RD- EH

Embedded thermal printers specification of Development

目录

Chapter 1: Overview

Easy paper loading structure.

Appearance: fashion, delicacy, small volume, light weight. And the printer supports the 3.3V voltage.

It can be easily integrated into customer system equipment. And high printing speed, and print clearly and smoothly.

RD-EH32-S micro thermal printer are widely used in medical, fire fighting, electric power, weighing apparatus, GPS navigation and other industries.

1.1Performance indicators

Printing performance	Model	RD-EH32-8+\SN\485	RD-EH32-V2
	printing method	thermal printing	
	Printing speed	50mm/s（MAX）	65mm/s（MAX）
	Effective print width	48mm	
	Resolution ration	8dots per millimeter, 384 dots per line	
	Paper feed step	0.125mm	
	Foreign language	1.support standard ASCII characters (96): 5×7， 2.Support extended ASCII characters (352): 6×8， 3.support standard ASCII characters (224): 12×24，	

	characters	4. support the User-defined character: 6×8 5.user option: ASCII characters of 12x24&8X16&8X12	
	Chinese characters	Equip with the GB2313 Chinese Character library of 24×24 (user option: 16×16 or 12×12)	Equip with the GBK font of 24×24, which contains a total of about 20000 Chinese characters，and supports the rare Chinese characters printing
	graphics	User option	
	bar code	User option	
Detection function	lack of paper detection	yes	
	Auto sleep function	No	Yes
	Voltage detection	No	Yes
Interface parameters	Wired Interface	standard parallel interface: 20 wire double row needle	
		Standard serial interface: 10 wire	Standard serial interface: 5 core white base

		double row needle (RS232&TTL)	
		485 interface: 10 wire double row needle	
control system	buffer	32K	8K
	instruction system	print command: ESC/P (Compatible with IBM/EPSON and ESC/P)	
	Print drivers	WIN2000/NT/XP/WIN7	
power supply	operating voltage	DC5V±5%，3A (can select DC5V～9V)	DC3.5V～9V，3A
	Operating current	average ：1 A ~ 1.5 A,; MAX ：2A～3A （according to customer requirements, We can adjust the power consumption）	
reliability	print head life	50km	
Printing paper	Paper type	Ordinary thermal Paper (width: 58±0.5mm，ø≤30mm)	
	paper loading way	loading paper from the front、easy paper loading	
	Cut type	tearing paper manually	
Physical characteristics	Operating environment	Temperature: -10～55℃ Humidity: 10～80%(RH)	Temperature:-20～55℃ Humidity: 10～80%RH
	Storage	Temperature: -20～60℃	

	temperature / humidity	Humidity: 10～90%(RH)	
	bare machine weight	approximately 150g（Contain printing paper）	
	Installation dimension (mm)	77×53×42 (W x H x D)	77×53×33 (W x H x D)
	outline dimension (mm)	82×58×45 (W x H x D)	82×58×36 (W x H x D)

1.2operating

1.2.1key

There is a key on the printer, and we can realize self-test and paper feed according to the key.

1.2.2self-test

Step 1: Load the printing paper

Step 2: In the case of no charging the printer, please hold down the key, then charge the printer hold down the key for 2 second, and the printer starts the self-test. (the self-test can print the model of the machine, company telephone, company name, interface parameters and other basic information)

1.2.3loading paper

Step 1: Open the paper storehouse door

Step 2: Directly put the thermal roll paper into the paper store house in the proper direction and the smooth side up

Step 3: Place the paper to the extent that it can be exposed from the printer and close the paper storehouse cover. And press the paper.

1.2.4 Paper feed

Under the printer charged state, hold down the key, and the printing paper starts feeding, and loosen the button, the printing paper stops feeding.

1.2.5Indicator light

RD-EM32-S printer has an Indicator light, which is green and has two functions. When the printer lacks the paper, the indicator light will be in a state of continuous glint. When the print is online, it will be bright for long. When the printer buffer is full or printer is busy, the green indicator light will go out.

Chapter 2: communication interface

2.1 serial interface

2.1.1Define of Interface

The default communication is as follows:

Baud rate: 9600 bps

Data Length: 8 Bit

Parity: None

Stopping bit: 1 bit

Interface: 5 core socket (figure2-1), and the distance between a needle and another needle is 2mm.

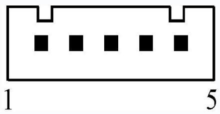


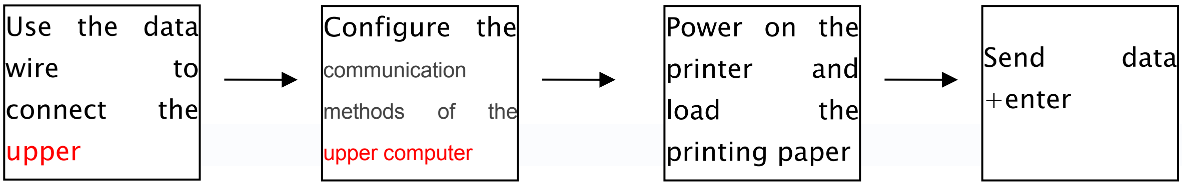
figure2-1 figure2-2

5 core single row socket (Pin No.)	DB-9 Core socket(pin NO.)	Signal name	signal source	direction	Illustration
3	3	TXD	Host	Import(in)	The printer receives the data from the main computer. (TRANSMIT DATA)
2	2	RXD	Printer	Export(out)	When using the 'X-ON/X-OFF' Handshake Protocol, the printer sends control code 'X-ON/X-OFF' to the computer. (RECEIVE DATA)
5	5	GND	——	——	Signal ground
4	8	CTS	Printer	Export(out)	When the signal is in a state of 'MARK', it means that the printer is busy and can't receive data. But when the signal is in a state of 'SPACE', it means that the printer is ready to receive data.
1	---	vcc	——	——	Power supply of the printer

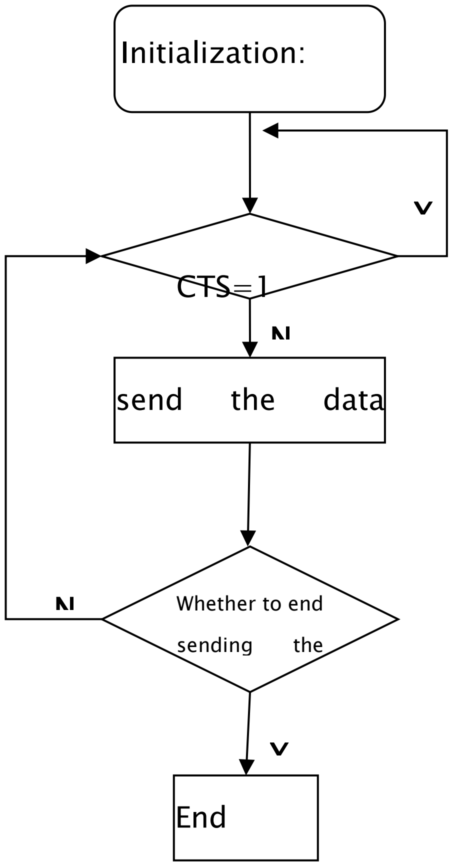
2.1.2Data transmission method of serial interface

The receiving buffer of the printer is 32K

(1) When the number of sending data once is less than 32K, the data can be directly sent. And the sending method is as follows:



(2) If large amounts of data once is sent, need to judge the mark 'CTS' when sending the data. When the mark is '1', the data can't be sent. When the mark is '0', the data can be sent. Data can be sent in the form of packets or byte. When the data is sent in the form of packets, each data packet can't exceed 256 bytes, and the sending flowchart is as follows:



Chapter 3: Command system

3.1 Command list

RD-DG Series thermal printers use the ESC / POS compatible command, and add tosome functions such as the Chinese characters printing, Character and Chinese characters rotation, word spacing adjustment, printing barcodes and other functions.

Command	Function
ESC c	To allow/ban reverse printing
FF	Print and feed paper to the top of the next page (only the models with the black mark detection function)
HT	To execute horizontal tab
LF	To print and line feed
CR	To print and carriage return
ESC SP	To set the character spacing
ESC $	To set printing absolute position
ESC ‘	To print curve
ESC v	Send the printer’s status to the host
FS &	Select the Kanji mode

FS .	Cancel the Kanji mode
ESC +	To allow/ban the overline printing
ESC –	To allow/ban the underline printing (to set/clear the underline mode)
ESC 1	To set the line spacing
ESC 6	To select Character Set 1 （6X8）
ESC 7	To select Character Set 2 （6X8）
ESC a	Select alignment methods
ESC @	To initialize the printer
ESC D	To set the position of horizontal tab
GS k	To print the bar code
ESC J	To print and feed paper
ESC d	To print and feed paper n lines
ESC K	Printing graphics command ①
ESC *	Printing graphics command ②
ESC Q	To set the right margin width
ESC U	Horizontally magnify character
ESC V	Vertically magnify character
GS H	Select the printing position for bar code character
GS Q	Set horizontal printing position for the bar code
GS k	Print the bar code(s)
ESC X	Magnify characters

GS h	Select bar code's height
GS w	Select bar code's width
GS B	To allow/ban white reverse printing mode
ESC I	To set the printing position
FS 2	To set character rotation Printing
FS r	To select the superscript and subscript

This chapter describes the commands of controlling the printer to print. Format specification is as follows:

【COMMAND】+【*parameter*】

1) 【COMMAND】 is the command, and consists of the escape character and command characters. But a small number of single-byte commands don't have the escape character.
2) 【*parameter*】 is the parameter, which is in italics. And the parameters are not numeric characters, but the value of the character.

All the examples in this chapter are compiled in C language. The 'SendDataToPrinter' function is a virtual functions. And require developers to write according to the actual situation of the mainframe.

This function is defined as follows:

SendDataToPrinter(unsigned char *buffer, unsigned int len) // Sending data to the printer

Unsigned char *buf // Print data command

Unsigned int len// Data length. Unit: byte.

3.2 Command Details

ESC @

[Name] To initialize the printer

[Type] ASCII: ESC @

Decimal: 27 64

Hex: 1B 40

[Explanation] Clear the data in the print buffer, and reset the printing parameters to the current printer default parameters.

[Note]

•The data in the receive buffer is not cleared.

[Example] unsigned char str[2];

```
str[0] = 0x1B;
str[1] = 0x40;
SendDataToPrinter(str,2);
```

FF

[Name] Print and feed paper to the top of the next page

[Type] ASCII: FF

Decimal: 12

Hex: 0C

[Explanation] Print all data in the printing buffer and return to the standard mode

[Note] If the paper has pre-printed black mark, take the paper to the black mark after printing the data in the data buffer. if the paper does not have black mark, feed paper 30cm. Pre-printed black mark specifications, see Appendix C. Pre-black Label printing instructions.

[Example] unsigned char str[2];

```
str[0] = 0x0C;
SendDataToPrinter(str,1);
```

LF

[Name] print and feed line

[Type] ASCII: LF

Decimal: 10

Hex: 0A

[Explanation] Print the data in the print buffer and feed one line

[Note] The command sets the print position to the beginning of the line

[Example] unsigned char str[2];

str[0] = 0x0A;//或str[0] = '\n'
SendDataToPrinter(str,1);

CR

[Name] print and carriage return

[Type] ASCII: CR

Decimal: 13

Hex: 0D

[Explanation] Print the data in the print buffer and carriage return

[Reference] LF

[Example] unsigned char str[2];

str[0] = 0x0D;//或str[0] = '\r'
SendDataToPrinter(str,1);

ESC J

[Name] print and feed paper

[Type] ASCII: ESC J n

Decimal: 27 74 n

Hex: 1B 4A n

[Explanation]Print the data in the print buffer and feeds forward paper [*n* x 0.125mm(0.0049")].

[Comment]

• After printing is finished, the command sets the print starting position to the beginning of the line.

[Scope] $0 \leq n \leq 255$

[Example] unsigned char str[3];

```
str[0] = 0x1B;
str[1] = 0x4A;
str[2] = 0x4;
SendDataToPrinter(str,3);// feeds forward paper 0.5mm
```

ESC d n

[Name] print and feed paper n lines

[Type] ASCII: ESC d n

Decimal: 27 100 n

Hex: 1B 64 n

[Scope] $0 \leq n \leq 255$

[Explanation] Print the data in the print buffer and feed paper n lines

[Comment]

• After finishing the printing, this command sets the print starting position to the beginning of the line.

• One line distance is 24 vertical pitch (0.125mm)

[Example] unsigned char str[3];

```
str[0] = 0x1B;
str[1] = 0x64;
str[2] = 0x4;
SendDataToPrinter(str,3);// feed forward paper 4 lines
```

ESC c

[Name] To allow/ban reverse printing

[Type] ASCII: ESC C n

Decimal: 27 99 n

Hex: 1B 63 n

[Scope] 0 ≤ *n* ≤ 1

[Explanation]

When n=1, allow the reverse printing and the printing direction is from left to right.

When n=0, ban the reverse printing and the printing direction is from right to left.

[Comment]

When the printer is vertically installed, the printer uses the reverse printing way.

Reversely printing not only supports character mode, and also supports graphical mode.

When reversely printing graphics, we should note the printing order of graphics unit. (See the ESC K command)

[Example] unsigned char str[3];

```
str[0] = 0x1B;
str[1] = 0x63;
str[2] = 0x1
SendDataToPrinter(str,3);// reverse printing
```

HT

[Name] horizontal tab

[Type] ASCII: HT

Decimal: 9

Hex: 09

[Explanation]Move the print position to the next horizontal tab position

[Note] • The command is ignored unless the next horizontal tab position has been set.

• Horizontal tab positions are set with the 'ESC D'.

[Reference] ESC D

ESC D n1 n2 … nk NULL

[Name] To set the position of horizontal tab

[Type] ASCII: ESC D *n1…nk NULL*

Decimal: 27 68 *n1…nk 0*

Hex: 1B 44 *n1…nk 00*

[Scope] $1 \leqslant n \leqslant 255$ $0 \leqslant k \leqslant 20$

[Explanation] Set the position of horizontal tab

n specifies the column number for setting a horizontal tab position from the beginning of a line.

k indicates the total number of horizontal tab positions to be set.

[Note]

• The horizontal position is stored as a value of [character width × n]measured from the beginning of the line. The character width includes the default width of the characters' spacing.

• This command deletes the previously set level positioning location.

• When n = 8, the printing position is moved to the 9th column by sending HT.

• The command is not affected by the ESC X command.

• This command cancels the previous tabulator position settings.

• The character printing position ,which exceeds the positioning location, will be processed as normal data.

• Transmit [n] k in ascending order and place a NULL code 0 at the end.

• When nk is less than or equal to the preceding value n (k-1),tab setting is finished and the following data is processed as normal data.

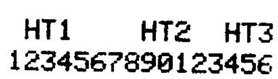
• ESC D NULL cancels all horizontal tab position.

• Even if the character width changes, previously specified horizontal tab positions don't also change.

[Default]The default tab positions are Font A (12*24).

[Example] unsigned str[8];

```
unsigned char Order = 9;
str[0] = 0x1B;
str[1] = 0x44;
str[2] = 2;// one character spacing from the first column
str[3] = 9;// eight character spacing from the first column
str[4] = 14;// thirteen character spacing from the first column
str[5] = 0; // end
SendDataToPrinter (str,6)
SendDataToPrinter (&Order,1);
SendDataToPrinter ("HT1",3);
SendDataToPrinter (&Order,1);
SendDataToPrinter ("HT2",3);
SendDataToPrinter (&Order,1);
SendDataToPrinter ("HT3",3);
Order = 0x0D;
SendDataToPrinter (&Order,1);
SendDataToPrinter ("1234567890123456\r",17)
```



ESC – n

[Name] To select/cancel the underline mode

[Type] ASCII: ESC – *n*

Decimal: 27 45 *n*

Hex: 1B 2D *n*

[Explanation] *n* = 1, select the underline mode

n = 0, cancel the underline mode

[Note]

- Underline can't act in the rotation and reverse characters.
- This command only affects the English and Kanji characters.

[Default] *n* = 0.

[Example]
```
unsigned char str[3];
str[0] = 0x1B;
str[1] = 0x2D;
str[2] = 0x1;
SendDataToPrinter (str,3);// set the underline
```

ESC +

[Name] allow/ban the overline printing

[Type] ASCII: ESC + n

Decimal: 27 43 n

Hex: 1B 2B n

[Explanation]When n=1, allow the overline printing

When n=0, ban the overline printing

[Note]

- Overline can't act in the rotation and reverse characters.
- This command only affects the English and Kanji characters.

[Default] n = 0

[Example]

```
unsigned char str[3];
 str[0] = 0x1B;
 str[1] = 0x2B;
 str[2] = 0x1;
SendDataToPrinter (str,3);//     set the overline
```

GS B n

[Name] select/cancel white reverse printing mode

[Type] ASCII: GS B n

Decimal: 29 66 n

Hex: 1D 42 n

[Scope] 0 ≤n ≤ 255

[Explanation] select/cancel white reverse printing mode

- When the LSB of n is 0, cancel white/black reverse printing mode.
- When the LSB of n is 1, select white/black reverse printing mode.

[Comment]

- Only the lowest bit of n is valid.
- The command is valid for the built-in and user-defined characters.
- This command only affects the English and Kanji characters.

[Default] n=0

[Example] unsigned char str[3];

```
 str[0] = 0x1D;
 str[1] = 0x42;
 str[2] = 1;// set the white reverse printing mode
SendDataToPrinter(str, 3);
```

FS 2 n

[Name] set character rotation Printing

[Type] ASCII: FS 2 n

Decimal: 28 73 n

Hex: 1C 49 n

[Scope] $0 \leqslant n \leqslant 3$

[Explanation]The command can rotate the character. The value of **n** is as follows:

n (Decimal)	Counterclockwise rotation
0	Does not rotate
1	90 degrees (Counterclockwise rotation)
2	180 degrees (Counterclockwise rotation)
3	270 degrees (Counterclockwise rotation)

[Note] Under the 90 degrees or 270 degrees rotation mode, the character width and height magnification direction is opposite to the magnification direction of the general mode.

[Default] n=0

[Example] unsigned char str[3];

```
str[0] = 0x1C;
str[1] = 0x49;
str[2] = 1;//    set 90 degrees rotation
SendDataToPrinter(str, 3);
```

ESC $ nL nH

[Name] Set absolute print position

[Type] ASCII: ESC $ nL nH

Decimal: 27 36 nL nH

Hex: 1B 24 nL nH

[Scope] $0 \leq nL + (nH \times 256) < 384$

[Explanation] Set the distance from the beginning of the line to the position at which subsequent characters are to be printed.

The distance from the beginning of the line to the printing position is N horizontal dot pitch

The nL and nH are the low and high bit of double-byte unsigned integer N .N=nL + nH×256

[Comment]

- Settings outside the specified printable area are ignored.
- In mode 1, n <= 372; In mode 2, n <= 420

[Example] unsigned char str[4];

```
str[0] = 0x1B;
str[1] = 0x24;
str[2] = 32;//
SendDataToPrinter (str, 3); // Set the absolute position to 32 horizontal dot pitch from the left margin
```

ESC l n

[Name] set the left margin

[Type] ASCII: ESC l n

Decimal: 27 108 n

Hex: 1B 6C n

[Scope] $0 \leqslant n \leqslant 32$

[Explanation]

The left margin is the number of characters, which isn't printed on the left side of the printing paper.

The distance from the beginning of the line to the printing position is the width of n English characters.

[Comment]

• If the printing position is outside the printable area, the command is ignored.

•The width of the character includes the default character width of the character spacing

[Example] unsigned char str[4];

```
str[0] = 0x1B;
str[1] = 0x6C;
str[2] = 3;//
SendDataToPrinter (str, 3); // the left position is set to the width of 3 English characters from the left margin
```

ESC Q n

[Name] set the right margin

[Type] ASCII: ESC Q n

Decimal: 27 81 n

Hex: 1B 51 n

[Scope] $0 \leqslant n \leqslant 32$

[Explanation]The right margin is the number of characters, which isn't printed on the right side of the printing paper.

[Comment]

• If the printing position is outside the printable area, the command is ignored.

•The width of the character includes the default character width of the character spacing

[Example] unsigned char str[4];

```
str[0] = 0x1B;
str[1] = 0x51;
str[2] = 3;//
```

SendDataToPrinter (str, 3); // set the area of three characters' width to the unprintable area on the right side

ESC 1 n

[Name] set the line spacing

[Type] ASCII: ESC 1 n

Decimal: 27 49 n

Hex: 1B 31 n

[Scope] 0 ≤ n ≤ 255 (The default value of '**n**' is 3)

[Default] n=3

[Explanation]Set the line spacing to n vertical dot pitch

[Example] unsigned char str[4];

```
str[0] = 0x1B;
str[1] = 0x31;
str[2] = 8;
SendDataToPrinter(str,3);// Set the line spacing to 8 vertical dot pitch
```

ESC SP n

[Name] set the character spacing

[Type] ASCII: ESC SP n

Decimal: 27 32 n

Hex: 1B 20 n

[Scope] 0 ≤ n ≤ 255 (The default value of '**n**' is 0)

[Explanation] Set the character spacing to n horizontal dot pitch

[Example] unsigned char str[4];

```
str[0] = 0x1B;
```

```
str[1] = 0x20;
str[2] = 8;
SendDataToPrinter(str,3);// Set the character spacing to 8 horizontal dot pitch
```

ESC a n

[Name]Select justification methods

[Type] ASCII: ESC a n

Decimal: 27 97 n

Hex: 1B 61 n

[Scope] $0 \leqslant n \leqslant 2$

[Explanation] Aligns all the data in one line to the specified position.

n selects the justification as follows:

n	justification methods
0	Left justification
1	Centering
2	Right justification

[Comment]

- This command is only valid at the beginning of the line.

[Default] n=0

[Example]

```
unsigned char str[4];
str[0] = 0x1B;
str[1] = 0x61;
str[2] = 1;
SendDataToPrinter(str,3);// select the centering to print
```

FS r n

[Name] select the superscript and subscript

[Type] ASCII: FS r n

Decimal: 28 114 n

Hex: 1C 72 n

[Scope] $0 \leqslant n \leqslant 1$

[Explanation]

The value of **n**	Result
n=o	superscript
n=1	subscript

[Comment]

The command is effective for all characters (including English characters and Kanji)

The command is ignored if n is outside the defined scope

[Example] unsigned char str[3];

```
str[0] = 0x1C;
str[1] = 0x72;
str[2] = 0;
SendDataToPrinter(str,3);//
```

ESC U

[Name] Horizontally magnify characters

[Type] ASCII: ESC U n

Decimal: 27 85 n

Hex: 1B 55 n

[Scope] $0 \leqslant n \leqslant 8$

[Comment]

The command is effective for all characters (including English characters and Kanji)

The command is ignored if n is outside the defined scope

[Reference] ESC X

[Example] unsigned char str[4];

```
str[0] = 0x1B;
str[1] = 0x55;
str[2] = 2;
SendDataToPrinter(str,3);// Horizontally magnify 2 times
```

ESC V

[Name]Vertically magnify characters

[Type] ASCII:	ESC	V	n
Decimal:	27	86	n
Hex:	1B	56	n

[Scope] $0 \leqslant n \leqslant 8$

[Comment]

The command is effective for all characters (including English characters and Kanji)

The command is ignored if n is outside the defined scope

[Reference] ESC X

[Example] unsigned char str[4];

```
str[0] = 0x1B;
str[1] = 0x56;
str[2] = 2;
SendDataToPrinter(str,3);// Vertically magnify 2 times
```

ESC X

[Name] Magnify characters

[Type] ASCII: ESC X n1 n2

Decimal: 27 88 n1 n2

Hex: 1B 58 n1 n2

[Scope] 0 ≤ n ≤8 (1 ≤ *n1* horizontal times ≤ 8,1 ≤ *n2* vertical times ≤ 8)

[Comment]

The command is effective for all characters (including English characters and Kanji), except barcode reading characters.

The command is ignored if n is outside the defined scope.

Vertical direction is the paper feeding direction, and horizontal direction is vertical with the paper feeding direction. When character clockwise rotate 90 °, the relationship between the vertical direction and horizontal direction is reversed, that is to say, this command’s priority is lower than the FS 2. And when two commands is effective at the same time, the characters firstly rotate, then enlarge.

[Example]unsigned char str[4];

```
str[0] = 0x1B;
str[1] = 0x58;
str[2] = 2;
str[3] = 2;
SendDataToPrinter(str,4);// Vertically and horizontally magnify 2 times
```

ESC K nL nH d1 d2 ……dk

[Name] Printing graphics command ①

[Type] ASCII ESC K nL nH d1…dk

Decimal: 27 75 nL nH d1…dk

Hex: 1B 4B nL nH d1…dk

[Scope] 0 ≤nL ≤255

0 ≤nH ≤1
0 ≤d ≤255

[Explanation]

This command can only print the black/white bit-image whose height is 8 dots and width does not exceed the printable area.

The nL and nH are the low and high bit of double-byte unsigned integer N. They express the number of the dots of the bit-image on the horizontal direction.

[Reference] ESC *

[Comment]

· The graphics command is influenced by the character enlargement command.

· When using reverse printing mode, successively print each graphics unit according to the order of the graphics from bottom to up.

[Example] unsigned char str[30];

```
unsigned char i=0;
str[i++] = 0x1B; str[i++] = 0x4B;
str[i++] = 15; //print the graphics whose width is 15 dots
str[i++] = 0x7C; str[i++] = 0x44; str[i++] = 0x44; str[i++] = 0xFF;
str[i++] = 0x44; str[i++] = 0x44; str[i++] = 0x7C; str[i++] = 0x00;
str[i++] = 0x41; str[i++] = 0x62; str[i++] = 0x54; str[i++] = 0xC8;
str[i++] = 0x54; str[i++] = 0x62; str[i++] = 0x41; str[i++] = 0x0D;
SendDataToPrinter(str,i);//send the printing graphics command.
```

ESC * m nL nH d1…dk

[Name] Printing graphics command ②

[Type] ASCII ESC * m nL nH d1…dk

Decimal: 27 42 m nL nH d1…dk

Hex: 1B 2A m nL nH d1…dk

[Scope] m = 0, 1, 32, 33

0 ≤nL ≤255
0 ≤nH ≤1
0 ≤d ≤255

[Explanation]

This command can only print the black/white bit-image whose height is 8 dots or 24 dots and width does not exceed the printable area.

The parameter meaning is as follows:

Using the m to select the bit image modes, and the dots of the bit image in the horizontal direction are specified by the nL and Nh.

m	The number of vertical dots (height)	Double-width mode
0	8	Twice as width
1	8	single-width
32	24	Twice as width
33	24	single-width

The nL and nH are the low and high bit of double-byte unsigned integer N. They express the number of the dots of the bit-image on the horizontal direction.

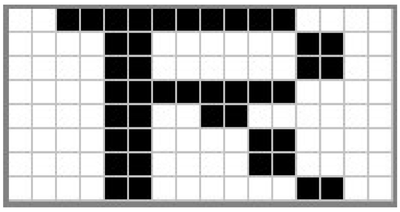
Mode 1: When the double-width mode is single-width, its maximum is 384. When the double-width mode is twice as width, its maximum is 192.

Mode 2: When the double-width mode is single-width, its maximum is 432. When the double-width mode is twice as width, its maximum is 216.

d1……dk express the bit-image data. And the specific format is as follows:

[Example 1] m =0 (8 dots, twice as width), d1 represents the data to be printed in the first and second column. And dk represents the data to be printed in the 2k[th] and (2k-1)[th] column. The bn represents the n[th] bit of the byte.

d1	d2	d3	d4	d5	d6	d7	d8	
0	1	1	1	1	1	0	0	b7
0	0	1	0	0	0	1	0	b6
0	0	1	0	0	0	1	0	b5
0	0	1	1	1	1	0	0	b4
0	0	1	0	1	0	0	0	b3
0	0	1	0	0	1	0	0	b2
0	0	1	0	0	1	0	0	b1
0	0	1	0	0	0	1	0	b0



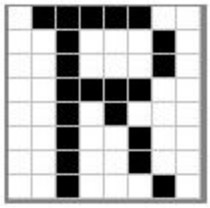
Program code is as follows:

```
unsigned char str[100];
j=0;
str [j++] = 0x1B;     str r[j++] = 0x2A;
str [j++] = 0; //m=0 (height is 8 dots, twice as width)
str [j++] = 8; //the width of the graphic is 8dots
str [j++] = 0;//the bit image data
str [j++] = 0x00;str [j++] = 0x80;str [j++] = 0xFF;str [j++] = 0x90;str [j++] = 0x98;
str [j++] = 0x96;str [j++] = 0x61;str [j++] = 0x00;str [j++] = 0x0D;//print the graphic
SendDataToPrinter(str,j);
```

[Example 2] m =1 (8 dots, single-width), d1 represents the data to be printed in the first column. And dk represents the data to be printed in the k[th] column. The bn represents the n[th] bit of the byte.

d 1	d 2	d 3	d 4	d 5	d 6	d 7	d 8	
0	1	1	1	1	1	0	0	b7

0	0	1	0	0	0	1	0	b6
0	0	1	0	0	0	1	0	b5
0	0	1	1	1	1	0	0	b4
0	0	1	0	1	0	0	0	b3
0	0	1	0	0	1	0	0	b2
0	0	1	0	0	1	0	0	b1
0	0	1	0	0	0	1	0	b0



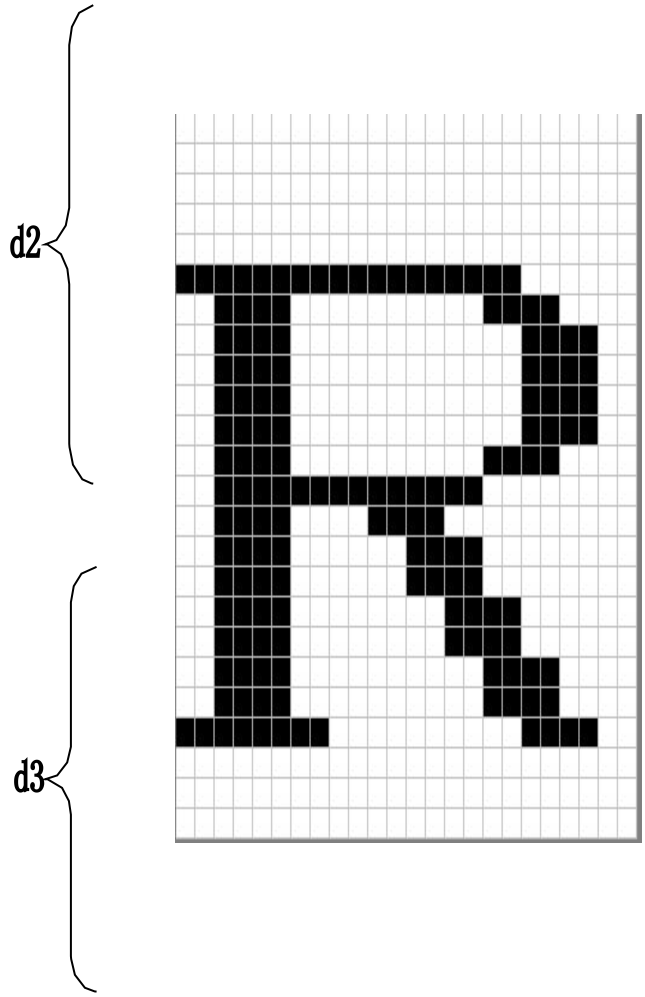
Program code is as follows:

```
unsigned char str[100];
j=0;
str[j++] = 0x1B;
str[j++] = 0x2A;
str[j++] = 1; //m=1(height is 8 dots, don't enlarge)
str [j++] = 8; //the graphic width is 8dots
str [j++] = 0;//bit image data
str[j++] = 0x00;str[j++] = 0x80;str [j++] = 0xFF;str[j++] = 0x90;str[j++] = 0x98;
strr[j++] = 0x96;str[j++] = 0x61;str[j++] = 0x00;str[j++] = 0x0D; ;//print the graphic
SendDataToPrinter(str,j);
```

[Example 3] m =32 (24 dots, twice as width), d1,d2 and d3 represent the data to be printed in the first, second and third column. And dk represents the data to be printed in the k^{th} column. The bn represents the n^{th} bit of the byte.

	d4	d7									D	d49
0	0	0	0	0	0	0	0	0	0	0	0	b7
0	0	0	0	0	0	0	0	0	0	0	0	b6
0	0	0	0	0	0	0	0	0	0	0	0	b5
1	1	1	1	1	1	1	1	1	0	0	0	b4
0	1	1	0	0	0	0	0	1	1	0	0	b3
0	1	1	0	0	0	0	0	0	1	1	0	b2
0	1	1	0	0	0	0	0	0	1	1	0	b1
0	1	1	0	0	0	0	0	0	1	1	0	b0
0	1	1	0	0	0	0	0	0	1	1	0	b7
0	1	1	0	0	0	0	0	1	1	0	0	b6
0	1	1	1	1	1	1	1	0	0	0	0	b5
0	1	1	0	0	1	1	0	0	0	0	0	b4
0	1	1	0	0	0	1	1	0	0	0	0	b3
0	1	1	0	0	0	1	1	0	0	0	0	b2
0	1	1	0	0	0	0	1	1	0	0	0	b1
0	1	1	0	0	0	0	1	1	0	0	0	b0
0	1	1	0	0	0	0	0	1	1	0	0	b7
0	1	1	0	0	0	0	0	1	1	0	0	b6

	1	1	1	1	0	0	0	0	0	1	1	1	b5
	0	0	0	0	0	0	0	0	0	0	0	0	b4
	0	0	0	0	0	0	0	0	0	0	0	0	b3
d1	0	0	0	0	0	0	0	0	0	0	0	0	b2
	0	0	0	0	0	0	0	0	0	0	0	0	b1
	0	0	0	0	0	0	0	0	0	0	0	0	b0



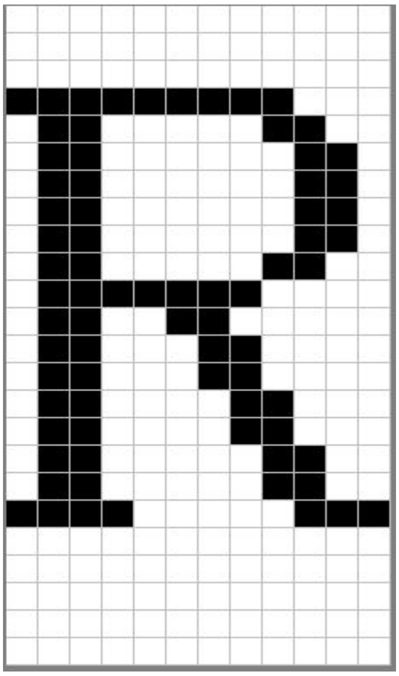
Program code is as follows:

```
unsigned char str[200];
j=0;
str[j++] = 0x1B;
```

```
str[j++] = 0x2A;
str[j++] = 32; //m=32(height is 24 dots, double-width)
str[j++] = 12; //graphic width is 12dots
str[j++] = 0;//bit image data
str[j++] = 0x10;str[j++] = 0x00;str[j++] = 0x20;str[j++] = 0x1F;str[j++] = 0xFF;str[j++] = 0xE0;
str[j++] = 0x1F;str[j++] = 0xFF;str[j++] = 0xE0;str[j++] = 0x10;str[j++] = 0x20;str[j++] = 0x20;
str[j++] = 0x10;str[j++] = 0x20;str[j++] = 0x00;str[j++] = 0x10;str[j++] = 0x30;str[j++] = 0x00;
str[j++] = 0x10;str[j++] = 0x3C;str[j++] = 0x00;str[j++] = 0x10;str[j++] = 0x2f;str[j++] = 0x00;
str[j++] = 0x18;str[j++] = 0x43;str[j++] = 0xC0;str[j++] = 0x0F;str[j++] = 0xC0;str[j++] = 0xE0;
str[j++] = 0x07;str[j++] = 0x80;str[j++] = 0x20;str[j++] = 0x00;str[j++] = 0x00;str[j++] = 0x20;
str[j++] = 0x0D;// Print out the current graphics
SendDataToPrinter(str,j);
```

[Example 4] m =33 (24 dots, don't enlarge), d1,d2 and d3 represent the data to be printed in the first, second and third column. And dk represents the data to be printed in the k^{th} column. The bn represents the n^{th} bit of the byte.

		d4	d7									D	d49
	0	0	0	0	0	0	0	0	0	0	0	0	b7
	0	0	0	0	0	0	0	0	0	0	0	0	b6
	0	0	0	0	0	0	0	0	0	0	0	0	b5
	1	1	1	1	1	1	1	1	1	0	0	0	b4
d1	0	1	1	0	0	0	0	0	1	1	0	0	b3
	0	1	1	0	0	0	0	0	0	1	1	0	b2
	0	1	1	0	0	0	0	0	0	1	1	0	b1
	0	1	1	0	0	0	0	0	0	1	1	0	b0
	0	1	1	0	0	0	0	0	0	1	1	0	b7
	0	1	1	0	0	0	0	0	1	1	0	0	b6
	0	1	1	1	1	1	1	1	0	0	0	0	b5
	0	1	1	0	0	1	1	0	0	0	0	0	b4
d2	0	1	1	0	0	0	1	1	0	0	0	0	b3
	0	1	1	0	0	0	1	1	0	0	0	0	b2
	0	1	1	0	0	0	0	1	1	0	0	0	b1
	0	1	1	0	0	0	0	1	1	0	0	0	b0
	0	1	1	0	0	0	0	0	1	1	0	0	b7
	0	1	1	0	0	0	0	0	1	1	0	0	b6
	1	1	1	1	0	0	0	0	0	1	1	1	b5
	0	0	0	0	0	0	0	0	0	0	0	0	b4
d3	0	0	0	0	0	0	0	0	0	0	0	0	b3
	0	0	0	0	0	0	0	0	0	0	0	0	b2
	0	0	0	0	0	0	0	0	0	0	0	0	b1
	0	0	0	0	0	0	0	0	0	0	0	0	b0



Program code is as follows:

```
unsigned char str[200];
j=0;
str[j++] = 0x1B;
str[j++] = 0x2A;
str[j++] = 32; //m=33 (height is 24 dots, don't enlarge)
str[j++] = 12; // graphic width is 12dots
str[j++] = 0;
// bit image data
str[j++] = 0x10;str[j++] = 0x00;str[j++] = 0x20;str[j++] = 0x1F;str[j++] = 0xFF;str[j++] = 0xE0;
str[j++] = 0x1F;str[j++] = 0xFF;str[j++] = 0xE0;str[j++] = 0x10;str[j++] = 0x20;str[j++] = 0x20;
str[j++] = 0x10;str[j++] = 0x20;str[j++] = 0x00;str[j++] = 0x10;str[j++] = 0x30;str[j++] = 0x00;
str[j++] = 0x10;str[j++] = 0x3C;str[j++] = 0x00;str[j++] = 0x10;str[j++] = 0x2f;str[j++] = 0x00;
str[j++] = 0x18;str[j++] = 0x43;str[j++] = 0xC0;str[j++] = 0x0F;str[j++] = 0xC0;str[j++] = 0xE0;
str[j++] = 0x07;str[j++] = 0x80;str[j++] = 0x20;str[j++] = 0x00;str[j++] = 0x00;str[j++] = 0x20;
str[j++] = 0x0D;// Print out the current graphics
SendDataToPrinter(str,j);
```

GS h n

[Name] Select bar code height

[Type] ASCII: GS h n

Decimal: 29 104 n

Hex: 1D 68 n

[Scope] $1 \leqslant n \leqslant 255$

[Explanation] Select bar code height. And N is the number of dots on the vertical direction.

[Default] n=48

[Example] unsigned char str[4];

```
str[0] = 0x1D;
str[1] = 0x68;
str[2] = 30;
SendDataToPrinter(str,3);//Set the bar code height to 30 vertical dot pitch
```

GS w n

[Name] Select bar code width

[Type] ASCII: GS w n

Decimal: 29 119 n

Hex: 1D 77 n

[Scope] 1≤ n ≤ 4

[Explanation] Set the horizontal width of the bar code.

And n specifies the bar code width as follows:

n	Module width for multi-level bar code （mm）	Binary-level bar code	
		Thin element width （mm）	Thick element width （mm）
1	0.125	0.125	0.25
2	0.25	0.25	0.50
3	0.375	0.375	0.75
4	0.50	0.50	1.0

[Example] unsigned char str[4];

```
str[0] = 0x1D;
str[1] = 0x77;
str[2] = 3;
SendDataToPrinter(str,3);//Set the bar code width
```

GS H n

[Name] Select the printing position for bar code character

[Type] ASCII: GS H n

Decimal: 29 72 n

Hex: 1D 48 n

[Scope] 0 ≤ n ≤ 2

[Explanation]Selects a font for the HRI characters used when printing a bar code.

Use n to specify the printing position of HRI:

n	printing position

0	Do not print
1	Above the bar code
2	Below the bar code

[Default] n=0

[Example] unsigned char str[4];

```
str[0] = 0x1D;
str[1] = 0x48;
str[2] = 2;
SendDataToPrinter(str,3);// The HRI is printed below the bar code
```

GS Q n

[Name] Set the printing position of the bar code on the horizontal direction

[Type] ASCII: GS Q *n*

Decimal: 29 81 *n*

Hex: 1D 51 *n*

[Scope] 0≤ n ≤ 255

[Explanation]Set the distance from the beginning of one line to the position of printing bar code to N horizontal dot pitch.

[Default] n=0

[Example] unsigned char str[4];

```
str[0] = 0x1D;
str[1] = 0x51;
str[2] = 32;
SendDataToPrinter(str,3);//
```

GS k

[Name] Print bar code

[Type] ASCII: ① GS k m d1...dk NUL ② GS k m n d1... dn

Decimal: 29 107 m d1...dk 0 29 107 m n d1... dn

Hex: 1D 6B m d1...dk 00 1D 6B m n d1... dn

[Scope] ① 0 ≤ m ≤ 6 ② 65 ≤ m ≤ 73

The range of k and d are determined by the type of bar code used.
The range of n and d are determined by the type of bar code used.
The n is the data length of the bar code to be printed.

[Explanation] Select a bar code system and print the bar code.

Use m to select a bar code system as follows:

m		Bar code system	length	scope
Format 1	0	UPC-A	11 k 12	48 d 57
	1	UPC-E	K	48 d 57
	2	JAN13 (EAN13)	12 k 13	48 d 57
	3	JAN 8 (EAN8)	7 k 8	48 d 57
	4	CODE39	1 k	48 d 57, 65 d 90, 32, 36, 37, 43, 45, 46, 47
	5	ITF	1 k (even number)	48 d 57
	6	CODABAR	1 k	48 d 57, 65 d 68 , 36, 43, 45, 46, 47, 58
Format 2	65	UPC-A	11 n 12	48 d 57
	66	UPC-E	n=8	48 d 57
	67	JAN13 (EAN13)	12 n 13	48 d 57
	68	JAN 8 (EAN8)	7 n 8	48 d 57
	69	CODE39	1 n 255	48 d 57, 65 d 90, 32, 36, 37, 43, 45, 46, 47
	70	ITF	1 n 255 (even number)	48 d 57
	71	CODABAR	1 n 255	48 d 57, 65 d 68, 36, 43, 45, 46, 47, 58
	72	CODE93	1 n 255	0 d 127
	73	CODE128	2 n 255	0 d 127

[Note]

• When using the format 1 command, if the bar code type specifies the data length of the bar code, k (the barcode data length received by the printer) should be equal to the specified data length, and if not equal to the specified data length, the instruction is invalid . See the related barcode data bit length [Appendix B].

• The barcode data received by the printer should be included in the character set specified by the bar code, if some characters of the bar code data characters are outside the character set, the command is invalid. See the related barcode character set [Appendix B].

• When using the format 2 command, the value of n should be equal to the specified data length (if the kind of bar code specifies the data bit length). And if the value of n is not equal to the specified data bit length, the command is invalid. See the related barcode data bit length [Appendix B].

•The number of ITF code data length must be even numbers. If using the format 1 to print ITF bar code, the value of k should be even numbers, but if it is odd number, the last one bit data will be ignored. If using the format 2 to print ITF bar code, the value of n should be even numbers, but if it is odd number, the last one bit data will be ignored.

•If the bar code on the horizontal direction exceeds the printable area, it is invalid.

•The command is not affected by the print modes (Eg: emphasized, double-strike print, underline, character size, or white/black reverse printing, etc.)

•Printing barcode need obey the barcode specifications, or will cause that the bar code cannot be scanned.

•The printer does not calculate the checksum, but if barcode needs the checksum, the checksum should be included in the bar code data, and the printer is not responsible for checking whether the checksum is wrong or right. The user calculates the checksum, and if it is wrong, it will cause that the bar code cannot be scanned.

•CODE39 code does not include the extended CODE39 code (EXTERN CODE 93).

•CODE93 code does not include the extended CODE93 code (EXTERN CODE 93).

•When using the CODE128, must first select the character set (CODE A, CODE B or CODE C) before the barcode data. Select the character set through sending the character "{" and another character; the ASCII code characters " {" is defined by sending " {" twice consecutively.

ASCII	HEX	Function
{A	7B, 41	Select the code set A
{B	7B, 42	Select the code set B
{C	7B, 43	Select the code set C
{S	7B, 53	SHIFT
{1	7B, 31	FNC1
{2	7B, 32	FNC2
{3	7B, 33	FNC3
{4	7B, 34	FNC4

ESC ‘

[Name] Print curve

[Type] ASCII: ESC ‘ nL nH x1L x1H x21L x21H .... xkL xkH CR

Decimal: 29 39 nL nH x1L x1H x21L x21H .... xkL xkH 13

Hex: 1B 27 nL nH x1L x1H x21L x21H .... xkL xkH 0D

[Scope] 0 ≤ nL ≤255

0 ≤ nHL ≤1

N is the number of the curve’s dots and N = nH x 256 + nL

The position of the curve’s dots in one horizontal line: X = xkH x 256 + xkL

[Explanation] Each curve consists of many dots. The command indicates that the printer prints n dots in one horizontal line, and continuously using the command can print out the curve which the user needs.

[Note] This command is only applicable to impact dot matrix printer and some thermal models.

[Example]

Print curve graphic of below five equations:

Y1＝50＋40*abs（-0.01*X）*sin（X/10）

Y2＝50-40*abs（-0.01*X）*sin（X/10）

Y3＝50

Y4＝50＋40*abs(-0.1*X)

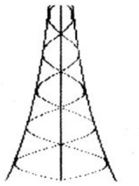
Y5＝50-40*abs(-0.01*X)

C program is as follows:

```
unsigned char str[50];
float X;
unsigned int   m_cur1,m_cur2,i;
for(X=0;X<150;X++)                                    //print one line of 150 dots
{
      m_cur1= 40*exp(-0.01*X);
      YY= Y*sin(X/10);
      str[i++] = 0x1b;
      str[i++] = 0x27;
      str[i++] = 0x5;//打印5条曲线
      str[i++] = 0x0;
      str[i++] = 50+m_cur2;
      str[i++] = 0;
      str[i++] = 50-m_cur2;
      str[i++] = 0;
      str[i++] = 50;
      str[i++] = 0;
      str[i++] = 50+m_cur1;
      str[i++] = 0;
```

```
        str[i++] = 50-m_cur1;
        str[i++] = 0;
        str[i++] = 0x0D;
        SendDataToPrinter(str,i);//
    }
```

[Print results]



ESC v

[Name] Send the printer's status to the host

[Type] ASCII： ESCv

Decimal: 27 118

Hex: 1B 76

[Explanation] Send the printer's status to the host

[Note] It is only effective for the printer with serial interface

Bit	Function	Value	
		0	1
0	Paper detector	No paper	Have paper
1	Work status	Idle	Printing
2	Receiving buffer	No full	Full
3	printer's status	Normal	Error
4	Undefined	——	——
5	Undefined	——	——
6	Undefined	——	——
7	Undefined	——	——

[Example]

```
unsigned char str[4];
str[0] = 0x1B;
str[1] = 0x76;
```

```
SendDataToPrinter(str,2);//Send status query command to the print
```

FS &

[Name] Select the Kanji mode

[Type] ASCII: FS &

Decimal: 28 38

Hex: 1C 26

[Explanation] The printer enters Kanji printing mode

[Note] After powering on the printer, the printer defaults the Kanji printing mode

[Example]

```
unsigned char str[4];
str[0] = 0x1C;
str[1] = 0x26;
SendDataToPrinter(str,2);// Enter Kanji printing mode
```

FS.

[Name] Cancel the Kanji mode

[Type] ASCII: FS .

Decimal: 28 46

Hex: 1C 2E

[Explanation] Cancel the Kanji characters mode

[Example] unsigned char str[4];

```
str[0] = 0x1C;
str[1] = 0x2E;
SendDataToPrinter(str,2);//Enter ASCII characters printing mode
```

ESC 6

[Name] To select Character Set 1 （6X8）

[Type] ASCII: ESC6

Decimal: 27 54

Hex: 1B 36

[Explanation] After inputting the command, all of printing characters use the characters in the character set 1 (see appendix D). The character set 1 has 224 ‘6 x 8 dot matrix’ characters, including ASCII characters and all kinds of graphic marks, etc. The range of code is 20H~FFH(32~255).

[Example] unsigned char str[4];

```
str[0] = 0x1B;
str[1] = 0x36;
SendDataToPrinter(str,2);//Print ‘6X8’ characters in the Character Set 1
```

ESC 7

[Name] To select Character Set 2（6X8）

[Type] ASCII: ESC 7

Decimal: 27 55

Hex: 1B 37

[Explanation] After inputting the command, all of printing characters use the characters in the character set 2 (see appendix D). The character set 2 has 224 ‘6 x 8 dot matrix’ characters, Including German, French, Russian, Japanese Katakana, etc. The range of code is 20H~FFH(32~255).

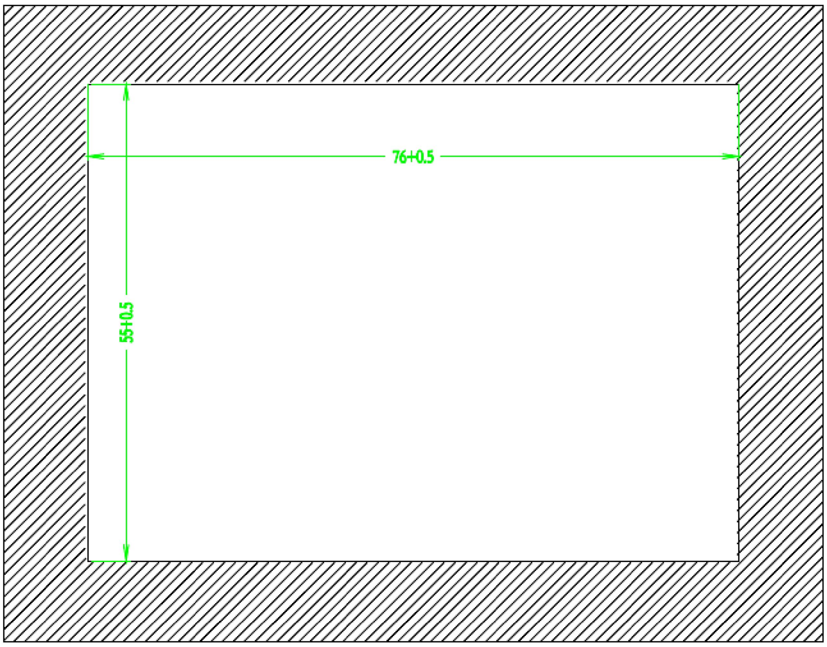
[Example] unsigned char str[4];

```
str[0] = 0x1B;
str[1] = 0x37;
SendDataToPrinter(str,2);// Print ‘6X8’ characters in the Character Set 2
```

Chapter 4: Installation

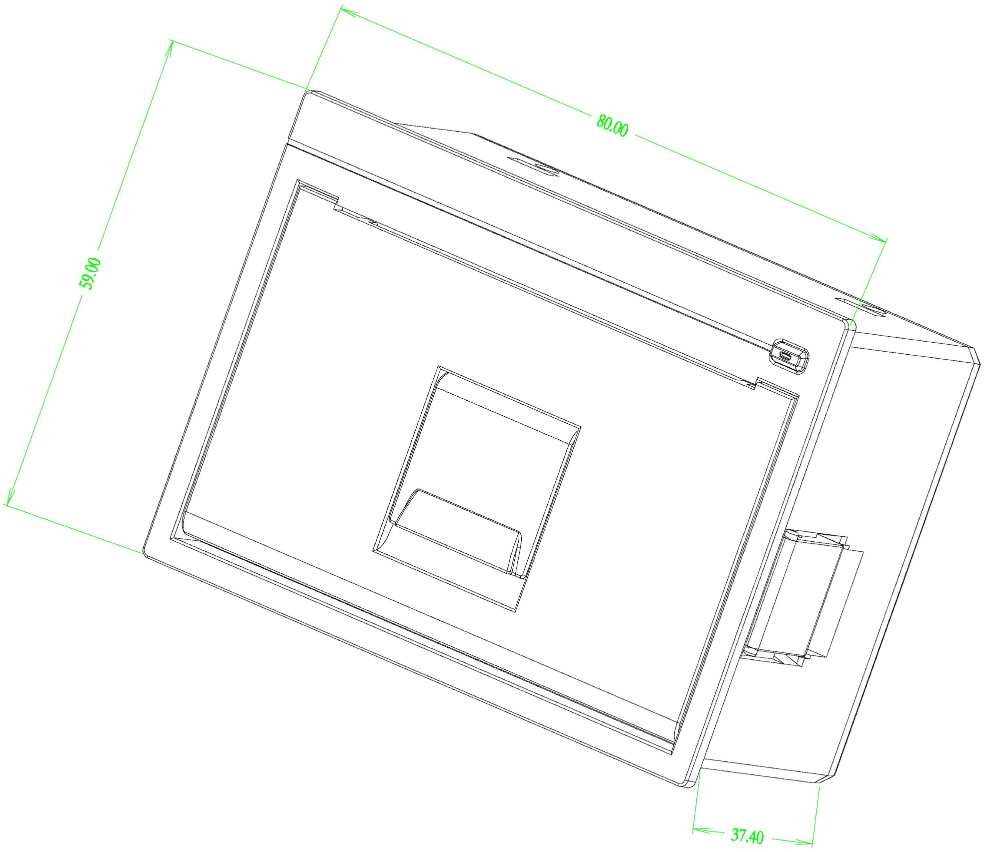
4.1 Installation dimension (Unit: mm)

Installation size: **(76mm*55mm)** (W * H)



4.2 Installation methods

External dimension: （**80mm*59mm*37.40mm**） (W * H * D)



Chapter5: Maintenance and Troubleshooting

To ensure the printer to work normally, particularly note that we don't optionally remove the print head and do not make changes to the printer through ourselves. For users not using the printer shell, more particularly note protecting the printing head.

1. If the printer is not used for a long time, we do not turn on the printer power.
2. If the printer is not working properly, please turn off the printer's power.
3. Power supply must meet the requirements, or it is unfavorable for the printing head, and even damages the printing head.
4. When replacing the paper roll, please note whether there are the paper scraps and dust on the printing head. If having paper scraps and dust, please gently remove. Note the thermal paper's obverse and reverse side, and if the reverse side is uncoated, the printer can't print out the handwriting.
5. When the printer is printing or paper feeding, we can't tear the paper, and can't more reversely drag the paper
6. Keep the printer control panel clean

7. When thermal printer prints unclearly, we can use the clean cotton ball soaked some alcohol to gently wipe the surface dirt on the print head chip heating element.
8. When we connect the printer to the host, we should connect the printer data cable, and then power on the printer.
9. To choose a good quality paper when we select the paper for the thermal printer can not only improve the printing quality, but also reduce the abrasion for thermal film.

APPENDIX

A : printing character set

HE		HE		HE		HE		HE		HE		HE		HE	
20	(空	21	!	22	“	23	#	24	$	25	%	26	&	27	‘

28	(	29	)	2A	*	2B	+	2C	,	2D	-	2E	.	2F	/
30	0	31	1	32	2	33	3	34	4	35	5	36	6	37	7
38	8	39	9	3A	:	3B	;	3C	<	3D	=	3E	>	3F	?
40	@	41	A	42	B	43	C	44	D	45	E	46	F	47	G
48	H	49	I	4A	J	4B	K	4C	L	4D	M	4E	N	4F	O
50	P	51	Q	52	R	53	S	54	T	55	U	56	V	57	W
58	X	59	Y	5A	Z	5B	[	5C	\	5D	]	5E	^	5F	
60	`	61	a	62	b	63	c	64	d	65	e	66	f	67	g
68	h	69	i	6A	j	6B	k	6C	l	6D	m	6E	n	6F	o
70	p	71	q	72	r	73	s	74	t	75	u	76	v	77	w
78	x	79	y	7A	z	7B	{	7C	|	7D	}	7E	~	7F	
80	Ç	81	ü	82	é	83	â	84	ä	85	à	86	å	87	ç
88	ê	89	ë	8A	è	8B	ï	8C	î	8D	ì	8E	Ä	8F	Å
90	É	91	æ	92	Æ	93	ô	94	ö	95	ò	96	û	97	ù
98	ÿ	99	Ö	9A	Ü	9B	¢	9C	£	9D	¥	9E	Pts	9F	ƒ
A0	á	A1	í	A2	ó	A3	ú	A4	ñ	A5	Ñ	A6	ª	A7	º
A8	¿	A9	⌐	AA	¬	AB	½	AC	¼	AD	¡	AE	«	AF	»
B0	░	B1	▒	B2	▓	B3	│	B4	┤	B5	╡	B6	╢	B7	╖
B8	╕	B9	╣	BA	║	BB	╗	BC	╝	BD	╜	BE	╛	BF	┐
C0	└	C1	┴	C2	┬	C3	├	C4	─	C5	┼	C6	╞	C7	╟
C8	╚	C9	╔	CA	╩	CB	╦	CC	╠	CD	═	CE	╬	CF	╧
D0	╨	D1	╤	D2	╥	D3	╙	D4	╘	D5	╒	D6	╓	D7	╫
D8	╪	D9	┘	DA	┌	DB	█	DC	▄	DD	▌	DE	▐	DF	▀
E0	α	E1	ß	E2	Γ	E3	π	E4	Σ	E5	σ	E6	µ	E7	τ
E8	Φ	E9	Θ	EA	Ω	EB	δ	EC	∞	ED	φ	EE	ε	EF	∩
F0	≡	F1	±	F2	≥	F3	≤	F4	⌠	F5	⌡	F6	÷	F7	≈
F8	°	F9	∙	FA	·	FB	√	FC	ⁿ	FD	²	FE	■	FF	

B: bar code

B.1 bar code coding rules

1. UPC-A: UPC-A coding should comply with the UCC Organization (http://www.uccnet.org) specification.

2. UPC-E: UPC-E coding should comply with the UCC Organization (http://www.uccnet.org) specification.

3. ENA8: ENA8 coding should comply with the ENA Organization (http://www.uccnet.org) specification.

4. ENA13: ENA13 coding should comply with the ENA Organization (http://www.uccnet.org) specification

5. CODE39: Also known as 39 codes, CODE39's starting bit and stopping bit characters must be the '*'character, and it cannot contain the characters '*' between the starting and stopping bits. And the printer's '*' is automatically given by the printer, and when programming the data need not be given, and the data can contain or cannot contain check code, and the check code have fixed algorithm.

6. ITF: Also known as INTERLEAVED 25, intersect 25 code, INTERLEAVED 2 of 5. The length for the data bit must be even number, and the data can contain or cannot contain check code, and the check code have fixed algorithm.

7. CODABAR: The starting bit and stopping bit must be anyone of A、B、C and D, and The starting bit character and stopping bit character can be different. The data can contain or cannot contain check code. The check code can be defined by the code man.

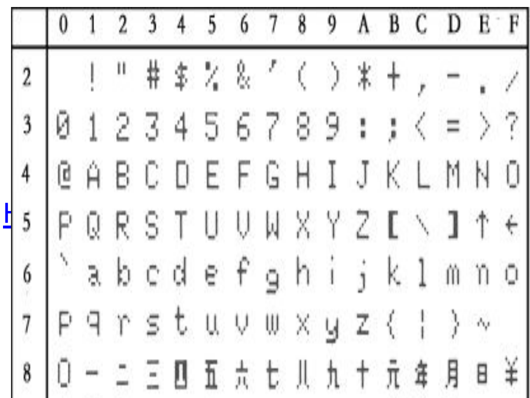
8. CODE93: CODE93's starting bit and stopping bit characters must be the '*'character, and it

cannot contain the characters '*' between the starting and stopping bits. And the printer's '*' is automatically given by the printer, and when programming the data need not be given, and the CODE93 data must contain two characters' check code, and the check code have fixed algorithm.

B.2 barcode length character set

Barcode Type	Length	character set (ASCII)
UPC-A	12	0~9
UPC-E	8	0~9
EAN8	8	0~9
EAN13	13	0~9
CODE39	27	0~9 A~Z - . SP $ / + % *
INTERLEAVED 25	even number 52	0~9
CODABAR	32	0~9 - : / % . A~D
CODE93	Unlimited	0~9 A~Z - . SP $ / + % *
CODE128	33	

C: Character Set 2



D: General Plan