`

5.16 Reverse Print Mode - `$1D $42`

Turn white/black reverse printing (inverted) mode on/off based on the LSB of n - LSB Set: reverse enabled - LSB Clear: reverse disabled

Format `Hex $1D $42 n`

`ASCII GS B n`

`Decimal 29 66 n`

Range `0  n  255`

Default `n=0, n is base 10`

Notes

- Only the LSB of `n` is inspected.
- This does not affect images, barcodes, or user defined images.
- This has a higher priority than underline.
  - Underline will stay enabled but not be applied if this setting is enabled.

Related `None`

Example `None`

5.17 Select Double-strike mode - `$1B $47`

Turns double-strike mode on or off depending on the LSB of n. - LSB Set: double-strike mode is turned on - LSB Clear: double-strike mode is turned off

Format `Hex $1B $47 n`

`ASCII ESC G n`

`Decimal 27 71 n`

Range `0 n 255`

Default `n=0, n is base 10`

Notes

- Only the LSB of `n` is inspected.
- This does not affect images, barcodes, or user defined images.

Related `None`

Example `None`

CHAPTER 6

Cursor Position Commands

This section describes all commands that affect the location of the print position. Consider the print position as a movable pointer that allows you to print anywhere on the print ticket.

6.1 Horizontal Tab - $09

Advances the horizontal print position to the next column as specified by the Set Horizontal Tab Position command.

Format Hex $09

ASCII HT

Decimal 9

Notes

- If no tab position has been set, default columns of 8 characters will be used.
- If this command is received at the end of a line, the current line buffer will be printed and the tab will be applied to the next line
- 1 column width is equal to the width of the current font

Range None

Default 8 Columns

Related None

Example

```
write("Hello\x09World!")    # \x09 is the raw 0x09 character
print()
>>> Hello   World?
```

6.2 Line Feed - `$0A`

Prints the data in the print buffer and feeds one line based on the current line spacing.

Format `Hex $0A`

`ASCII LF`

`Decimal 10`

Notes

- Sets the print position to the beginning of the line.

Range `None`

Default `None`

Related *Carriage Return*

Example

```
write("Hello World!\x0A")
print()
>>> Hello World?
>>>
```

6.3 Form Feed - $0C

Prints the data in the print buffer, cuts the paper and presents the ticket.

Format Hex $0C

ASCII FF

Decimal 12

Notes

- Sets the print position to the beginning of the line.

Range None

Default None

Related *Total Cut*

Example

```
write("Hello World!\x0C")
print()
>>> Hello World?
# Paper is cut and presented, buffer is now empty and awaiting␣
↪more data
>>>
```

6.4 Carriage Return - `$0D`

If CR command is enabled, this command will function exactly like the command $0A does, otherwise, the command is ignored.

Format `Hex $0D`

`ASCII CR`

`Decimal 13`

Notes

- Sets the print position to the beginning of the line
- CR can be enabled or disabled with Reliance Tools

Range `None`

Default `None`

Related *Line Feed*

6.5 Cancel Current Line- `$18`

Deletes/Cancels the current line

Format `Hex $18`

`ASCII CAN`

`Decimal 24`

Notes

- Sets the print position to the beginning of the line

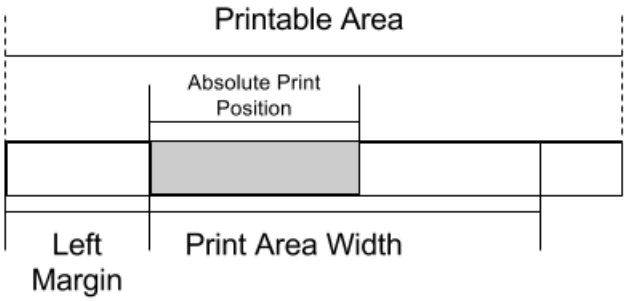
Range `None`

Example

```
write('Hello World')                # Put some text in the buffer
write('\x18')                       # Send cancel command
write('Thank you!')                 # Write some other data
write('\x1b\x69')                   # Force-print the buffer
print()
>>>Thank you!
```

6.6 Absolute Print Position - `$1B $24`

Moves the print position to [(nL + (nH × 256)) × (horizontal or vertical motion unit)] from the left edge of the print area. Uses Two Byte Number Definitions. See *Terminology Section*



Format `Hex $1B $24 nL nH`

`ASCII ESC $ nL nH`

`Decimal 27 36 nL nH`

Range `None`

Default `nL = 0, nH = 0`

Notes

- Settings that exceed the printable area are ignored.
- If settings exceed the print area width, the absolute print position is set, but no text will be able to fit in the print area width and any text will be treated as a line feed.
- When standard mode is selected, the horizontal (perpendicular to paper feed) motion unit is used.
- When this command is executed, the printer is no longer in a "New Line" state. See *Terminology Section*
- The horizontal and vertical motion units are specified by *Motion Units*. Changing the horizontal or vertical motion unit does not affect the current absolute print position.
- Absolute print position is effective until it is changed, a new line event occurs, *Initialize* is executed, the printer is reset, or the power is turned off.
- Even if underline mode is turned on, areas skipped with this command are not underlined.

Related *Motion Units*

Relative Print Position

Example

```
write('\x1d\x57\x2c\x01')    # Set print area width of 300
write('\x1b\x24\x64\x00')    # Set absolute print position to 100
write('\x1b\x4d\x01')        # Select character font
# Write text to see multiple lines
write('Print area width of 300 and absolute print position of 100.
 Only the first line should have this absolute print position.')
print()
```

(continues on next page)

(continued from previous page)

```
>>>
       Print area wid
th of 300 and absolut
e print position of 1
00. Only the first li
ne should have this a
bsolute print positio
n.
```

6.7 Relative Print Position - `$1B $5C`

Relative Print Position

Format `Hex $1B $5C nL nH`

`ASCII ESC \ nL nH`

`Decimal 27 92 nL nH`

Notes

- Moves the print position to [(nL + (nH × 256)) × (horizontal or vertical motion unit)] from the current position. Uses *Two Byte Number Definitions*.
- A positive number specifies movement to the right, and a negative number specifies movement to the left.
- Negative numbers are represented by the complement of 65536. For example, when moving in the left direction N motion units, use:
  - nL + nH × 256 = 65536-N
- Settings that exceed the printable area are ignored.
- If settings exceed the print area width, the relative print position is set, but no text will be able to fit in the print area width and any text will be treated as a line feed.
- When standard mode is selected, the horizontal (perpendicular to paper feed) motion unit is used.
- When this command is executed, the printer is no longer in a *New Line State*
- The horizontal and vertical motion units are specified *Motion Units Tab* Changing the horizontal or vertical motion unit does not affect the current relative print position.
- Even if underline mode is turned on, areas skipped with this command are not underlined.

Range `0 nL, nH 255, −32768 (nL + (nH × 256)) 32767`

Default `nL = 0, nH = 0`

Related *Motion Units Tab*

Absolute Print Position

Example `None`

CHAPTER 7

Paper Movement Commands

This section describes all commands that move the ticket from an idle to print state. This include cuts, ejection, presenting and retracting.

7.1 Partial Cut - `$1B $69`

Performs a partial cut on the current ticket.

Notes

- For Reliance, this command will always execute a Full Cut.
- If a ticket is not at least the minimum ticket size, then a blank portion will be printed/added to the ticket to make it the minimum size before the cut.
- If nothing has been printed, then the command is ignored.

Format `Hex $1B $69`

```
ASCII ESC i
Decimal 27 105
```

Range `None`

Default `None`

Related *Form Feed*

7.2 Full Cut - `$1B $6D`

Performs a full cut on the current ticket.

Notes

- If a ticket is not at least the minimum ticket size, then a blank portion will be printed/added to the ticket to make it the minimum size before the cut.
- If nothing has been printed, then the command is ignored.

Format `Hex $1B $6D`

`ASCII ESC m`

`Decimal 27 109`

Range `None`

Default `None`

Related *Form Feed*

7.3 Ejector - `$1D $65`

Ejector Commands

Notes

- The `m` parameter must be sent for `n = 3`, and `n = 32` .
- The `t` parameter must be sent for `n = 32`.
- When `n = 3`, 32 and the value of `m` is longer than the current ticket, the ticket will be ejected the length of the ticket.
- When `n = 2, 3, 5, 32`, the printer will cut the ticket before it executes.
- When `n = 32`, and the printer is told to print another ticket, the current ticket will be ejected or retracted based on the printer configuration. When the timeout condition has been met, the ticket is ejected or retracted based on the printer configuration.
- When in continuous mode and `m = 3, 32`, the ticket is not presented any further if the ticket is at least the minimum ticket size. This command will just enable ticket pull detection and/or the set timeout.
- This command controls the operation of the ejector and presenter. The command can be used to present, retract and/or produce a blank ticket. Also, this command can enable and disable the continuous mode feature. The value of `n` determines what the command will do and what additional (if any) parameters it may need. All additional parameters will use `m` or `t`. See table below.

n	Args	Description
1	None	`None`
2	None	Retract ticket • Only if paper retracting is enabled • This command will cut the ticket if it is not already
3	0 m 255	Present ticket with `m` steps • 1 step = 7 mm • This command will cut the ticket if it is not already.
5	None	Eject ticket • This command will cut the ticket if it is not already
6	None	Transmit ejector status byte • See *Ejector Status Table*
18	None	Disable dispenser continuous mode • While printing, the ticket remains at printer bezel. • The ticket can be cut and presented to the customer • The ticket can be cut and retracted back in
20	None	Enable dispenser continuous mode • While printing, the ticket is continuously pushed from the outlet • This is the default printer state on power up.
32	0 m 255 0 t 255	Present the ticket with m* steps and a timeout *t* • 1 step = 7 mm if it is not already.
		• This command will cut the ticket if it is not already

Ejector State Byte Table				
BIT	OFF/ON	HEX	DECI-MAL	DESCRIPTION
0	Off	00	0	Paper is present
	On	01	8	Near paper end
1	Off	00	0	`Reserved`
2	Off	00	0	Paper is not present at printer entry
	On	04	8	Paper is present at printer entry
3	Off	00	0	No presented ticket at output
	On	08	8	Presented ticket at output
4	Off	00	0	Printer's stepper motor is off
	On	10	16	Printer's stepper motor is on
5	Off	00	0	Printer's ejector motor is off
	On	20	32	Printer's ejector motor is on
6	Off	00	0	No error
	On	40	64	Error
7	Off	00	0	Printer has no jam
	On	80	128	Printer is jammed

Format `Hex $1D $65 n m t`

`ASCII GS E n m t`

`Decimal 29 101 n m t`

Range `1 n 3, 5 n 6, n = 18, n = 20, n = 32`

`0 m 255`

`0 t 255`

Default `N/A`

Related *Form Feed*

Example clear paper path

```
write("\x1d\x65\x05")   # Eject Ticket
write("\x1d\x65\x02")   # Retract Ticket
```

Example cut and present printed ticket

```
write("\x1d\x65\x03\x0c")       # Present 84 mm
write("\x1d\x65\x20\x0c\x1e")   # Present 84 mm with timeout of␣
↪30 seconds
```

Example Set and clear continuous mode

```
write("\x1d\x65\x14")   # Set continuous mode
write("\x1d\x65\x12")   # Disable continuous mode
```

7.4 Enable and Disable Auto Cut - `$1C $7D $60`

This command changes if the printer will auto-cut a ticket or not.

Format `Hex $1C $7D $60 n`

`ASCII FS } ' n`

`Decimal 28 125 140 n`

Notes

- Reliance supports enable/disable autocut through the Reliance Tools program.
- Phoenix supports enable/disable autocut through the Phoenix tools program.

n	Function
$00	Disable Auto-Cut, must send cut command
$01	Enable Auto-Cut, paper cuts itself

- If a ticket is not at least the minimum ticket size, then a blank portion will be printed/added to the ticket to make it the minimum size before the cut.

Range `n = $00, $01`

Default `None`

Related *Select Cut Mode and Cut Paper*

Example

7.5 Select Cut Mode and Cut Paper - `$1D $56`

Select the cut mode for the printer and execute a cut command.

Format `Hex $1D $56 n`

`ASCII GS V n`

`Decimal 29 86 n`

Notes

n	Function
$00	Full Cut mode
$01	Partial Cut mode

- Dip switches overide choice of full or partial

Range `n = 0,1`

Default `None`

Related *Full Cut*

Partial Cut

Example `None`

7.6 Print and Feed Paper n Lines - `$1B $64`

Print current buffer and feed n number of lines up to a maximum of 200.

Format `Hex $1B $64 n`

`ASCII ESC d n`

`Decimal 27 100 n`

Notes

- After printing, the print postion is moved to left side of the printable area. Also, the printer is in the status "Beginning of the line".

Range `n>=0, n<=255`

Default `None`

Related *Print and Feed Paper*

Example

7.7 Print and Feed Paper - `$1B $4A`

Print current buffer and feed paper.

Format `Hex $1B $4a n`

`ASCII ESC j n`

`Decimal 27 74 n`

Notes

- Any passed n value is ignored.
- Optional because the Phoenix prints on both (either) CR or LF

Range `n>=0, n<=255`

Default `None`

Related *Print and Feed Paper n Lines*

Example

CHAPTER 8

Layout Commands

This section describes all commands that affect the layout of text in terms of spacing and margins. These are advanced features that are not commonly by most users.

8.1 Right Side Character Spacing - `$1B $20`

Sets the right-side character spacing to [n × (horizontal or vertical motion unit)].

Format `Hex $1B $20 n`

`ASCII ESC SP n`

`Decimal 27 32 n`

Notes

- Settings that exceed the printable area are ignored.
- The maximum right side character spacing is 255/204 inches.
- The horizontal (perpendicular to paper feed) motion unit is used.
- The horizontal and vertical motion units are specified by *Motion Units*. Changing the horizontal or vertical motion unit does not affect the current right-side character spacing.
- Right-Side character spacing is effective until it is changed, *Initialize* is executed, the printer is reset, or the power is turned off.
- When underline mode is turned on, the right side character spacing is underlined.
- In standard mode, right side character spacing has no effect when characters are rotated 90° or 270°.

Range `0 n 255`

Default `0`

Related *Motion Units*

Relative Print Position

Example `None`

8.2 Line Spacing - `$1B $33`

Sets the line spacing to [n (vertical or horizontal motion unit)] in inches

Format `Hex $1B $33 n`

`ASCII ESC 3 n`

`Decimal 27 51 n`

Notes

- This command sets the line spacing in standard mode.
- The vertical and horizontal motion units are specified by *Motion Units*.
- Changing the vertical or horizontal motion units does not affect the current line spacing.
- If the calculation results in a fraction, the decimal portion will be ignored.
- The vertical motion unit is used.
- Minimum line spacing = 0.00492 inches (0.125mm)
- Maximum line spacing = 4 inches (101.6mm)
- Line spacing is effective until it is changed by another command, *Initialize* is executed, the printer is reset, or the power is turned off.

Range `0  n  255`

Default `34 (1/6"), n is base 10`

Related *Motion Units*

Line Spacing

Select 1/6" Line Spacing

Select 1/8" Line Spacing

Example `None`

8.3 Select 1/6 Inch Line Spacing - `$1B $32`

Sets the line spacing to 1/6 an inch

Format `Hex $1B $32`

`ASCII ESC 2`

`Decimal 27 50`

Notes

- This command sets the line spacing in standard mode.
- Line spacing is effective until it is changed by another command, *Initialize* is executed, the printer is reset, or the power is turned off.

Range `None`

Default `None`

Related

Select 1/8" Line Spacing

Motion Units

Line Spacing

Example `None`

8.4 Select 1/8 Inch Line Spacing - `$1B $30`

Sets the line spacing to 1/8 an inch

Format `Hex $1B $30`

`ASCII ESC 0`

`Decimal 27 48`

Notes

- This command sets the line spacing in standard mode.
- Line spacing is effective until it is changed by another command, *Initialize* is executed, the printer is reset, or the power is turned off.

Range `None`

Default `None`

Related *1/6 inch spacing*

Motion Units

Line Spacing

Example `None`

8.5 Select Justification - `$1B $61`

Select justification mode

Format `Hex $1B $61 n`

`ASCII ESC a n`

`Decimal 27 97 n`

Notes

- This command is enabled only when processed at the beginning of a line. (When the current line is empty)
- This command applies the justification within the area set by *Left Margin* and *Print Area Width*
- This command will justify all data in the printing area such as characters, graphics, bit images, barcode and space area set by *Horizontal Tab*, *Absolute* and *Relative* print positions.
- Settings of this command are effective until the *Initialize* command is executed, the printer is reset, or the power is turned off.
- When n=0 or 48, left justification is enabled
- When n=1 or 49, center justification is enabled
- When n=2 or 50, right justification is enabled

Range `n=0,1,2,48,49,50`

Default `0`

Related *Left Margin*

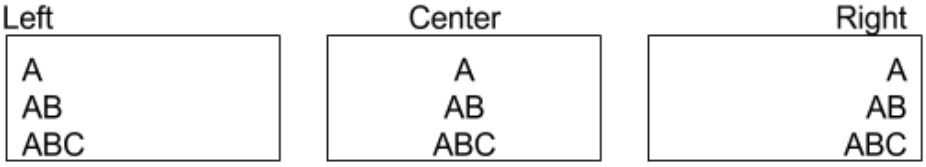
Print Area Width

Absolute Print Position

Relative Print Position

Horizontal Tab

Example



—

8.6 Left Margin - `$1D $4C`

Set left margin

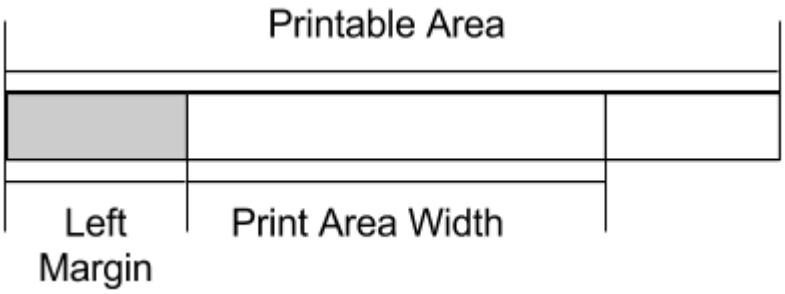
Format `Hex $1D $4C nL nH`

`ASCII GS L nL nH`

`Decimal 27 76 nL nH`

Notes

- In standard mode, sets the left margin to [(nL + (nH × 256)) × (horizontal motion unit)] from the left edge of the printable area. Uses *Two Byte Number Definitions*.
- This command is enabled only when processed at the beginning of a line. (When the current line is empty)
- This command applies the justification within the area set by *Left Margin* and *Print Area Width*
- This command will justify all data in the printing area such as characters, graphics, bit images, bar code and space area set by *Horizontal Tab*, *Absolute* and *Relative* print positions.
- Settings of this command are effective until *Initialize* is executed, the printer is reset, or the power is turned off.



Range `0 nL, nH 255, 0 (nL + (nH × 256)) 65535`

Default `nL = 0, nH =0`

Related *Motion Units*

Print Area Width

Example `None`

8.7 Motion Units - `$1D $50`

Set horizontal and vertical motion units

Format `Hex $1D $50 x y`

`ASCII GS P x y`

`Decimal 29 80 x y`

Notes

- Sets the horizontal and vertical motion units to approximately 25.4/x mm {1/x"} and approximately 25.4/y mm {1/y"}, respectively.
- When x = 0, the default value of the horizontal motion unit is used.
- When y = 0, the default value of the vertical motion unit is used.
- When x > 204, the default value of the horizontal motion unit is used.
- When y > 204, the default value of the vertical motion unit is used.
- The horizontal direction is perpendicular to the paper feed direction and the vertical direction is the paper feed direction.
- The horizontal and vertical motion units indicate the minimum pitch used for calculating the values of related commands
- In standard mode, the following commands use x or y.
  - Commands using x: *Left Margin*, *Print Area Width*
  - Commands using y
- If the result is a decimal number, the decimal is ignored.
- This command does not affect the previously defined values for settings that use the horizontal or vertical motion units.
- Settings of this command are effective until it is changed, *Initialize* is executed, the printer is reset, or the power is turned off.

Range `0  x, y  204`

Default `x = 204, y = 204`

Related *Left Margin*

Print Area Width

Example `None`

8.8 Print Area Width - `$1D $57`

Set print area width

Format `Hex $1D $57 nL nH`

`ASCII GS W nL nH`

`Decimal 29 87 nL nH`

Notes

- In standard mode, sets the print area width to [(nL + (nH × 256)) × (horizontal motion unit)] from the right edge of the left margin. See *Two-byte Numbers*
- This command is only executed when the printer is in a *New Line State*
- If the setting exceeds the printable area, the print area width is automatically set to the maximum value of the printable area.
- If the [left margin + print area width] is greater than the printable area, the print area will be truncated automatically to [printable area - left margin]. If the left margin is changed, the print area width will also change until there is room to fit the specified print area width.
- If the print area width equals 0, then the print area width is automatically set to the maximum value of the printable area.
- The horizontal (perpendicular to paper feed direction) motion unit is used to set print area width.
- The horizontal and vertical motion units are specified by *Motion Units*. Changing the horizontal or vertical motion unit does not affect the current print area width.
- Settings of this command are effective until it is changed, *Initialize* is executed, the printer is reset, or the power is turned off.

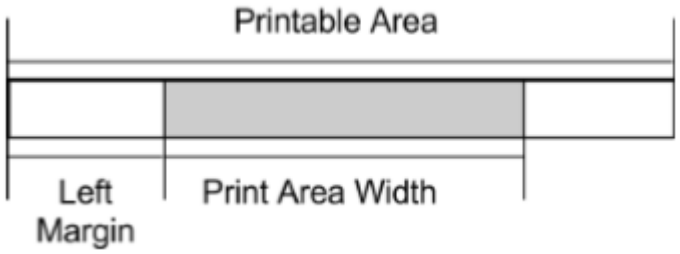
Range `0 nL, nH 255, 0 (nL + (nH × 256)) 65535`

Default `nL = 64, nH = 2`

Related *Motion Units*

Print Area Width

Example



CHAPTER 9

Images and Barcode

This section describes functions for raster images, bitmaps, bar codes, and QR Code®.

9.1 2D Barcode Generator - `$1C $7D $25 k d1...dk`

Encodes the specified string as a center justified 2D barcode. Only k bytes of the string will be read and any remaining will be treated as regular text or ESC/POS commands. The command and data must be enclosed by *Line Feeds*.

Note: Requires firmware 1.9 or newer

Format `Hex $1C $7D $25 k d1...dk`

`ASCII FS } % k d1...dk`

`Decimal 28 125 37 k d1...dk`

Notes

- This 2D barcode is compliant with the QR Code® specification and can be read by all 2D barcode readers.
- This command must be sent when the current line is empty. If not, the command will be ignored, and the bytes to be encoded will be printed as text.
- Up to 154 8-bit characters are supported when used with firmware before version 1.29. Firmware version 1.29 and higher supports up to 255 characters.
- If the input string length exceeds the range specified by the k parameter, only the first 154 characters (255 character if using version 1.29 or higher) will be encoded. The rest of the characters to be encoded will be printed as regular ESC/POS characters on a new line.

Range `0 < k  154 8-bit alphanumeric and URL-safe characters for version < 1.29  0 < k  255 8-bit alphanumeric and URL-safe characters for version  1.29`

Default `None`

Related `None`

Example

```
write("\x0a")           # Beginning line feed
write("\x1c\x7d\x25")   # Start QR Code® command
write("\x1C")           # Length of string to follow (28 bytes in␣
↪this example)
write("https://pyramidacceptors.com")
write("\x0a")           # Ending line feed
print()
>>>
```

Example

```
write("\x0a")           # Beginning line feed
write("\x1c\x7d\x25")   # Start QR Code® command
write("\x06")           # Length of string to follow (6 bytes in␣
↪this example)
write("\xE5\x90\x8C\xE5\x83\x9A") #  (Colleague)
write("\x0a")           # Ending line feed
print()
>>>
```

QR Code® is a registered trademark of DENSO WAVE INCORPORATED.

9.2 Dynamic 2D Barcode - `$1D $28 $6B`

Encodes and prints a string of characters up to 32 characters long

---

Note: Requires Universal firmware for Phoenix.

Format `Hex $1D $28 $6B pL pH cn fn [parameters]`

`ASCII GS ( k pL pH cn fn [parameters]`

`Decimal 29 40 107 pL pH cn fn [parameters]`

Notes

- pL is the length of the string to encode plus three. This represent the total count of bytes following the pL parameter.
- pL and pH specify the number of bytes following cn as (pL + pH × 256).
- The function is specified with the function code (fn). Details of the performance differ according to the function.
- cn and fn are specified in decimal format.

cn	fn	Function no.	Function Name
49	80	Function 180	QR Code: Store the data in the symbol storage area.
	81	Function 181	QR Code: Print the symbol data in the symbol storage area.

cn	fn	Function no.	[parameters]
49	80	Function 180	m = 48
	81	Function 181	m = 48, d1. . . dk

- **Function 180:** Stores the QR Code symbol data (d1. . . dk) in the symbol storage area.
- **Function 181:** Prints the 2D barcode if available. Must be called after sending the Generate command. Once this command is sent, the generated barcode is erased. Each QR code must be printed by first generating then printing.

Range `(pL + pH x 256) = 4 - 7092` **Function 180**

`(pL + pH x 256) = 3` **Function 181**

`cn = 49, fn = 80,81`

`m = 48`

`d = 0 - 255`

`k = (pL + pH x 256) - 3`

Default `None`

Related `None`

Example Print URL as 2D barcode (sample binary `bin`)

```
write("\x0d")                                  # newline
write("\1d\x28\x6b")                           # start command
write("\x1f\x00")                              # string length (28␣
↪bytes + 3)
write("\x31\x50\x31")                          # rest of command...
write("https://pyramidacceptors.com")          # Actual string
write("\x1d\x28\x6b\x03\x00\x31\x51\x31") # Print now
```

9.3 Set 2D Barcode Size - `$1C $7D $74 k`

Sets the width/height of QR code cells, measured in dots (8 dots = 1mm).

Note: Requires firmware 1.29 or newer

Format `Hex $1C $7D $74 k`

`ASCII FS } t k`

`Decimal 28 125 116 k`

Notes

- If the QR code generated with the configured size setting is too big to be printed on the paper being used, the printer will automatically choose smaller sizes, down to the smallest valid size (3 dots per cell) until a suitable size has been found. If no suitable size can be found, the QR code will not be printed. This will only occur when using paper smaller than 80mm with the largest valid QR code size.
- Paper size is configured through the Pyramid Reliance Tools application.
- If the size requested by the host is outside of the valid range, the size setting will not be changed.

Range `3 ≤ k ≤ 8`

Default `k = 8`

Related `None`

Example

```
write("\x1c\x7d\x74\x04")      # Set the QR code to 4 dots per␣
↪cell (4 dots = 0.5mm)
```

9.4 Barcode Generator (1) - `$1D $6B m d1...dk $00`

9.5 Barcode Generator (2) - `$1D $6B m n d1...dn`

- Defines and prints a 1D barcode using the mode specified by *m*. This command has two forms. Form 1 does not take the string length *n*, but reads all bytes after *m* and before the first NUL byte (0x00) received as the string to encode. Form 2 of the command reads *n* bytes following *n* as the string to encode. The form used is determined by the value of *m* received.
- Form 1: 0 *m* 20

m	Barcode System	No. of Characters	Valid Characters (decimal)	Minimum Firmware Version
4	Code 39	1 k	48 d 57, 65 d 90, 32, 36, 37, 43, 45, 46, 47, 58	1.18
5	ITF	1 k, k must be even	48 d 57	1.21
8	Code 128	1 k	1 d 127	1.18

- Form 2: 65 *m* 90

m	Barcode System	No. of Characters	Valid Characters (decimal)	Minimum Firmware Version
69	Code 39	1 n	48 d 57, 65 d 90, 32, 36, 37, 43, 45, 46, 47, 58	1.21
70	ITF	1 n, k must be even	48 d 57	1.21
73	Code 128	1 n	0 d 127	1.18

- Form 2 of the command allows a NUL byte to be encoded when used with Code 128.

Note: Requires firmware 1.18 or newer

Format `Hex (1) $1D $6B m d1...dk $00`

`Hex (2) $1D $6B m n d1...dk`

`ASCII (1) GS k m d1...dk NUL`

`ASCII (2) GS k m n d1...dk`

`Decimal (1) 29 107 m d1...dk 0`

`Decimal (2) 29 107 m n d1...dk`

Notes

- If there is data in the buffer when the printer receives this command, the buffered data will be printed, and the barcode will be printed on the following line.
- If the barcode generated is too long to be printed, nothing will be printed.

- If an invalid value of *m* is sent, no barcode will be printed, and the string sent will be parsed normally.
- If an invalid character is sent, the text "HRI NOT OK" will be printed.
- Barcode justification is set by the `$1B $61` (Select Justification) command.
- Barcode height is set by the `$1D $68` (Set 1D Barcode Height) command.
- Barcode width is set by the `$1D $77` (Set 1D Barcode Width Multiplier) command.

Notes for Code 128

- To encode a string with a NUL byte, the second form of the barcode generator command must be used. In this case, the string length *n* equals the count of all characters following *n*.
- Characters that are within the valid range defined in the table above, but are invalid to the current mode are ignored, and not encoded.
- Special characters (mode select, mode shift, FNC) are transmitted by sending the '{' character before the special character. The first two characters following *m* must select either mode A, B, or C. The '{' character is transmitted by sending two '{' characters. A special character (one that is preceded by '{') that is not defined in the table below is ignored, and not encoded.
- If the first two characters following *m* do not select a valid mode, the text "HRI NOT OK" is printed.

Character	Hexadecimal	ASCII	Decimal
Shift	$7B $53	{ S	123 83
Mode A	$7B $41	{ A	123 65
Mode B	$7B $42	{ B	123 66
Mode C	$7B $43	{ C	123 67
FNC 1	$7B $31	{ 1	123 49
FNC 2	$7B $32	{ 2	123 50
FNC 3	$7B $33	{ 3	123 51
FNC 4	$7B $34	{ 4	123 52
'{'	$7B $7B	{ {	123 123

Range See table above for range of valid barcode systems, and the range of valid characters and string lengths for each system.

Default `None`

Related *Select Justification*

Set 1D Barcode Height

Set 1D Barcode Width Multiplier

Example

```
# Encode the text "CODE 39" as a Code 39 barcode
write("\x1d\x6b\x04\x43\x4f\x44\x45\x20\x33\x39\x00")

# Encode the text "Code 128" as a Code 128 barcode,
# using form 1 of the command, and mode B
write("\x1d\x6b\x08\x7b\x42\x43\x6f\x64\x65\x20\x31\x32\x38\x00")
```

(continues on next page)

(continued from previous page)

```
# Encode the text "pi = 3.14159265" as a Code 128 barcode,
# using form 2 of the command, and modes B and C
# Command header (includes code system and string length)
write("\x1d\x6b\x49\x0f")

# Mode B select, and the string "pi = 3."
write("\x7b\x42\x70\x69\x20\x3a\x20\x33\x2e")

# Mode C select, and the string "14159265"
write("\x7b\x43\x0e\x0f\x5c\x41")
```

9.6 Set 1D Barcode Width Multiplier - `$1D $77 n`

Sets the 1D barcode width multiplier.

Format `Hex $1D $77 n`

`ASCII GS w n`

`Decimal 29 119 n`

Notes

- The barcode is scaled horizontally by *n* units. A value of 2 doubles the width of each bar in the barcode, doubling the width of the entire barcode. A value of 1 does not scale the barcode. In an unscaled barcode, the thinnest bar has a width of one dot (0.12499975mm, or 0.00492125 inches).
- This parameter does not need to be set for every barcode. It is only reset to the default value when the printer is rebooted.
- If an invalid (out of range) value of *n* is sent, the command is ignored.
- When using code 128, a scalar of 1 may produce barcodes that are valid, but too small to be properly read.

Range `1 n 6`

Default `n = 2`

Example

```
# Set the 1D barcode width to be three times the base width
write("\x1d\x77\x03")
```

9.7 Set 1D Barcode Height - `$1D $68 n`

Sets the 1D barcode height, measured in dots.

Format `Hex $1D $68 n`

`ASCII GS h n`

`Decimal 29 104 n`

Notes

- The barcode height *n* is measured in dots. One dot equals 0.12499975mm, or 0.00492125 inches.
- This parameter does not need to be set for every barcode. It is only reset to the default value when the printer is rebooted.
- If an invalid (out of range) value of *n* is sent, the command is ignored.

Range `1 n 255`

Default `n = 100`

Example

```
# Set the 1D barcode height to 50 dots
write("\x1d\x68\x32")
```

9.8 Set HRI Printing Position - `$1D $48 n`

Sets HRI Printing Position based on *n*:

n	Position
0, 48	Not printed
1, 49	Above the barcode
2, 50	Below the barcode
3, 51	Both above and below the barcode

If an invalid value of *n* is used, the command is ignored.

Format `Hex $1D $48 n`

`ASCII GS H n`

`Decimal 29 72 n`

Range `0 n 3, 48 n 51`

Default `n = 0`

Example

```
write("\x1d\x48\x32")    # Set HRI characters to print below the␣
↪barcode
```

9.9 Set HRI Font - `$1D $66 n`

Sets HRI Font *n*:

n	Position
0, 48	Font A (15 CPI)
1, 49	Font B (20 CPI)

If an invalid value of *n* is used, the command is ignored.

Format `Hex $1D $66 n`

`ASCII GS f n`

`Decimal 29 102 n`

Notes

- CPI means "characters per inch". A higher CPI equates to a smaller, more compact font.

Range `n = 0, 1, 48, 49`

Default `n = 0`

Example

```
write("\x1d\x66\x31")    # Set HRI characters to print using font␣
↪B.
```

9.10 Raster Image - `$1D $76 $30 m xL xH yL yH d1...dk`

Prints a raster image

Format `Hex $1D $76 30 m xL xH yL yH d1...dk`

`ASCII GS v % m xL xH yL yH d1...dk`

`Decimal 29 118 48 m xL xH yL yH d1...dk`

Notes

- When standard mode is enabled, this command is only executed when there is no data in the print buffer. (Line is empty)
- The defined data (d) defines each byte of the raster image. Each bit in every byte defines a pixel. A bit set to 1 is printed and a bit set to 0 is not printed.
- If a raster bit image exceeds one line, the excess data is not printed.
- This command feeds as much paper as is required to print the entire raster bit image, regardless of line spacing defined by *1/6" or 1/8"* commands.
- After the raster bit image is printed, the print position goes to the beginning of the line.
- The following commands have no effect on a raster bit image:
  - Emphasized
  - Double Strike
  - Underline
  - White/Black Inverse Printing
  - Upside-Down Printing
  - Rotation
  - Left margin
  - Print Area Width
- A raster bit image data is printed in the following order:

d1	d2	. . .	dx
dx + 1	dx + 2	. . .	dx * 2
.	.	.	.
. . .	dk - 2	dk - 1	dk

- Defines and prints a raster bit image using the mode specified by m:

m	Mode	Width Scalar	Heigh Scalar
0, 48	Normal	x1	x1
1, 49	Double Width	x2	x1
2, 50	Double Height	x1	x2
3, 51	Double Width/Height	x2	x2

- xL, xH defines the raster bit image in the horizontal direction in bytes using two-byte number definitions. (xL + (xH * 256)) Bytes

- yL, yH defines the raster bit image in the vertical direction in dots using two-byte number definitions. (yL + (yH * 256)) Dots
- d specifies the bit image data in raster format.
- k indicates the number of bytes in the bit image. k is not transmitted and is there for explanation only.
- Image bytes should be in MSB order.

Range `0 m 3, 48 m 51`

```
1 xL + (xH * 256) 65535
1 yL + (yH * 256) 2047
0 D1...Dk 255
k = (xL + (xH * 256)) * (yL + (yH * 256))
```

Default `N/A`

Related `N/A`

Example Github

9.11 Print Graphic Bank/Logo - `$1B $FA`

Prints logo `n` from internal storage using dimensions defined as *Two Byte Numbers*.

Format `Hex $1B $FA n xH xL yH yL`

`ASCII ESC {} n xH xL yH yL`

`Decimal 27 250 n xH xL yH yL`

Notes

- n specifies which logo to print.
- $xL + (xHx256)$ specifies the starting dotline.
- Dolines start at line 0.
- $yL + (yHx256)$ specifies the number of dotlines to print.
- If $xL + (xHx256)$ is greater than the specified logo's height, the printer does not execute the command.
- If $[(xL + (xHx256)) + (yL + (yHx256))]$ is greater than the specified logo's height, only dotlines from the specified start dotline to the end of the logo will be printed.
- If the logo specified by n has not been downloaded or n is out of range, then logo 1 will be printed.

Range `1 n 255`

`0 xH, xL, yH, yL, 255`

Default `N/A`

Related *Print Graphic Bank/Logo (Simplified)*

Example Print logo 1 from dotlines 10 to 200

```
write('\x1b\xfa\x01\x00\x0a\x00\xc8')
```

Example Print logo 2 from dotlines 0 to 861

```
write('\x1b\xfa\x02\x00\x00\x03\x5e')   #  862 dotlines total
```

9.12 Print Graphic Bank/Logo (Simplified) - `$1C $79`

Prints logo `n` from internal storage using the dimensions stored in flash. This command is similar to the "Print Graphic Bank/Logo" command using the command bytes `$1B $FA`, but does not need the dimensions to be specified as part of the command.

Format `Hex $1C $79 n $00`

`ASCII FS y n NUL`

`Decimal 28 121 n 0`

Notes

- n specifies which logo to print.
- The fourth byte of this command is an option specifier reserved for future use. It must be set to zero.
- If the logo specified by n has not been downloaded or n is out of range, then nothing will be printed.

Range `1 n 255`

Default `N/A`

Related *Print Graphic Bank/Logo*

Example Print the second logo

```
write('\x1c\x79\x02\x00')
```

CHAPTER 10

Reliance Status

This command provides a wealth of information about the printers current status. The command will always receive a response unless the printer is offline (powered down, disconnected, etc.) or paper is actively being fed through the printer. A numeric argument is provided which controls the meaning of each bit in the returned byte(s). There is some duplication between fields for legacy support reasons but you effectively have access to all error and status conditions.

Tip: See realtime status examples on Github: Thermal Talk API

10.1 Real Time Status - `$10 $04`

Transmits the printer status in real time.

Format `Hex $10 $04 n`

`ASCII DLE EOT n`

`Decimal 16 04 n`

Notes

- This command is processed in real time. The reply to this command is sent whenever it is received and does not wait for previous ESC/POS commands to be executed first.

Range `n = 1 , 2, 3, 4, n = 17, n = 20`

n	Status
1	Transmit the printer status
2	Transmit the off-line printer status
3	Transmit error status
4	Transmit paper roll sensor status
17	Transmit the print status
20	Transmit Full Status (6 Byte Reply)

n=1 (hexadecimal $01) Printer Status

BIT	OFF/ON	HEX	DECIMAL	DESCRIPTION
0	-	-	-	Reserved
1	-	-	-	Reserved
2	-	-	-	Reserved
3	Off	00	0	Online
	On	08	8	Offline
4	-	-	-	Reserved
5	-	-	-	Reserved
6	-	-	-	Reserved
7	-	-	-	Reserved

n=2 (hexadecimal $02) Offline Status

BIT	OFF/ON	HEX	DECIMAL	DESCRIPTION
0	-	-	-	Reserved
1	-	-	-	Reserved
2	Off	00	0	Cover is closed
	On	04	4	Cover is open
3	Off	00	0	Paper is not fed with DIAG button
	On	08	8	Paper is fed with DIAG button
4	-	-	-	Reserved
5	Off	00	0	Paper is present
	On	20	32	Printing stopped due to paper end
6	Off	00	0	No error
	On	40	64	Error
7	-	-	-	Reserved

Note: DIAG Button: This bit is *always* set because our diagnostic button is always enabled.

Note: Error: This bit means that *any* error has been reported. Query the other status commands to determine the precise error.

n=3 (hexadecimal $03) Error Status

BIT	OFF/ON	HEX	DECIMAL	DESCRIPTION
0	-	-	-	Reserved
1	-	-	-	Reserved
2	-	-	-	Reserved
3	Off	00	0	Cutter Okay
	On	08	8	Cutter Error
4	-	-	-	Reserved
5	Off	00	0	No unrecoverable error
	On	20	32	Unrecoverable error
6	Off	00	0	No auto-recoverable error
	On	40	64	Auto-recoverable error
7	-	-	-	Reserved

n=4 (hexadecimal $04) Paper Roll Sensor Status

BIT	OFF/ON	HEX	DECIMAL	DESCRIPTION
0	-	-	-	Reserved
1	-	-	-	Reserved
2,3	Off	00	0	Paper present in abundance
	On	0C	12	Paper low
4	-	-	-	Reserved
5,6	Off	00	0	Paper present
	On	60	96	Paper not present
7	-	-	-	Reserved

n=17 (hexadecimal $11) Print Status

BIT	OFF/ON	HEX	DECIMAL	DESCRIPTION
0	-	-	-	Reserved
1	-	-	-	Reserved
2	Off	00	0	Paper motor off
	On	04	4	Paper motor on
3	-	-	-	Reserved
4	-	-	-	Reserved
5	Off	00	0	Paper present
	On	20	32	Printing stopped due to paper end
6	-	-	-	Reserved
7	-	-	-	Reserved

n=20 (hexadecimal $14) Full Status 1st Byte = $10 (DLE)

2nd Byte = $0F

3rd Byte

BIT	OFF/ON	HEX	DECIMAL	DESCRIPTION
0	Off	00	0	Paper Present
	On	01	1	Paper Not Present
1	-	-	-	Reserved
2	Off	00	0	Paper present in abundance
	On	04	4	Near paper end
3	-	-	-	Reserved
4	-	-	-	Reserved
5	Off	00	0	Ticket not present at output
	On	20	32	Ticket present at output
6	-	-	-	Reserved
7	-	-	-	Reserved

4th Byte

BIT	OFF/ON	HEX	DECIMAL	DESCRIPTION
0	Off	00	0	Cover is closed
	On	01	1	Cover is open
1	Off	00	0	Cover is closed
	On	02	2	Cover is open
2	-	-	-	Reserved
3	Off	00	0	Paper motor off
	On	08	8	Paper motor on
4	-	-	-	Reserved
5	Off	00	0	DIAG button released
	On	20	32	DIAG button pressed
6	-	-	-	Reserved
7	-	-	-	Reserved

5th Byte

BIT	OFF/ON	HEX	DECIMAL	DESCRIPTION
0	Off	00	0	Head temperature ok
	On	01	1	Head temperature ok
1	Off	00	0	No Communication Error
	On	02	2	RS232 Error
2	-	-	-	Reserved
3	Off	00	0	Power supply voltage ok
	On	08	8	Power supply voltage error
4	-	-	-	Reserved
5	-	-	-	Reserved
6	Off	00	0	Free paper path
	On	40	64	Paper jam
7	-	-	-	Reserved

6th Byte

BIT	OFF/ON	HEX	DECIMAL	DESCRIPTION
0	Off	00	0	Cutter ok
	On	01	1	Cutter error
1	-	-	-	Reserved
2	-	-	-	Reserved
3	-	-	-	Reserved
4	-	-	-	Reserved
5	-	-	-	Reserved
6	-	-	-	Reserved
7	-	-	-	Reserved

Default `None`

Related `None`

Example of No Paper

```
write("\x10\x04\x04")    # Paper Roll Status
>>> 0b01101100           # $6C or 108, this means that there is no␣
↪paper
```

Example of Low Paper

```
write("\x10\x04\x04")    # Paper Roll Status
>>> 0b00001100           # $0C or 12, this means that the paper␣
↪level is low
```

CHAPTER 11

Phoenix Status

This command provides a wealth of information about the printers current status. The command will always receive a response unless the printer is offline (powered down, disconnected, etc.) or paper is actively being fed through the printer. A numeric argument is provided which controls the meaning of each bit in the returned byte(s). There is some duplication between fields for legacy support reasons but you effectively have access to all error and status conditions.

Tip: See realtime status examples on Github: Thermal Talk API

11.1 Real Time Status - `$10 $04`

Transmits the printer status in real time.

Format `Hex $10 $04 n`

`ASCII DLE EOT n`

`Decimal 16 04 n`

Notes

- This command will respond with the current real-time status of the printer. It will be processed in order of reception and after any current printing is complete.

Range `n = 1 , 2, 3, 4`

n	Status
1	Transmit the printer status
2	Transmit the off-line printer status
3	Transmit error status
4	Transmit paper roll sensor status

n=1 (hexadecimal $01) Printer Status

BIT	OFF/ON	HEX	DECIMAL	DESCRIPTION
0	-	-	-	Reserved
1	-	-	-	Reserved
2	-	-	-	Reserved
3	Off	00	0	Online
	On	08	8	Offline
4	-	-	-	Reserved
5	-	-	-	Reserved
6	-	-	-	Reserved
7	-	-	-	Reserved

n=2 (hexadecimal $02) Offline Status

BIT	OFF/ON	HEX	DECIMAL	DESCRIPTION
0	-	-	-	Reserved
1	-	-	-	Reserved
2	-	-	-	Reserved
3	-	-	-	Reserved
4	-	-	-	Reserved
5	Off	00	0	Paper is present
	On	20	32	Printing stopped due to paper end
6	Off	00	0	No error
	On	40	64	Error
7	-	-	-	Reserved

Note: Error: This bit means that *any* error has been reported.

n=3 (hexadecimal $03) Error Status

Note: Error Status: This will always return $00 since no errors in Phoenix are auto-recoverable.

n=4 (hexadecimal $04) Paper Roll Sensor Status

HEX	DECIMAL	DESCRIPTION
1E	30	Paper low
72	114	Paper not present

Default `None`

Related `None`

Example of No Paper

```
write("\x10\x04\x04")    # Paper Roll Status
>>> 0b01110010           # $72 or 114, this means that there is no␣
↪paper
```

Example of Low Paper

```
write("\x10\x04\x04")    # Paper Roll Status
>>> 0b00011110           # $1E or 30, this means that the paper␣
↪level is low
```

CHAPTER 12

Printer Command Set Table

Function Name	HEX	ASCII	Reliance	Phoenix
Line feed	0A	LF	✓	✓
Carriage return	0D	CR	✓	✓
Initialize printer	1B 40	ESC @	✓	✓
Horizontal tab	09	HT	✓	
Form Feed	0C	FF	✓	
Cancel Current Line	18	CAN	✓	
Absolute Print Position	1B 24	ESC $	✓	
Relative Print Position	1B 5C	ESC	✓	
Select font A	1B 50	ESC P		✓
Select font C	1B 54	ESC T		✓
Select font D	1B 55	ESC U		✓
Paper Status	1B 76	ESC v		✓
Print and feed paper n lines	1B 64	ESC d		✓
Print and paper feed	1B 4A	ESC J		✓
Reliance Real-time status	10 04	DLE EOT	✓	
Phoenix Real-time status	10 04	DLE EOT		✓
Select print mode	1B 21	ESC !	✓	✓
Underline mode	1B 2D	ESC -	✓	✓
Italics mode	1B 34	ESC 4	✓	✓
Emphasis mode	1B 45	ESC E	✓	✓
Select character font	1B 4D	ESC M	✓	✓
90° Rotation	1B 56	ESC V	✓	
Select Character Page	1B 74	ESC t	✓	
Upside-down mode	1B 7B	ESC {	✓	
Set CPI mode	1B C1	ESC Á	✓	
Select Codepage	1C 7D 26	FS } &	✓	
Select character size	1D 21	GS !	✓	✓
Reverse print mode	1D 42	GS B	✓	✓

Continued on next page

Table 1 – continued from previous page

Function Name	HEX	ASCII	Reliance	Phoenix
Select double strike mode	1B 47	ESC G		✓
Right side character spacing	1B 20	ESC SP	✓	
Line spacing	1B 33	ESC 3	✓	
Select 1/6 inch line spacing	1B 32	ESC 2	✓	
Select 1/8 inch line spacing	1B 30	ESC 0	✓	
Select justification	1B 61	ESC a	✓	✓
Left margin	1D 4C	GS L	✓	
Motion units	1D 50	GS P	✓	
Print area width	1D 57	GS W	✓	
Select Cut Mode and Cut Paper	1D 56	GS V		✓
Full Cut	1B 6D	ESC m		✓
Partial Cut	1B 69	ESC i	✓	✓
Ejector	1D 65	GS E	✓	
Printer ID	1D 49	GS I	✓	✓
Transmit Status	1D 72	GS r	✓	
Enable and Disable Auto Cut	1C 7D 60	FS } ‘		✓
Raster Image	1D 76 30	GS v %	✓	✓
Dynamic 2D Barcode	1D 28 6B	GS (k		✓
2D Barcode Generator	1C 7D 25	FS } %	✓	
Set 2D Barcode Size	1C 7D 74	FS } t	✓	
Barcode Generator	1D 6B	GS k	✓	
Set 1D Barcode Width Multiplier	1D 77	GS w	✓	
Set 1D Barcode Height	1D 68	GS h	✓	
Set HRI Printing Position	1D 48	GS H	✓	
Set HRI Font	1D 66	GS f	✓	
Print Graphic Bank/Logo	1B FA	ESC {}	✓	
Print Graphic Bank/Logo Simplified	1C 79	FS y	✓	

CHAPTER 13

Indices and tables

- genindex

Index

Symbols

D

E

F

I

L

M

P

R

T