MPT Series

Mobile Thermal Printer

Technical Manual

Content

Chapter 1 Brief Introduction

Based on the platform of ARM, our mobile thermal printers (MPT Series) are applicable for receipts and barcode labels, supporting wireless communication, such as infrared ray (IR), Bluetooth, and wifi (reserved) etc.

There are two models for MPT series: MPT-II (58mm paper width), MPT-III (80mm paper width).

Specification:

Item	MPT-II	MPT-III
Printing Method	Thermal Line	Thermal Line
Printing Speed	50～80mm/s	50～80mm/s
Effective Printing Width	48mm	72mm
Resolution	8 dots/mm, 384 dots/line	8 dots/mm, 576 dots/line
Print Font	ANK: 8x16, 12 x24 GBK:16x16, 24 x 24	ANK: 8x16, 12 x24 GBK:16x16, 24 x 24
Character per Line	32	48
Paper Type	Thermal paper	Thermal paper/lable paper
Paper Width	58mm	80mm
Paper Roll Diameter	Max: 30.0mm	Max: 40.0mm
Roll Core Inner Diameter	13mm(min.)	13mm(min.)
Paper Thickness	0.06 to 0.07mm	0.06 to 0.07mm
Paper Loading Method	Drop-in Easy loading	Drop-in Easy loading
TPH Reliability	50km	100km
Barcode	1-D:UPC-A, UPC-E, EAN-13, EAN-8, CODE39, ITF25, CODABAR, CODE93, CODE128 2-D:PDF417, QR code, DATA Matrix	1-D:UPC-A, UPC-E, EAN-13, EAN-8, CODE39, ITF25, CODABAR, CODE93, CODE128 2-D:PDF417, QR code, DATA Matrix
Emulation	ESC/POS	ESC/POS
Sensor	Paper end sensor, Power near end sensor, Cover open sensor	Paper end sensor, Power near end sensor, Cover open sensor
Interface	RS-232, USB, Bluetooth, IrDA, WIFI(reserved)	RS-232, USB, Bluetooth, IrDA, WIFI(reserved)
Power Adaptor	DC12V/500mAh	DC12V/1A
Battery	DC7.4V,1.5A rechargeable Li-ion battery	DC7.4V,1.5A rechargeable Li-ion battery

Battery Duration	150~200m continuous printing(12.5％density); 80~100m continuous printing(25％density)	150~200m continuous printing(12.5％density); 80~100m continuous printing(25％density)
Weight	205g	340g
Dimension (WxLxH)	92.5 x 75 x38mm	102 x 108 x50mm
Color	Dark Gray	Black
Operating Temp	-10~50°C	-10~50°C
Storage Temp	-20~70°C	-20~70°C
Operating Humidity	20~85%	20~85%
Storage Humidity	5~95%	5~95%

Chapter 2 Communication interface

MPT Series communicate with the host devices by five ways: RAW-IR, IrDA (IrCOMM), Bluetooth, RS232 interface and USB. Please make sure the host device supports at least one of these communication interfaces.

2.1 IR interface

IR is the wireless communication interface with low power consumption, sophisticated technology and easy using, is widely applicable to portable devices, such as notebook, mobile cell, palm computer, wince computer etc.

On hardware, MPT series meet the regulations of IrDA1.1 physical layer.

When MPT-II/ IR and MPT-III/IR communicate with host device, whose hardware also meets the regulation of IrDA 1.1 physical layer, please make the IR interfaces (both the printer and host device) right towards to each other (angle shall not be more than 30°), and no obstacles between them, the distance shall be less than 0.5M.

MPT-II/IR and MPT-III/IR can communicate with the host device not only by RAW-IR, also by IrCOMM protocol (IrDA).

When MPT-II/IR and MPT-III/IR communicate with the host device, IR transceiver just change the serial interface signal into IR signal or change light signal into serial interface signal according to the certain code. Under this situation, this IR interface is called RAW-IR. When you use RAW-IR, it is not compatible with IrDA for there is no software using IrDA protocol during receiving and sending data.

Compared with RAW-IR, IrCOMM, which is IR communication made by international IR Communication on the base of RAW IR hardware protocol, has the wider application. On the base of RAW-IR hardware, IrDA transfers data more stably and reliably and much easier. IrCOMM is the subset of IrDA. All portable devices supporting IrDA (such as IR cell phones with wince, pocket pc, palm OS etc.) all support IrCOMM. If the host devices (driving printer) use the operation system (such as wince, palm OS) supporting IrDA , for user, IrCOMM interface is the virtual serial port. If the host device has not IrDA protocol and wants to realize IrCOMM data transmission, the user has to compose IrDA protocol.

Please turn to supplier for more info and to make sure whether host device supports IrCOMM or not. RAW-IR interface and IrCOMM interface in MPT-II and MPT-III have the same hardware source, so they can't work at the same time. The original mode of MPT-II and MPT-III is IrCOMM. If you want to change IR interface mode, please refer to MPT-II/MPT-III——CONFIG modification setting. Details in 【MPT-II/MPT-III printing tool software】
Detailed IrDA protocol, please refer to IrDA official website: http://www.irda.org/

2.1.1 RAW-IR

User can learn RAW-IR interface password from host device manual and its supplier. RAW-IR interface is a little different from standard RS232 serial interface and RAW-IR interface has wireless connection and is valid for serial TXD signal and RXD signal. Other operation on other RS232 pin of host device has no effect to RAW-IR.
It is available to wake printer up by RAW-IR interface.
The baud rate of MPT series is: 9600bps, 19200bps, 38400bps, 57600bps, 115200bps. The default baud rate is 9600bps. If user wants to change baud rate, please refer to MPT-II/MPT-III_CONFIG modification setting.
Note: Before using, make sure whether RAW-IR of your host device can work under this baud rate. Otherwise please change baud rate.
When host device is on, set RAW-IR interface as follow:
Data bit: 8 bit
Stop bit: 1 bit
Parity check: No
Flow control: No
Compared with other general IR, RAW-IR has the anti-disturbance ability while please do not close to other IR sources, since printer can not shield these disturbances that other IR devices (such as notebook, cell phone with IR) send out infrared ray to search for other IR devices.

2.1.2 IRDA（ IrCOMM）

There are many IrDA protocols, and IrCOMM is one of IR communication protocols recommended by IrDA Association.
Using IrCOMM protocols, printer can totally avoid the interference from other IR.
MPT series support the IrCOMM protocol.
Although regard IrCOMM as virtual serial interface, there is no need to set baud rate, because real baud rate of data transmission will adapt automatically in IrCOMM. IrCOMM is responsible for verification and data buffer etc, so there is no sense for other IrCOMM serial interface setting.
User should know COM port password and how to open it. User can learn this info from host device manual or its suppliers.
It is available to wake the print up by IrCOMM interface.
If users want to develop the IrCOMM protocols, please refer to IrDA official website: http://

www.irda.ort.

2.2 Bluetooth interface

Bluetooth is the wireless technology supporting short distance communication (within 10m). It can transfer the data among many devices, such as mobile phone, PDA, mobile printer, notebook, wireless earphone etc. Bluetooth standard is IEEE802, 2.4 GHz, bandwidth 1Mb/s. For more details of Bluetooth, please refer to Bluetooth official website: http://www.bluetooth.org. MPT-II/BT and MPT-III/BT support wireless data transmission interface, meet Bluetooth 1.1 regulations, power class 2.

MPT-II/BT and MPT-III/BT are only sub-devices, and user needs to drive the printer by Bluetooth host devices (PDA, mobile phone, notebook). It won't drive the printer to print without other host devices.

MPT-II/BT and MPT-III/BT default device Name is MPT-II/BT and MPT-III/BT. User can change the device name according to your requirements. Please refer to【Chapter 3 MPT series printing tools software】.

Default Bluetooth connecting password is 0000. User can change the connecting name according to your requirements. Please refer to 【Chapter 3 MPT series printing tools software】.

2.2.1 Pairing

Before printing, MPT-II/BT and MPT-III/BT need pairing with main devices that drives the mobile thermal printer. Pairing process originates from main devices.

Pairing processes as follow:

1. Power on printer
2. Main device search outer Bluetooth device
3. If there are some outer Bluetooth devices, select **MPT-II or MPT-III**.
4. Enter password " **0000** ".
5. Finish pairing.

Detailed pairing methods, please refer to main devices Bluetooth Names. In pairing, mobile thermal printer must be turn on.

Note: when pairing, do not set many printers on, otherwise it can't figure out which one succeeds in pairing.

Having finished pairing, other main devices (main devices) still can pair with mobile thermal printer. The max main devices of each printer is 8 . If more than 8, the earliest ones will be deleted from printer paring list automatically. If you these earliest main devices want to drive printer to print and need pairing again.

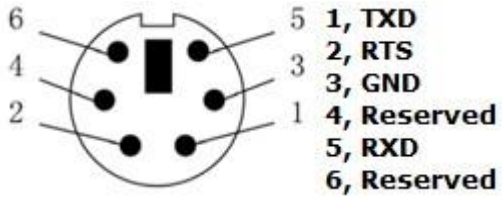
2.2.2 Printing by Bluetooth interface

After pairing, host devices with virtual Bluetooth interface（smart phone, pocket PC, palm, notebook）can drive printer through such virtual Bluetooth interface. If there is no virtual Bluetooth interface on host devices, please turn to these supplies for more information.

2.3 RS232 interface

RS232 is frequently-used interface. MPT series all have the RS232 serial interface.
Mobile thermal printer MPT-II RS232 interface specification:
Date transfer: serial
Synchronous mode: asynchronization
Hand-shaking signal: No
Flow control: hardware flow control / software flow control/ non optional
Baud rate: 9600bps、19200bps、38400bps、57600bps、115200bps optional.
Date byte length: 8 phase
Verification mode: NO
Stop bit: 1 phase
Socket pin definition (printer): 6 pin mini-din, as described in the following picture.

1. TXD
2. RTS
3. DND
4. Reserved pins
5. RXD
6. Reserved pins



Note: Reserved pins have other usages. Please do not use the reserved pins if you want to make the connection yourself, otherwise the printer can't work well or even cause loss to printer.
Buffer- area capacity of printer is about 3K bytes. When printing data is less than 3K byte, there is no need to use flow control, connection as follow:

Printer	Host device
RXD	TXD
GND	GND

When print date is more than 3K byte, flow control is needed. If connection is through Bluetooth and IrCOMM, there is no need to consider the flow control. No flow control in RAW-IR. When printer is on hardware flow control, connection between host device and printer as follow:

Printer	Host device
TXD	RXD
RTS	CTS
RXD	TXD
GND	GND

During communication, host device can monitor the signal intensity of CTS. When the signal intensity is strong, main device can send data to printer. On the contrary, it means the printer is busy and need to stop sending data. Until the signal intensity is strong, start sending data again.

When use software flow control with XON/XOFF method, connection as follow:

Printer	Main device
TXD	RXD
RXD	TXD
GND	GND

When use flow control, host device needs to test print buffer is full or not according to RXD data of main device itself. Detailed method as follow: Starting printing, main device send the data to the printer, and monitor data receiving of serial interface at the same time. When receiving XOFF (0x13), stop sending data to the printer; when receiving XON (0x11), send the data to printer again. It recycles until finishing printing.

2.4 WiFi (reserved)

Chapter 3 Command set

3.1 Basic Function Commands

3.1.1 ESC @

[Name] Initialize printer
[Format] ASCII ESC @
Hex 1B 40
Decimal 27 64
[Description] Initialize the printer. All settings, including character font and line spacing settings, are canceled. The data in the print buffer is cleared and the printer mode is reset to the mode that was in effect when the power was turned on. The DIP switch settings are not checked again, the data in the receive buffer is not cleared, and any macro definitions are not cleared.
[Program example] char SendStr[3];
SendStr[0] = 0x1B
SendStr[1] = 0x40;
PrtSendData(SendStr, 2);

3.1.2 FF

[Name] Print and return to standard mode (in page mode)
[Format] ASCII FF
Hex OC
Decimal 12
[Description] FF prints the data in the printer buffer collectively and returns to standard mode.
[Note]
• The buffer data is deleted after being printed.

• This command sets the print position to the beginning of the line.
• After printing data in print buffer, printer stops when meeting testing mark if there is the detecting marks on paper. If no detecting mark, it stops after feeding 0.5m paper. Please refer to the Appendix C preprinting specification.
[Program example] char SendStr[2];
SendStr[0] = 0x0C;
PrtSendData(SendStr, 1);

3.1.3 LF

[Name] Print and line feed
[Format] ASCII LF
Hex 0A
Decimal 10
[Description] LF prints the data in the printer buffer and feeds one line.
[Note] This command sets the print position to the beginning of the line.
[Reference] **CR**
[Program example] char SendStr[2];
SendStr[0]='\n'; //C language'\n' feed line
PrtSendData(SendStr, 1);

3.1.4 CR

[Name] Print and carriage return
[Format] ASCII CR
Hex 0D
Decimal 13
[Description] When auto line feed is disabled, this command is ignored.
[Reference] **LF**

3.1.5 ESC J *n*

[Name] Print and feed paper
[Format] ASCII ESC J n
Hex 1B 4A n
Decimal 27 74 n
[Range] 0 ≤ *n* ≤ 255
[Description] Prints the data in the print buffer and feeds the paper n vertical motion units.
[Note]
• This command sets the print position to the beginning of the line after being printed.
•Vertical motion unit space is 0.125mm.
[Reference] **ESC d**
[Program example] char SendStr[4];
SendStr[0] = 0x1B;

SendStr[1] = ’J’;
SendStr[2] = 8;
PrtSendData(SendStr, 3);// feeding1mm

3.1.6 ESC d *n*

[Name] Print and feed *n* lines
[Format] ASCII ESC d n
Hex 1B 64 n
Decimal 27 100 n
[Range] 0 ≤ *n* ≤ 255
][Description] Prints the data in the buffer and feeds *n* lines
[Note] • This command sets the print position to the beginning of the line after being printed.
[Reference] **ESC J**
[Progam example] char SendStr[4];
SendStr[0] = 0x1B;
SendStr[1] = ’d’;
SendStr[2] = 2;
PrtSendData(SendStr, 3);//feeding 2 lines

3.1.7 HT

[Name] Horizontal tab
[Format] ASCII HT
Hex 09
Decimal 9
[Description] HT moves the print start position to the next horizontal tab
[Note]
• Set horizontal tab position by ESC D command ESCD command set horizontal tab position.
• This command is ignored unless the next horizontal tab position has been set.
• Set Horizontal tab default to 8 character width (9^{th}, 17^{th},25^{th} …….list) of character A (12 ? 24).
[Reference] **ESC D**
[Program example] char NextPos = 9;
PrtSendData(“commodity Name”,6);
PrtSendData(&NextPos,1);
PrtSendData(“unti price”,4);
PrtSendData(&NextPos,1);
PrtSendData(“quantity”,4);
PrtSendData(&NextPos,1);
PrtSendData(“amount”,4);

3.2 Character commands

3.2.1 ESC ! *n*

[Name] Select print mode

[Format] ASCII ESC _ *n*

Hex 1B 21 *n*

Decimal 27 33 *n*

[Range] 0 ≤ *n* ≤ 255

[Description] Select printing mode by defined parameter n . Parameter n defined as follow:

Bit	Value	Name
0	—	Undefined
1	—	Undefined
2	—	Undefined
3	0	Emphasized mode not selected
	1	Emphasized mode selected
4	0	Double-height mode not selected
	1	Double-height mode selected
5	0	Double-width mode selected
	1	Double-width mode not selected
6	—	Undefined
7	0	Underline mode not selected
	1	Underline mode selected

[Note]

• When select double-height and double –width mode, prints character that is forth as much.

• When underline mode is turned on, 90° clockwise-rotated characters and white/black reverse characters cannot be underlines.

• There are some characters with double height or higher height in a line, all characters align at reference axis

• ESC M sets character font. The last command received is valid.

• ESC E selects emphasized mode or not. The last command received is valid.

• ESC – selects underline mode or not. The last command received is valid.

• GS ! sets character size. The last command received is valid.

• This command is effective for all characters (except for HRI characters)

[Default] *n* = 0

[Reference] **ESC -, ESC E, GS !, ESC M**

[Program example] char SendStr[4];

SendStr[0] = 0x1B;

SendStr[1] = '!';

SendStr[2] = 0x28;// 00101000 double-width emphasized mode

PrtSendData(SendStr, 3);

3.2.2 GS ! *n*

[Name] Select character size

[Format] ASCII GS ! *n*

Hex 1D 21 *n*

Decimal 29 33 *n*

[Range] 0 ≤ *n* ≤ 255 (1 ≤vertical number of times normal font size ⩽ 8, 1 ≤ horizontal number of times normal font size⩽ 8)

[Description] The number 0~3 to select character height, The number 4~7 to select character width, as follow:

0	1	2	3	Height
0	0	0	0	1 times
1	0	0	0	2 times

4	5	6	7	Width
0	0	0	0	1 times
1	0	0	0	2 times

[Note]

• This is command is effective for all characters (except for HRI characters).

•If n is outside of the specified range, this command is ignored.

•Feeding paper in vertical direction and then all characters are 90° clockwise rotated. Vertical direction and horizontal direction reverse, and that means the priority of this command is lower than SC V. When these two commands are valid, character rotates first and then enlarges.

•Enlarge characters in the same line with different size, all characters align at reference axis.

• ESC ! sets character size. The last command received sets current mode.

[Default] *n* = 0

[Reference] **ESC !**

[Program example] char SendStr[4];

SendStr[0] = 0x1D;

SendStr[1] = '!';

SendStr[2] = 0x01;// 00000001 double height

PrtSendData(SendStr, 3);

3.2.3 ESC – *n*

[Name] Turn underline mode on or off

[Format] ASCII ESC – *n*

Hex 1B 2D *n*

Decimal 27 45 *n*

[Rang] 0 ≤ *n* ≤ 2

[Description] Base on following n value, turn underline mode on or off

n Decimal	Definition
0	Turn underline mode off
1	Turn underline on (one-dot width)

[Note]

• When underline mode is on, 90 clockwise rotated characters and white/black reverse characters cannot be underlined.
• Character size selection is not effective for underline width
• ESC ! turns underline on or off. The last command received is valid.
• This command is valid for all English and Chinese characters.
[Default] *n* = 0
[Reference] **ESC !**
[Program example] char SendStr[3];
SendStr[1] = '-';
SendStr[2] = 1;// uniline underline
PrtSendData(SendStr, 3);

3.2.4 ESC E *n*

[Name] Turn emphasized mode on / off
[Format]

ASCII	ESC	E	*n*
Hex	1B	45	*n*
Decimal	27	69	*n*

[Range] 0 ≤ *n* ≤ 255
[Description] Turn emphasized mode on/ off
When the LSB (least significant bit) of n is 1, emphasize mode is turned on;
when LSB is 0, emphasized mode is turned off.
[Note]
• LSB of n is allowed to be used.
• ESC ! turns emphasized mode on / off . The last command received is valid.
[Default] *n* = 0
[Reference] **ESC !, ESC G**
[Program example] char SendStr[3];
SendStr[0] = 0x1B;
SendStr[1] = 'E';
SendStr[2] = 1;// emphasized mode
PrtSendData(SendStr,3);

3.2.5 ESC G *n*

[Name] Turn double-strike mode on/off
[Format]

ASCII	ESC	G	*n*
Hex	1B	47	*n*
Decimal	27	71	*n*

[Range] 0 ≤ *n* ≤ 255
[Description] Turn double-strike mode on/off
When the LSB (least significant bit) of n is 1, double-strike mode is turned on;
When LSB is 0, double-strike mode is turned off.
[Note]

• Only LSB of n is allowed to be used.
•Output of printer appear the same in Double –strike mode and emphasized mode.
[Default] *n* = 0
[Reference] **ESC E, ESC !**
[Program example] char SendStr[3];
SendStr[0] = 0x1B;
SendStr[0] = 0x1B;
SendStr[1] = '-';
SendStr[2] = 1;/ uniline underline
PrtSendData(SendStr, 3);

3.2.6 GS B n

[Name] Turn white/black reverse printing mode on/off
[Format] ASCII GS B *n*
Hex 1D 42 *n*
Decimal 29 66 *n*
[Range] 0 ≤ *n* ≤ 255
[Description] Turn white/black reverse printing mode on/off.
When the LSB (least significant bit) of n is 0, white/ black reverse printing mode is turned on;
when LSB is 1, white/black reverse printing mode is turned off.
Only LSB of n is allowed to be used.
•This command is effective for all characters (except for HRI characters)
• In white/black reverse printing mode, characters are printed in white on a black background.
• This command is not effective for bitmap, user-defined bitmap, bar code, barcode graphic character and the space skipped by HT $ and ESC\.
• White /black reverse mode prior to underline mode. When select white/black reverse mode, underline mode is prohibited but not canceled even turn underline mode on.
[Default] *n* = 0
[Program example] char SendStr[3];
SendStr[0] = 0x1D;
SendStr[1] = 'B';
SendStr[2] = 1;// white/black reverse printing
PrtSendData(SendStr, 3);

3.2.7 ESC V n

[Name] Turn 90° clockwise rotation mode on/off
[Format] ASCII ESC V *n*
Hex 1B 56 *n*
Decimal 27 86 *n*
[Range] 0 ≤ *n* ≤ 3

[Description] Turn 90° clockwise rotation mode on/off

n Decimal	Definition
0	Turn 90° clockwise rotation mode off
1	turn 90° clockwise rotation mode
2	turn 180° clockwise rotation mode
3	turn 270° clockwise rotation mode

[Note]

• When turn underline mode, 90°clockwise-rotated characters, underline mode is canceled.

• Under 90° clockwise rotation mode, the direction of characters with double-width and double-width enlargement reverse to that of characters with double-width and double-width enlargement under general mode.

[Default] *n* = 0

[Reference] **ESC !, ESC –**

[Program example] char SendStr[3];

SendStr[0] = 0x1B;

SendStr[1] = 'V';

SendStr[2] = 2;// 180° rotation

PrtSendData(SendStr, 3);

3.3 Print position commands

3.3.1 ESC $ *nL nH*

[Name] Set absolute print position

[Format] ASCII ESC $ *nL nH*

Hex 1B 24 *nL nH*

Decimal 27 36 *nL nH*

[Range] 0 ≤ *nL* ≤ 255, 0 ≤ *nH* ≤ 255

[Description] Set the print starting position from the beginning of the line. From the beginning of the line to the printer starting position is about N horizontal motion units. nL and nH are the low order and high order of double-byte unsigned integer N.

.

[Note]

• If set print position exceeds printable area （N>384）, the value of printable area is N=384.

[Reference] **ESC **

[Program example] char SendStr[4];

SendStr[0] = 0x1B;

SendStr[1] = '$';

SendStr[2] = 24;//3×8＝24

PrtSendData(SendStr, 3); Set the absolute position 3 MM from the left margin (24 horizontal motion unit)

PrtSendData (Start printing from left margin 3 mm \n, 22)

3.3.2 ESC D n1 n2…nk NULL

[Name] Set horizontal tab positions

[Format]	ASCII	ESC	D	*n1…nk NULL*
	Hex	1B	44	*n1…nk 00*
	Decimal	27	68	*n1…nk 0*

[Range] 1 ≤ *n* ≤ 255 0 ≤ *k* ≤ 8

[Description] Set horizontal tap position.

n columns from the beginning of a line.

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

[Note]

• Tab position as the value storage, the value is n characters width, and measures from the beginning of line. Character width includes default character width between characters.

• Characters enlargement (ESC ! GS !) is not effective for this command.

• This command cancels any previous horizontal tab settings.

• Set n=8, through sending HT, print position is moved to the ninth list.

• Set 8 tap position (k=8). Date more than 8 tap position will be processed as general data.

• Transfer [n]k according to ascending order, put NULL code 0 at the end.

• In this command, nk >n(k-1), if nk ≦ n(k-1), data after n(k-1) processes as the general date processing after setting finished.

• **ESC D NULL** cancels horizontal tap position.

• Previous horizontal tap position stay the same even change of character width.

[Default] Default tap position is character A (12 ×) and character spacing 8 (list9, 17,25,…….). 24

[Reference] **HT**

[Program example]
```
char SendStr[16];
char NextPos = 9;
SendStr[0] = 0x1B;
SendStr[1] = 'D';
SendStr[2] = 11;// 10 charcter spacing  from first list
SendStr[3] = 17;// 16 charcter spacing  from first list
SendStr[4] = 23;// 22 charcter spacing  from first list
SendStr[5] = 29;// 28 charcter spacing  from first lsit
SendStr[6] = 0; // End
PrtSendData(SendStr,7)
PrtSendData("Name",4);
PrtSendData(&NextPos,1);
PrtSendData("Chinses",4);
PrtSendData(&NextPos,1);
PrtSendData("Mathematics",4);
PrtSendData(&NextPos,1);
```

PrtSendData(”Foreign language”,4);
PrtSendData(&NextPos,1);
PrtSendData(“Amount”,4);

3.3.3 ESC 2

[Name] Select default line spacing
[Format] ASCII ESC 2
Hex 1B 32
Decimal 27 50

[Description] Set default line spacing to 1mm (8 vertical motion units)
[Note] • This command is effective for line spacing between bit image and character.
[Reference] **ESC 3**
[Program example] char SendStr[4];
SendStr[0] = 0x1B;
SendStr[1] = ’2’;
PrtSendData(SendStr,2);

3.3.4 ESC 3 *n*

[Name] Set line spacing
[Format] ASCII ESC 3 *n*
Hex 1B 33 *n*
Decimal 27 51 *n*
[Range] 0 ≤ *n* ≤ 255
[Description] Sets the line spacing to n × (vertical motion unit)
[Note] • This command is effective for line spacing between image and characters.
[Default] *n* = 8
[Reference] **ESC 2**
[Program example] char SendStr[4];
SendStr[0] = 0x1B;
SendStr[1] = ’3’;
SendStr[2] = 16;
PrtSendData(SendStr,3);// set the line spacing to vertical motion 16 units (2mm).

3.3.5 ESC SP *n*

[Name] Set character spacing
[Format] ASCII ESC SP *n*
Hex 1B 20 *n*
Decimal 27 32 *n*
[Range] 0 ≤ *n* ≤ 255

[Description] Sets the right-side character spacing to n × (horizontal motion unit)
[Note]
• Under double-width mode, right-side character spacing is twice as much as that of normal characters spacing.
• This command is effective for all characters. (except for HRI characters.)
[Default] *n* = 0
[Program example] char SendStr[4];
SendStr[0] = 0x1B;
SendStr[1] = 0x20;
SendStr[2] = 8;

3.3.6 ESC a *n*

[Name] Select justification
[Format] ASCII ESC a *n*
Hex 1B 61 *n*
Decimal 27 97 *n*
[Range] 0 ≤ *n* ≤ 2
[Description] Align all the data in one line to position specified by n.
n value and definition

n	definition
0	Left justification
1	Center
2	Right justification

[Note]
• This command is enable only when processed at the beginning of a line.
• This command is effective in the printing area.
• This command execute the blank area justification by **HT, ESC $** or **ESC \ .**
[Default] *n* = 0
[Program example] char SendStr[4];
SendStr[0] = 0x1B;
SendStr[1] = 'a';
SendStr[2] = 1;
PrtSendData(SendStr,3);// Set horizontal justification to center.

3.3.7 GS L *nL nH*

[Name] Set left margin
[Format] ASCII GS L *nL nH*
Hex 1D 4C *nL nH*
Decimal 29 76 *nL nH*
[Range] 0 ≤ *nL* ≤ 255 0 ≤ *nH* ≤ 255
[Description] Set left margin to N horizontal motion units. nL is low-order byte and nH is high-order byte, of the integer with no mark and double bytes. N=nL+nH*256, left margin

is width between left side of printing area

[Note]

• This command is enabled only when processed at the beginning of a line.

• Max left margin is 336.If it Exceeds 336, will regards left margin as 336.

[Default] *nL* = 0, *nH* = 0

[Program example] char SendStr[4];

SendStr[0] = 0x1B;

SendStr[1] = 'L';

SendStr[2] = 16;

SendStr[3] = 0;

PrtSendData(SendStr,4);// Set left margin to 16 horizontal motion units (2mm).

3.4 Bit-image commands

3.4.1 ESC * m nL nH d1…dk

[Name] Select bit-image mode

[Format] ASCII ESC * *m nL nH d1…dk*

Hex 1B 2A *m nL nH d1…dk*

Decimal 27 42 *m nL nH d1…dk*

[Range] *m* = 0, 1, 32, 33

0 ≤ *nL* ≤ 255

0 ≤ *nH* ≤ 1

0 ≤ *d* ≤ 255

[Description] Printing height is 8 dots or 24 dots, width should not exceed printable area of black & white bitmap. Parameter definition as following:

Use m to select bitmap mode, nl and nH set the dots of horizontal direction of bitmap.

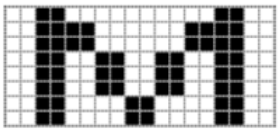
m	Vertical dots (Heights)	Double width mode
0	8	double width
1	8	Single width
32	24	double width
33	24	Single width

nL nH refer to the low – order and high-order byte of the integer with no mark and double bytes , means the dots in the horizontal direction in bitmap. Max value of single width is 384 and max value of double width is 192.

d1···dk indicates bitmap data, details as following figure.

[Program example] example 1. m=0 (8 dots, double width) d1 indicates the first and the second list data, dk indicates the $(2k-1)^{th}$ and $2k^{th}$ list data, bn indicates the n^{th} of byte.

d1	d2	d3	d4	d5	d6	d7	d8	d9	
0	1	0	0	0	0	0	1	0	b7
0	1	1	0	0	0	1	1	0	b6
0	1	1	0	0	0	1	1	0	b5
0	1	0	1	0	1	0	1	0	b4
0	1	0	1	0	1	0	1	0	b3
0	1	0	1	0	1	0	1	0	b2
0	1	0	0	1	0	0	1	0	b1
0	1	0	0	1	0	0	1	0	b0

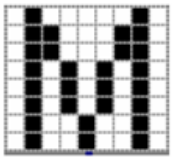


Print above image showed, program code as follows:

```
char SendStr[16];
SendStr[0] = 0x1B;
SendStr[1] = '*';
SendStr[2] = 0;//m=0 ( height 8 dots, double width)
SendStr[3] = 9;// ( image width 9 dotds)
SendStr[4] = 0;
SendStr[5] = 0;// (image dot line bitmap)
SendStr[6] = 0xFF;
SendStr[7] = 0x60;
SendStr[8] = 0x1C;
SendStr[9] = 0x03;
SendStr[10] = 0x1C;
SendStr[11] = 0x60;
SendStr[12] = 0xFF;
SendStr[13] = 0;
PrtSendData(SendStr,14);// print image
```

Example2: m=1 (8 dots, single width) d1 indicates the first list data, dk indicates the k^{th} list data, bn indicates the n^{th} of byte

d1	d2	d3	d4	d5	d6	d7	d8	d9	
0	1	0	0	0	0	0	1	0	b7
0	1	1	0	0	0	1	1	0	b6
0	1	1	0	0	0	1	1	0	b5
0	1	0	1	0	1	0	1	0	b4
0	1	0	1	0	1	0	1	0	b3
0	1	0	1	0	1	0	1	0	b2
0	1	0	0	1	0	0	1	0	b1
0	1	0	0	1	0	0	1	0	b0

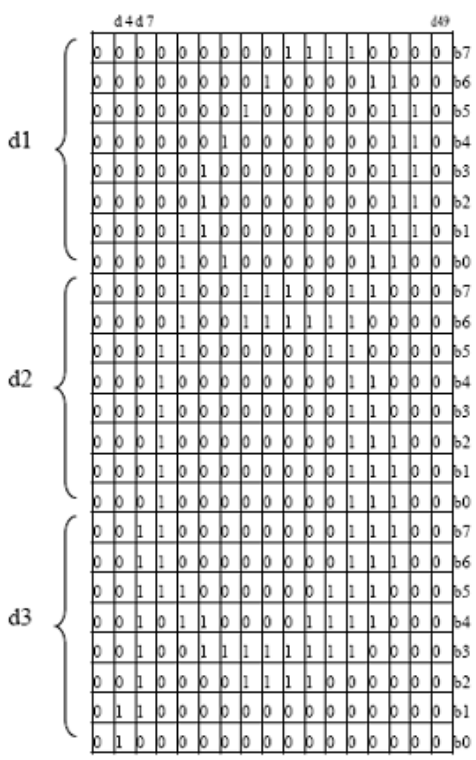


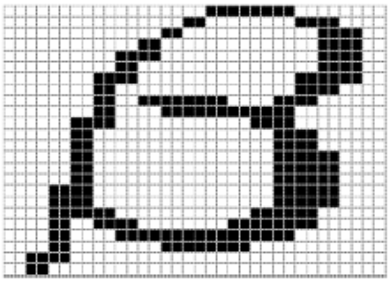
Print above image showed, program code as follows:

```
char SendStr[16];
SendStr[0] = 0x1B;
SendStr[1] = '*';
```

SendStr[2] = 1;//m=1 (height 8 dots, no enlargement)
SendStr[3] = 9;// 9dots image width 9 dots
SendStr[4] = 0;
SendStr[5] = 0;// image dot line bitmap
SendStr[6] = 0xFF;
SendStr[7] = 0x60;
SendStr[8] = 0x1C;
SendStr[9] = 0x03;
SendStr[10] = 0x1C;
SendStr[11] = 0x60;
SendStr[12] = 0xFF;
SendStr[13] = 0;
PrtSendData(SendStr,14);//print image

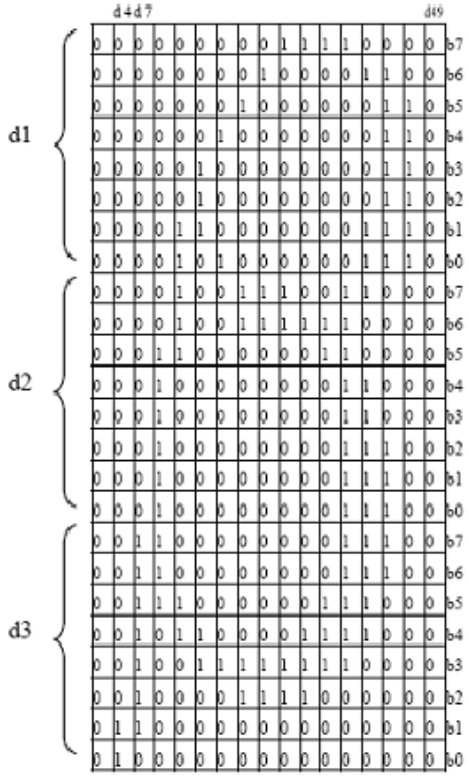
Example3: m=32 (24 dots, double width) d1,d2,d3 indicate the first, second and third list data, dk indicates the k^{th} list data, bn indicates the n^{th} of byte.

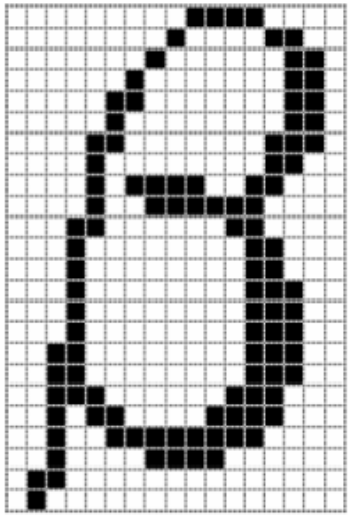




Print above image showed, program code as follows:
char SendStr[64];
SendStr[0] = 0x1B;
SendStr[1] = '*';
SendStr[2] = 32;//m=32 (Height 24 dots, double width)
SendStr[3] = 17;// 17dots image width 17 dots
SendStr[4] = 0;
// image data

SendStr[5] = 0x00; SendStr[6] = 0x00; SendStr[7] = 0x00;//1
SendStr[8] = 0x00; SendStr[9] = 0x00; SendStr[10] = 0x03;//2
SendStr[11] = 0x00; SendStr[12] = 0x00; SendStr[13] = 0xFE;//3
SendStr[14] = 0x00; SendStr[15] = 0x3F; SendStr[16] = 0xE0;//4
SendStr[17] = 0x03; SendStr[18] = 0xE0; SendStr[19] = 0x30;//5
SendStr[20] = 0x0E; SendStr[21] = 0x00; SendStr[22] = 0x18;//6
SendStr[23] = 0x11; SendStr[24] = 0x00; SendStr[25] = 0x08;//7
SendStr[26] = 0x20; SendStr[27] = 0xC0; SendStr[28] = 0x0C;//8
SendStr[29] = 0x40; SendStr[30] = 0xC0; SendStr[31] = 0x0C;//9
SendStr[32] = 0x80; SendStr[33] = 0xC0; SendStr[34] = 0x0C;//10
SendStr[35] = 0x80; SendStr[36] = 0x40; SendStr[37] = 0x1C;//11
SendStr[38] = 0x80; SendStr[39] = 0x60; SendStr[40] = 0x1C;//12
SendStr[41] = 0x80; SendStr[42] = 0xFF; SendStr[43] = 0xF8;//13
SendStr[44] = 0x43; SendStr[45] = 0x9F; SendStr[46] = 0xF0;//14
SendStr[47] = 0x7F; SendStr[48] = 0x07; SendStr[49] = 0xC0;//15
SendStr[50] = 0x3E; SendStr[51] = 0x00; SendStr[52] = 0x00;//16
SendStr[53] = 0x00; SendStr[54] = 0x00; SendStr[55] = 0x00;//17
PrtSendData(SendStr,56);//print image
m=33 (24 dot, singel width) d1、d2、d3 indicats the first list of priting data, so bn indicats the n^{th} of





Print above image showed, program code as follows:
char SendStr[64];
SendStr[0] = 0x1B;
SendStr[1] = '*';
SendStr[2] = 33;// m=33 (height 24 dots, no enlargement)

```
SendStr[3] = 17;// 17dots image width 17 dots
SendStr[4] = 0;
// image data
SendStr[5] = 0x00; SendStr[6] = 0x00; SendStr[7] = 0x00;//1
SendStr[8] = 0x00; SendStr[9] = 0x00; SendStr[10] = 0x03;//2
SendStr[11] = 0x00; SendStr[12] = 0x00; SendStr[13] = 0xFE;//3
SendStr[14] = 0x00; SendStr[15] = 0x3F; SendStr[16] = 0xE0;//4
SendStr[17] = 0x03; SendStr[18] = 0xE0; SendStr[19] = 0x30;//5
SendStr[20] = 0x0E; SendStr[21] = 0x00; SendStr[22] = 0x18;//6
SendStr[23] = 0x11; SendStr[24] = 0x00; SendStr[25] = 0x08;//7
SendStr[26] = 0x20; SendStr[27] = 0xC0; SendStr[28] = 0x0C;//8
SendStr[29] = 0x40; SendStr[30] = 0xC0; SendStr[31] = 0x0C;//9
SendStr[32] = 0x80; SendStr[33] = 0xC0; SendStr[34] = 0x0C;//10
SendStr[35] = 0x80; SendStr[36] = 0x40; SendStr[37] = 0x1C;//11
SendStr[38] = 0x80; SendStr[39] = 0x60; SendStr[40] = 0x1C;//12
SendStr[41] = 0x80; SendStr[42] = 0xFF; SendStr[43] = 0xF8;//13
SendStr[44] = 0x43; SendStr[45] = 0x9F; SendStr[46] = 0xF0;//14
SendStr[47] = 0x7F; SendStr[48] = 0x07; SendStr[49] = 0xC0;//15
SendStr[50] = 0x3E; SendStr[51] = 0x00; SendStr[52] = 0x00;//16
SendStr[53] = 0x00; SendStr[54] = 0x00; SendStr[55] = 0x00;//17
PrtSendData(SendStr,56);// Print image
```

[Note]

• If m exceeds set range, the unexpected results may appear.

• If bit image data exceeds set line dots, the exceeding data will be ignored.

• Printer returns to normal date procession mode after printing bit image.

• The print position is (emphasized mode, double strike mode, underline, character size, or white/black reverse printing mode) not effective for this command.

3.4.2 GS * *x y d1…dk*

[Name] Define downloaded bit image

[Format]	ASCII	GS	*	*x y d1…dk*
	Hex	1D	2A	*x y d1…dk*
	Decimal	29	42	*x y d1…dk*

[Range] 1 ≤ *x* ≤ 255

1 ≤ *y* ≤ 48

x * *y* ≤ 576

0 ≤ *d* ≤ 255

k=x*y*8

[Description] Use x and y dots to define downloaded bit image.

• *x*8 dots in the horizontal direction*

• *y*8 dots in the vertical direction*

[Note]

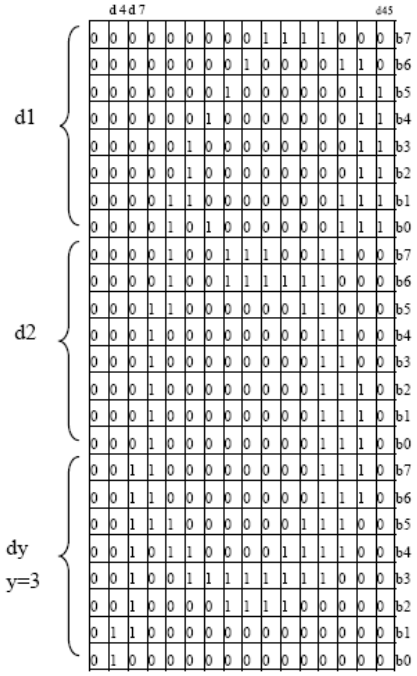
• If x ＊ y exceeds specified range, this command is prohibited.

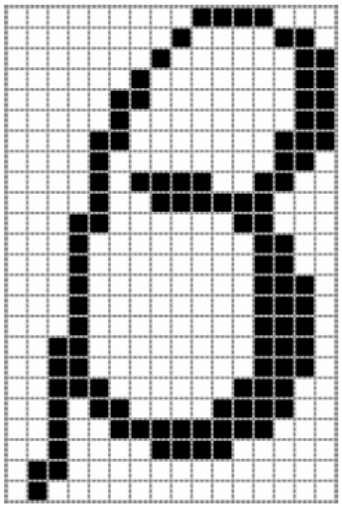
- d indicates bit image data. Data d set printing o, no printing 1.
- Bit image specified by this command print through the command GS/n.
- Clear the downloaded bit image under following cases;

1. Execute ESC@.
2. The printer is reset and the poser is turned off.

[Program example] The relation between downloaded bit image and printing date showed as follows:

If x=2, y=3, d1……dk in left picture, bitmap right picture





Print above image showed, program code as follows:

```
char SendStr[64];
SendStr[0] = 0x1D;
SendStr[1] = '*';
SendStr[2] = 2;//x=2 ( Width 16 dots)
SendStr[3] = 3;//y=3 ( Height 24 dots)
// Image data
SendStr[4] = 0x00; SendStr[5] = 0x00; SendStr[6] = 0x00;//1
SendStr[7] = 0x00; SendStr[8] = 0x00; SendStr[9] = 0x03;//2
SendStr[10] = 0x00; SendStr[11] = 0x00; SendStr[12] = 0xFE;//3
SendStr[13] = 0x00; SendStr[14] = 0x3F; SendStr[15] = 0xE0;//4
SendStr[16] = 0x03; SendStr[17] = 0xE0; SendStr[18] = 0x30;//5
SendStr[19] = 0x0E; SendStr[20] = 0x00; SendStr[21] = 0x18;//6
SendStr[22] = 0x11; SendStr[23] = 0x00; SendStr[24] = 0x08;//7
SendStr[25] = 0x20; SendStr[26] = 0xC0; SendStr[27] = 0x0C;//8
SendStr[28] = 0x40; SendStr[29] = 0xC0; SendStr[30] = 0x0C;//9
SendStr[31] = 0x80; SendStr[32] = 0xC0; SendStr[33] = 0x0C;//10
SendStr[34] = 0x80; SendStr[35] = 0x40; SendStr[36] = 0x1C;//11
SendStr[37] = 0x80; SendStr[38] = 0x60; SendStr[39] = 0x1C;//12
```

```
SendStr[40] = 0x80; SendStr[41] = 0xFF; SendStr[42] = 0xF8;//13
SendStr[43] = 0x43; SendStr[44] = 0x9F; SendStr[45] = 0xF0;//14
SendStr[46] = 0x7F; SendStr[47] = 0x07; SendStr[48] = 0xC0;//15
SendStr[49] = 0x3E; SendStr[50] = 0x00; SendStr[51] = 0x00;//16
PrtSendData(SendStr,52);/ defined image
SendStr[0] = 0x1D;
SendStr[1] = 0x2F;
SendStr[2] = 0x00;
PrtSendData(SendStr,3);// print image
```

3.4.3 GS / *n*

[Name] Print a downloaded bit image

[Format]

ASCII	GS	/	*n*
Hex	1D	2F	*n*
Decimal	29	47	*n*

[Range] 0 ≤ *n* ≤ 3

[Description] \Print a downloaded bit image using the mode specified by n.

The definition of n:

n	Enlargement
0	Normal
1	Double width
2	Double height
3	Double width double height

[Note]

• If bit image data is not defined, this command is ignored

• The print position is (emphasized mode, double strike mode, underline, character size, or white/black reverse printing mode) not effective for this command.

• If the downloaded bit image exceeds printable area, the exceeding data won't be printed.

• If print area specified by GSL is smaller than image width, it will minish left margin to print bit image.

• If bit image height exceeds 64 dots, this command is ignored. If printing with double height, bit image height shall not be more than 32 dots.

[Reference] **GS ***

[Program example] Please refer to 4.4.2 GS program example

3.5 Bar code commands

3.5.1 GS h *n*

[Name] Set bar code height

[Format]

ASCII	GS	h	*n*
Hex	1D	68	*n*

Decimal 29 104 *n*

[Range] 1 ≤ *n* ≤ 40

[Description] Select the height of a bar code.

n specified the number of dots in the vertical direction.

[Note]

• If n>40, bar code height will be set 40.

[Default] *n* = 36

[Reference] **GS k**

[Program example] Please refer to 4.5.5 GS k example.

3.5.2 GS w *n*

[Name] Set bar code width

[Format] ASCII GS w *n*

Hex 1D 77 *n*

Decimal 29 119 *n*

[Range] 1 ≤ *n* ≤ 4

[Description] Select the width of a bar code.

n definition as follow:

N	Module width (mm) for multilevel bar code	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

• Multilevel bar code as follows:

UPC-A, UPC-E, EAN13, EAN8, CODE93

• Binary level bar code as follows;

CODE39, CODABAR

[Default] *n* = 2

[Reference] **GS k**

[Program example] Please refer to 4.5.5 GS k program example

3.5.3 GS H *n*

[Name] Select printing position of HRI characters

[Format] ASCII GS H *n*

Hex 1D 48 *n*

Decimal 29 72 *n*

[Range] 0 ≤ *n* ≤ 2

[Description] Select the printing position for HRI characters when printing a bar code.

n select print position

n	select printing position
0	not printed
1	above the bar code
2	below the bar code

[Note]

• HRI characters are printed using the font specified by **GS f .**

• Print mode (emphasized mode, double strike mode, underline, character size, or white/black reverse printing mode etc.) is not effect for this command.

[Default] *n* = 0

[Reference] **GS f**, **GS k**

[Program example] Please refer to 4.5.5. GS k program example

3.5.4 GS k

[Name] print bar code

[Format] two formats in this command

Format 1（0 ≤ *m* ≤ 8）

ASCII GS k *m d1*…d*k NUL*

Hex 1D 6B *m d1*…d*k 00*

Decimal 29 107 *m d1*…d*k 0*

Format 2 (65 ≤ *m* ≤ 73)

ASCII GS k *m n d1*…*dn*

Hex 1D 6B *m n d1*…*dn*

Decimal 29 107 *m n d1*…*dn*

[Range] 0 ≤ *m* ≤ 8 (k and d specified by bar code system)

65 ≤ *m* ≤ 73 (n and d specified by bar code system)

n indicates the number of the bar code data.

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

m specified as a bar code system as follows:

m	Bar code system
0, 65	UPC-A
1, 66	UPC-E
2, 67	EAN8
3, 68	EAN13
4, 69	CODE39
5, 70	INTERLEAVED 25(ITF)

6, 71	CODABAR
7, 72	CODE93
8, 73	CODE128

[Note]

• When using format 1 command, if it specifies the length of barcode data in barcode system, then k (bar code data length printer received) should be in the range of specified length of bar code data), if not, the command will be invalid. The data bit length of the bar code, see Appendix B.

• The bar code data characters printer received should be included in the specified character set of the bar code system. If the bar code data characters beyond the character set, the command is invalid. The bar code character set, see Appendix A.

• When using format 2 command, the value of n should be equal to the specified bar code data length, if not, then the command is invalid. The data bit length of the barcode, see Appendix B.

• INTERLEAVED 25 (ITF) bar code data length is even, if you use format 1 to print ITF bar code, then k (bar code data length printer received) should be even, if odd, the last data will be ignored. If you use the format 2 to print ITF bar code, then the value for n should be even, if n is odd, then the last data will be ignored.

• If horizontal direction size exceeds print area, the exceeding part will be ignored.

• Print mode (emphasized mode, double strike mode, underline, character size, or white/black reverse printing mode etc.) is not effect for this command.

• Printing barcode should meet the barcode regulations, otherwise it may cause no scanning.

• The printer does not calculate parity check codes. If bar codes need parity check codes, then need including the check codes in bar code data. The printer is not responsible to check whether the parity check code is correct or not. Error parity check codes will cause bar code unable to scan.

• If the readable characters set to print, and the invisible characters in CODE93 and CODE128 codes can't be printed out, please use '□' for instead.

• CODE39 codes don't include the extension code (EXTERN CODE 39).

• CODE93 codes don't include the extension code (EXTERN CODE 93).

• The top of the bar code data string for CODE128 must be code set selection character (any of CODE A, CODE B or CODE C) which selects the first code set. Special characters are defined by combining two characters "{" and one character. The ASCII character "{" is defined by transmitting "{" twice consecutively.

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

[Program example] char SendStr[16];
SendStr[0] = 0x1D;
SendStr[1] = ‘h’;
SendStr[2] = 40;
PrtSendData(SendStr,3);//Set barcode height to 40 vertical dots(5mm)
SendStr[0] = 0x1D;
SendStr[1] = ‘w’;
SendStr[2] = 2;
PrtSendData(SendStr,3);//Set barcode width to 2
SendStr[0] = 0x1D;
SendStr[1] = ‘H’;
SendStr[2] = 2;
PrtSendData(SendStr,3);//Set the position of barcode readable character to below barcode
SendStr[0] = 0x1D;
SendStr[1] = ‘f’;
SendStr[2] = 0;
PrtSendData(SendStr,3);//Set the readable characters are12*24 characters
SendStr[0] = 0x1D;
SendStr[1] = ‘k’;
SendStr[2] = 4;
SendStr[3] = ‘*’; SendStr[4] = ‘T’; SendStr[5] = ‘E’ ;
SendStr[6] = ‘S’; SendStr[7] = ‘T’; SendStr[8] = ‘8’ ;
SendStr[9] = ‘0’; SendStr[10] = ‘5’; SendStr[11] = ‘2’ ;
SendStr[12] = ‘*’; SendStr[13] = 0;
PrtSendData(SendStr,14);//Use format 1 to print CODE39 code “TEST8052”

3.6 User-defined character commands

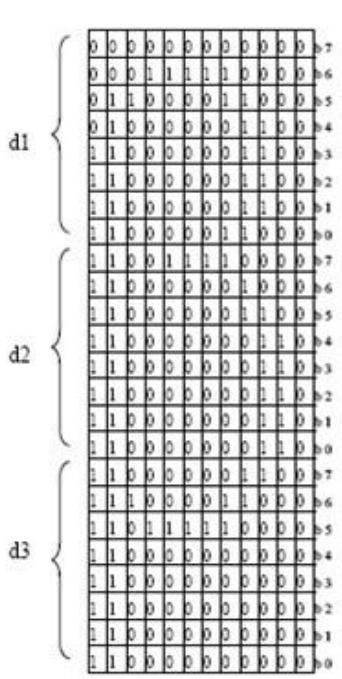
3.6.1 ESC % *n*

[Name] Select /cancel user-defined character set
[Format] ASCII ESC % *n*
Hex 1B 25 *n*
Decimal 27 37 *n*
[Range] 0 ≤ *n* ≤ 255
[Description] Select /cancel user-defined character set
When n LSB = 0, cancel user-defined character set.
When n LSB =1, select user-defined character set.
[Note]
• When user-defined character set is canceled, it selects internal character set.
• N is only enable when n = LSB
[Default] *n* = 0
[Reference] **ESC &**, **ESC?**

[Program example] char SendStr[4];
SendStr[0] = 0x1B;
SendStr[1] = '%';
SendStr[2] = 1;
PrtSendData(SendStr,3);// select user-defined character set

3.6.2 ESC & y c1 c2 [x1 d1...d(y * x1)]...[xk d1...d(y * xk)]

[Name] Define user-defined characters
[Format] ASCII ESC & *y c1 c2 [x1 d1...d(y * x1)]...[xk d1...d(y * xk)]*
Hex 1B 26 *y c1 c2 [x1 d1...d(y * x1)]...[xk d1...d(y * xk)]*
Decimal 27 38 *y c1 c2 [x1 d1...d(y * x1)]...[xk d1...d(y * xk)]*
[Range] *y* = 3 font A (12 ＊ 24)
y = 2 font B (8 ＊ 16)
32 ≤ *c1* ≤ *c2* ≤ 126
x = 12 font A (12 ＊ 24)
x = 8 font B (8 ＊ 16)
0 ≤ *d1 ... d(y ＊ xk)* ≤ 255
[Description] Define user-defined character
y specifies the number of bytes in the vertical direction.
define character A font y=3
define character B font y =2
• C1 specifies beginning character code, c2 specifies end character code.
• x specifies the number of dots in the horizontal direction.
define character A font x=12,
define character B font y=12
• *k=c2-c1+1*
[Note]
• From 0x20 to 0x7E ASCII (95 characters), the range of defined character code.
• define several character codes, if only one character code required, define c1 = c2.
• d is dot line data of character
• Define user-defined character data is (y * x) byte.
• Set 1 for print and 0 for not printed.
• This command is effect for user-defined character mode with different character definition. ESC ! sets character font.
• User-defined character and bitmap is not more than 32K.
• User-defined characters will be cleared under following cases:
1. Execute **ESC** @
2. Execute **ESC ?**
3. Printer is reset or the power is off.
[Program example] y=3, x=12 specifies character B (code 0x42)as character 12*24



char SendStr[16];
SendStr[0] = 0x1B;
SendStr[1] = '&';
SendStr[2] = 3; //y=3 24 character height 24
SendStr[3] = 0x42; //character ,'B' (code 0x42) redefined
SendStr[4] = x042;
SendStr[5] = 12; //x=12 character width 12
SendStr[6] = 0x0F; SendStr[7] = 0xFF; SendStr[8] = 0xFF;//1
SendStr[9] = 0x3F; SendStr[10] = 0xFF; SendStr[11] = 0xFF;//2
SendStr[12] = 0x20; SendStr[13] = 0x00; SendStr[14] = 0x40;//3
SendStr[15] = 0x40; SendStr[16] = 0x00; SendStr[17] = 0x20;//4
SendStr[18] = 0x40; SendStr[19] = 0x80; SendStr[20] = 0x20;//5
SendStr[21] = 0x40; SendStr[22] = 0x80; SendStr[23] = 0x20;//6
SendStr[24] = 0x40; SendStr[25] = 0x80; SendStr[26] = 0x20;//7
SendStr[27] = 0x61; SendStr[28] = 0x80; SendStr[29] = 0x60;//8
SendStr[30] = 0x3F; SendStr[31] = 0x60; SendStr[32] = 0xC0;//9
SendStr[33] = 0x1E; SendStr[34] = 0x3F; SendStr[35] = 0x80;//10
SendStr[36] = 0x00; SendStr[37] = 0x1F; SendStr[38] = 0x00;//11
SendStr[39] = 0x00; SendStr[40] = 0x00; SendStr[41] = 0x00;//12
PrtSendData(SendStr,42);// select user-deifned character set
[Default] internal character set
[Reference] **ESC %, ESC ?, FS 2**

3.6.3 ESC ?

[Name] Cancel user-defined characters
[Format] ASCII ESC ? *n*

Hex	1B	3F	*n*
Decimal	27	63	*n*

[Range] 32 ≤ *n* ≤ 126

[Description] Cancel user-defined characters

[Note]

• Cancel the user-defined characters defined for the character code n. After the user-defined characters are canceled, the internal character set is printed.

• If there is no specified character code for the user-defined character, the printer will ignore this command.

[Reference] **ESC &**, **ESC %**

[Program example] char SendStr[4];

SendStr[0] = 0x1B;

SendStr[1] = '?';

SendStr[2] = 0x42;

PrtSendData(SendStr,3);// Clear characters defined by 0x42 .

Appendix A Index of Print Codes

HEX		HEX		HEX		HEX		HEX		HEX		HEX		HEX	
20	Blank	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	c	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	β	E2	Γ	E3	Π	E4	Σ	E5	σ	E6	μ	E7	γ
E8	Φ	E9	θ	EA	Ω	EB	δ	EC	∞	ED	φ	EE	Є	EF	∩
F0	≡	F1	±	F2	≥	F3	≤	F4	⌠	F5	⌡	F6	÷	F7	≈
F8	°	F9	•	FA	·	FB	√	FC	ⁿ	FD	²	FE	▪	FF	

Appendix B Barcode

B.1 Barcode

UPC-A: UPC-A codes should comply with the regulations of UCC Organization (http://www.uccnet.org).
UPC-E: UPC-E codes should comply with the regulations of UCC Organization (http://www.uccnet.org).
ENA8: ENA8 codes should comply with the regulations of EAN Organization (http://www.ean-int.org).
ENA13: ENA13 codes should comply with the regulations of EAN Organization (http://www.ean-int.org).
CODE39: also known as 39 codes, the starting character and ending character of CODE39 must be '*', and among start bit and stop bit there can't contain the character '*', the data can include parity check codes, parity check codes have fixed algorithm.
ITF: also known as INTERLEAVED 25, cross 25 codes, INTERLEAVED 2 of 5, the length of data only can be even, the data can include parity check codes, parity check codes have fixed algorithm.
CODABAR: also known as Codabar. The start bit and stop bit must be one of A, B, C, D four characters. Stop bit doesn't need to be same with start bit. The data also can contain parity check codes, parity check codes are defined by the coder.
CODE93: the starting character and ending character of CODE93 must be '*', and among start bit and stop bit there can't contain the character '*', the ending data of CODE93 must include two characters of parity check codes, parity check codes have fixed algorithm.

B.2 Barcode length and ASCII Tab

Barcode Type	Length	ASCII
UPC-A	12	0~9
UPC-E	8	0~9
EAN8	8	0~9
EAN13	13	0~9
CODE39	No limit	0~9 A~Z - . SP $ / + % *
INTERLEAVED 25	even number	0~9
CODABAR	No limit	0~9 - : / % .

		A~D
CODE93	No limit	0~9 A~Z - . SP $ / + % *

Appendix C Black mark pre-printing specifications

If user uses ready printing black mark to localize the receipt, please be sure to abide by the following black mark ready printing regulation when print the black mark, otherwise the black mark can't be recognized by the printer. The black mark ready printing regulation:

Print position: as the following figure shows, the black mark should be printed on the right of the word side.

Width range: ≥7mm

Height range: 4mm≤height≤6mm

Reflectivity to infrared ray: <10% (the reflectivity of the other parts of the paper black mark width to infrared ray >65%)