Mobile thermal printer

MPT-II
Programming Manual

REVISION RECORDS

REV.	DATE	DESCRIPTION	DRAWN	CHECKED	APPROVED
1.0	2014.11.24	Preliminary	Cui Xiao	Yang Kaiqu	Ren Xiaowei
1.1	2015.05.12	1. Update bluetooth version and standard(Page 3)	Ji Qiuxia	Yang Kaiqu	Ren Xiaowei

Content

1. Brief introduction

On the platform of ARM, mobile thermal printer (MPT-II) is applicable for receipts and bar code labels, supporting the wireless communication, such as infared ray (IR), Bluetooth, Wi-Fi etc.

Specification:

Item		**Parameter**
		40mm paper house
Printing	Printing method	Thermal line printing
	Resolution	203dpi, 8dots/mm
	Printing speed	50~70mm/s
	Valid printing width	48mm
	Interface	Optional interfaces(Bluetooth,RS232,USB)
Power saving	Sleep mode	YES
Character Set	Font	ASCII: 8×16 9×24 12×24;Multiple code pages support
Barcode Symbologies	1D	UPC-A, UPC-E, EAN8, EAN13, CODE39, ITF, CODEBAR, CODE128, CODE93
	2D	QR code
Graphics		Support bitmap printing with different density and user defined bitmap printing (Max. 40K for per bitmap, and Max. 76K for total)
Detection	Sensors	Paper out detection
LED indicator	Power indicator	Red
	Error indicator	Blue
Power supply	Power supply	12V ⎓ 0.5A
	Battery	1500mAh 7.4V rechargeable Li-ion battery
Paper	Paper type	Thermal paper
	Recommended paper	FD210,PD150R,PD160R (OJI Paper CO.,LTD.)
	Paper width	58mm
	Paper thickness	0.06~0.07mm
	Paper roller diameter	≤40mm
	Paper loading	Easy loading mechanism
Physical characteristics	Working condition	0°C~40°C , 25%~80%RH
	Storage condition	-20°C~70°C,5%~95%RH
	Dimension	102.5*75*45mm
	Weight	200g (without paper roll)
Reliability	TPH	50km(not more than 12.5% printing density)/100 million pulses
Software	Emulation	ESC/POS
	Driver	Windows XP/Vista/Win7/Win8

2. Communication Interfaces

Mobile thermal printer communicates with the main devices by 4 ways: IrDA IrCOMM , RAWIR, VIR Bluetooth, and RS232 serial interface. MPT-II supports one or more communication interfaces and please make sure your main device supports at least one of these communication interfaces.

2.1 Infrared ray: IR interface

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

On hardware, MPT-II meets the regulations of IrDA1.1 physical layer.

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

MPT-II/IR communicates with main device according to Vir protocol or IrCOMM protocol (IrDA).

When MPT-II/IR communicates with main 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.

VIR is much safer and more reliable IR communication way, for there is the VIR protocol on the base of RAW-IR.

Compared with VIR protocol, IrCOMM, which is IR communication made by international IR Communication on the base of RAW IR hardware protocol, has the wide application. On the base of RAW-IR hardware, IrDA transfers data more stably and reliably and much easier. IrCOMM is the subset of IrDA. Wireless communication protocol developed by IrDA, All portable devices supporting IrDA (such as wince, pocket pc, palm, IR cell phones etc.) all support IrCOMM. If main device (driving printer) use the operation (such as winge, palmOS) supporting IrDA , for user, IrCOMM interface is the virtual serial port. If the main device has no IrDA protocol and want to realize IrCOMM data transmission, the user has to compose IrDA protocol. Please turn to supplier for more info and to make sure whether main device support IrCOMM or not. RAW-IR interface, VIR interface and IrCOMM interface in MPT-II have the same hardware source, so they can't work at the same time. The original mode of MPT-II is IrCOMM. If you want to change IR interface mode, please refer to MPTTools modification setting. Details in 【2.4 Software Tools of MPT-II】

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 main 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 Main device has no effect to RAW-IR.

Wake printer up by RAW-IR interface.

The baud rate of mobile thermal printer MPT-II/IR is: 9600bps，19200bps，38400bps，57600bps，

115200bps. Original baud rate of MPT-II/IR is 9600bps. If user wants to change baud rate, please refer to MPTTools modification setting.
Note: Before using, make sure whether RAW-IR of your main device can work under this baud rate. Otherwise please change baud rates.
When main device is on, set RAW-IR interface as follow:
Byte length: 8 phase
Stop bit: 1 phase
Even-odd check: No
Flow control: No
Compared with other general IR, RAW-IR has the anti-disturbance ability while do not close to other IR sources, for printer can not shield these disturbances that other IR devices (such as notebook, cellphone with IR) send out infrared ray to find other IR devices.

2.1.2 VIR (Developing)

For MPT-II better communication with hand terminals without IrDA reliably, we develop the VIR protocols on the base of printing data communication protocol that meets the regulation of IrDA physical layer.
Wake printer up by VIR protocol.
For IR communication device without compositive IrDA protocol, VIR is the good instead of IrDA protocol with low power loss, easy realization and avoiding messy code during data transmission.
With regards to details of VIR definition and how to realize VIR, please refer to addenda D. if you need any technical support, please mail to mk@prttech.com.

2.1.3 IRDA IrCOMM

There are many IrDA protocols, and IrCOMMM is one of IR communication protocols recommended by IrDA Association.
Using IrCOMM protocols, printer can totally avoid the interference from other IR.
MPT-II support the IrCOMM protocol.
Although regard IrCOMM as virtual serial interface, and there is no need to set baud rate for 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 main device manual or its suppliers.
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 and support short distance communication (within 10m). It can transfer the data among many devices, such as mobile phone, PDA, printer, note book, wireless earphone etc. Bluetooth standard is IEEE802, 2.4 GHz. For more details of Bluetooth, please refer to Bluetooth official website: http://www.bluetooth.org.
MPT-II/BL support wireless data transmission interface, meet Bluetooth 4.0 regulations, power class 2.

MPT-II/BL is only sub-device, and user need to drive the printer by Bluetooth main devices (PDA, mobile phone, and notebook). It won't drive the printer to print without other main devices.
MPT-II/BL default device Name is MPT-II/BL. User can change the device Name according to your requirements. Please refer to 【2.4 Software Tools of MPT-II】.
Default Bluetooth connecting password is 0000. User can change the connecting Name according to your requirements. Please refer to 【2.4 Software Tools of MPT-II】.

2.2.1 Pairing

Before printing, MPT-II/BL 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** .
4. Enter password "**0000**".
Note:Self test-when power off, please push the paper **FEED** button, at the same time t push **PWOER** button, printer can print the self-testing paper which shows the printer status, printer setting parameters and printing sample.
5. Finish pairing.
Detailed pairing methods, please refer to main devices Bluetooth Names. In pairing, mobile thermal printer MPT-II 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 are 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.

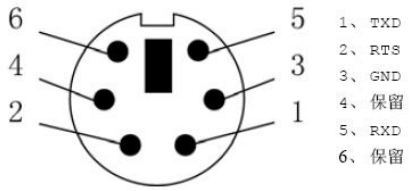
2.2.2 Printing by Bluetooth interface

After pairing, main devices with virtual Bluetooth interface （ smart phone , pocket PC, palm, notebook ）can drive mobile thermal printer MPT-II/BL to printer through such virtual Bluetooth interface. If there is no virtual Bluetooth interface on main devices, please turn to these suppliers for more information.

2.3 RS232 interface

RS232 is frequently-used interface. Mobile thermal printers MPT-II all have the RS232 serial interface.
Mobile thermal printer MPT-II RS232 interface specification:
Date transfer: serial
Synchronous mode: synchronization
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	Main devices
RXD	TXD
GND	GND

When print date is more than 3K byte, need flow control. When connection through Bluetooth and IrCOMM, there is no need to consider the flow control. When using VIR protocols, users need flow control to avoid the overflow from data buffer, according to VIR protocols. No flow control in RAW-IR. When printer is on hardware flow control, connection between main device and printer as follow:

Printer	Main device
TXD	RXD
RTS	CTS
RXD	TXD
GND	GND

During communication, main device can monitor the signal intensity of CTS. When the signal intensity is strong, main device can send data to printer. Reversely, 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 devices
TXD	RXD
RXD	TXD
GND	GND

When using flow control, main devices need 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 Software Tools of MPT-II

There is software which called MPTTools.
MPTTools is software to set the printer parameters, it used to set the baud rate of serial port, Infrared mode, RAW-IR and VIR, it also support setting the printer Bluetooth device name and password.
The instructions pls refer to MPTTools Guide .

3. Miscellaneous Commands

The command set of Mobile thermal printer (MPT-II) is compatible with ESC/POS.
Function type:

Chapter	Command	Description
Print commands		
3.1.1	ESC @	Initialize printer
3.1.2	FF	Print and return to standard mode (in page mode)
3.1.3	LF	Print and line feed
3.1.4	ESC J n	Print and feed paper
3.1.5	ESC d n	Print and feed **n** lines
3.1.6	HT	Horizontal tab
3.1.7	ESC v	Transmit printer status
3.1.8	GS E n	Select head control method
3.1.9	ESC K n	Reverse feed
Character commands		
3.2.1	ESC ! n	Select print mode
3.2.2	GS ! n	Select character size
3.2.3	ESC M n	Select character font
3.2.4	ESC – n	Turn underline mode on/off
3.2.5	ESC E n	Turn emphasized mode on/off
3.2.6	ESC G n	Turn double-strike mode on/off
3.2.7	GS B n	Turn white/black reverse printing mode on/off
3.2.8	ESC R n	Select an international character set
3.2.9	ESC t n	Select character code table
Print position commands		
3.3.1	ESC $ nL nH	Set absolute print positin
3.3.2	ESC D n1 n2…nk NULL	Set horizontal tab position
3.3.3	ESC 2	Select default line spacing
3.3.4	ESC 3 n	Set line spacing
3.3.5	ESC SP n	Set character spacing
3.3.6	ESC a n	Select justification
3.3.7	GS L nL nH	Set left margin
3.3.8	GS W	Set print area width
Bit image commands		
3.4.1	ESC * m nL nH d1…dk	Select bit-image mode
3.4.2	GS * x y d1…dk	Define downloaded bit image

3.4.3	GS / n	Print downloaded bit image
3.4.4	FS p n m	Print bit image in NV memory
3.4.5	FS q n [xL xH yL yH d1...dk]1...[xL xH yL yH d1...dk]n	Print NV bit image
3.4.6	GS v 0 m xL xH yL Yh d1....dk	Print raster bit image
Bar code commands		
3.5.1	GS h n	Set bar code height
3.5.2	GS w n	Set bar code width
3.5.3	GS H n	Select printing position of HRI characters
3.5.4	GS f n	Select font for Human Readable Interpretation (HRI) characters
3.5.5	GS k	Print bar code
3.5.6	GS k m v r d1……dk [NULL]	Print QRCODE
3.5.7	GS (k <Function 167>	QR Code: Set the size of module
3.5.8	GS (k <Function 169>	QR Code: Select the error correction level
3.5.9	GS (k <Function 180>	QR Code: Store the dat a in the symbol storage area
3.5.10	GS (k <Function 181>	QR Code: Pr int the symbol data in the symbol storage area
Curve print commands		
3.6.1	GS ‘	print line section on a horizontal
User-defined character commands		
3.7.1	ESC % n	Select/cancel user-defined character set
3.7.2	ESC & y c1 c2 [x1 d1...d(y * x1)]...[xk d1...d(y * xk)]	Define user-defined characters
3.7.3	ESC ?	Cancel user-defined characters
Kanji character commands		
3.8.1	FS &	Select Chinese character mode
3.8.2	FS 2 c1 c2 d1…dk	Define user-defined Kanji characters
3.8.3	FS .	Canceling chinese mode

3.1 Miscellaneous 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 0C

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 Addenda 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**

[Progame example] char SendStr[2];

SendStr[0]= 0x0A; //C language'\n' feed line

PrtSendData(SendStr, 1);

3.1.4 ESC J *n*

[Name] Print and feed paper

[Format] ASCII ESC J n

Hex 1B 4A n

Decimal 27 74 n

[Rang] 0 ≤ *n* ≤ 255

[Description] Prints the data in the print buffer and feeds the paper n × (vertical motion unit).

[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] = 0x4A;

SendStr[2] = 0x08;

PrtSendData(SendStr, 3);// feeding1mm

3.1.5 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] = 0x64;

SendStr[2] = 0x02;

PrtSendData(SendStr, 3);//feeding 2 lines

3.1.6 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);

3.1.7 ESC v

[Name] Transmit printer status

[Format] ASCII ESC v
Hex 1B 76
Decimal 27 118

[Description] • This command is only effective with a serial interface printer to transmit printer status to host machine.

• When the printer receive command, transfer a byte to the hardware.

Defined as follows:

the command returns 1：Printer is not ready.

the command returns 0：Printer is not ready

[Note] • This command is only effective with a serial interface mode

3.1.8 GS E n

[Name]Select head control method

[Format] ASCII GS E n
Hex 1D 45 n
Decimal 29 69 n

[Range]0 ≤ n ≤ 255

[Description] Selects the head control method.

n	dot-line/ms	n	dot-line/ms
0	0.8	8	1.6
1	0.9	9	1.7
2	1.0	10	1.8
3	1.1	11	1.9
4	1.2	12	2.0
5	1.3	13	2.1
6	1.4	14	2.2
7	1.5	15	2.3

3.1.9 ESC k n

[Name] Reverse feed

[Format] ASCII ESC K n
Hex 1B 4B n
Decimal 27 75 n

[Range] 0 ≤ n ≤ 255

[Description] feeds the paper n × 0.125 in the reverse direction.

Reverse direction paper feeding causes the following problems:

Paper feed pitch is incorrect.

Printer noise is louder than normal.

The paper may rub against the ribbon and become dirty.

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 ～1	0	0	Western character Font A (12 × 24), Chinese character Font A (24× 24)
	1	0	
	0	1	Western character Font B (8 × 16), Chinese character Font B (16× 16)
	1	1	Western character Font B (9 × 24), Chinese character Font B (18× 24)
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] = 0x21;
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 !，ESC @**

[Program example]
```
char SendStr[4];
SendStr[0] = 0x1D;
SendStr[1] = 0x21;
SendStr[2] = 0x01;// 00000001 double height
PrtSendData( SendStr, 3);
```

3.2.3 ESC M *n*

[Name] Select character font

[Format] ASCII ESC M *n*

Hex 1B 4D *n*

Decimal 27 77 *n*

[Rang] *n* = 0, 1, 16, 17,18,19

[Description] Selects the character font.

n	Function

0	Western character font (12 × 24)
1	Western character font (8 × 16)
16	Simplified chinese character font (24×24)
17	Simplified chinese character font (16×16)
18	BIG5 chinese character font（24×24）
19	BIG5 chinese character font（16×16）

[Note] • **ESC !** can also select character font types. However the setting of the last received command is effective.

• When you use this command to set the font,can be set separately chinese font and western font, and independently of each other.

[Reference] **ESC !, ESC @**

[Program example] char SendStr[8];
SendStr[0]=0x1B;
SendStr[1]=0x4d;
SendStr[2]=0x00;// Western character font (12×24)
SendStr[0]=0x1B;
SendStr[1]=0x4D;
SendStr[2]=0x10;// Simplified chinese character font（16×16）
PrtSendData(SendStr, 6);// Chinese character font for （16 × 16）, Western is (12 × 24) after printing

3.2.4ESC – *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)
2	Turn underline on (two-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[16];`
SendStr[0] = 0x1B;

```
SendStr[1] = 0x2D;// uniline underline
SendStr[2] = 0x01;
SendStr[3] = 't';
SendStr[4] = 'e';
SendStr[5] = 's';
SendStr[6] = 't';
SendStr[7] = 0x0A;
SendStr[8] = 0x1B;
SendStr[9] = 0x2D;
SendStr[10] = 0x00;
SendStr[11] = 't';
SendStr[12] = 'e';
SendStr[13] = 's';
SendStr[14] = 't';
SendStr[15] = 0x0A;
PrtSendData( SendStr, 16);
```

3.2.5 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] = 0x45;
SendStr[2] = 0x01;// emphasized mode
PrtSendData(SendStr,3);
```

3.2.6 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[1] = 0x47;

SendStr[2] = 0x01;// turn double-strike mode on

PrtSendData(SendStr, 3);

3.2.7 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 off;

When LSB is 1, white/black reverse printing mode is turned on

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);

3.2.8 ESC R *n*

[Name] Select an international character set

[Format]

ASCII	ESC	R	n
Hex	1B	52	n
Decimal	27	82	n

[Range] 0≤n≤15

n	ASCII code

0	U.S.A.
1	France
2	Germany
3	U.K.
4	Denmark I
5	Sweden
6	Italy
7	Spain I
8	Japan
9	Norway
10	Denmark II
11	Spain II
12	Latin America
13	Korea
14	Slovenia/Croatia
15	China

[Note] Only character Font 0 and Font 1 has international character set. The command is ineffective with other fonts.

[Default] n=0

County	ASCII Code(Hex)											
	23	24	40	5B	5C	5D	5E	60	7B	7C	7D	7E
U.S.A.	#	$	@	[	\	]	^	`	{	|	}	~
France	#	$	à	◦	ç	§	^	`	é	ù	è	¨
Germany	#	$	§	Ä	Ö	Ü	^	`	ä	ö	ü	ß
U.K.	£	$	@	[	\	]	^	`	{	|	}	~
Denmark Ⅰ	#	$	@	Æ	Ø	Å	^	`	æ	ø	å	~
Sweden	#	¤	É	Ä	Ö	Å	Ü	é	ä	ö	å	ü
Italy	#	$	@	°	\	é	^	ù	à	ò	è	ì
Spain Ⅰ	Pt	$	@	¡	Ñ	¿	^	`	¨	ñ	}	~
Japan	#	$	@	[	¥	]	^	`	{	|	}	~
Norway	#	¤	É	Æ	Ø	Å	Ü	é	æ	ø	å	ü
Denmark Ⅱ	#	$	É	Æ	Ø	Å	Ü	é	æ	ø	å	ü
Spain Ⅱ	#	$	á	¡	Ñ	¿	é	`	í	ñ	ó	ú
Latin	#	$	á	¡	Ñ	¿	é	ü	í	ñ	ó	ú
Korea	#	$	@	[	₩	]	^	`	{	|	}	~
Slovenia/Croatia	#	$	Ž	Š	Ď	Ć	Č	ž	š	ď	ć	č
China	#	¥	@	[	\	]	^	`	{	|	}	~

3.2.9 ESC t *n*

[Name] Select character code table

[Format] ASCII ESC t n

Hex 1B 74 n

Decimal 27 116 n

[Range] 0≤n≤47

[Description] Select character code table

n	Code Page	n	Code Page
0	CP437 [U.S.A., Standard Europe]	24	CP737 [Greek]
1	Katakana	25	WCP1257 [Baltic]
2	CP850 [Multilingual]	26	Thai
3	CP860 [Portuguese]	27	CP720[Arabic]
4	CP863 [Canadian-French]	28	CP855
5	CP865 [Nordic]	29	CP857[Turkish]
6	WCP1251 [Cyrillic]	30	WCP1250[Central Eurpoe]
7	CP866 Cyrilliec #2	31	CP775
8	MIK[Cyrillic /Bulgarian]	32	WCP1254[Turkish]
9	CP755 [East Europe，Latvian 2]	33	WCP1255[Hebrew]
10	Iran	34	WCP1256[Arabic]
11	reserve	35	WCP1258[Vietnam]
12	reserve	36	ISO-8859-2[Latin 2]
13	reserve	37	ISO-8859-3[Latin 3]
14	reserve	38	ISO-8859-4[Baltic]
15	CP862 [Hebrew]	39	ISO-8859-5[Cyrillic]
16	ＷCP1252 Latin I	40	ISO-8859-6[Arabic]
17	WCP1253 [Greek]	41	ISO-8859-7[Greek]
18	CP852 [Latina 2]	42	ISO-8859-8[Hebrew]
19	CP858 Multilingual Latin Ⅰ+Euro)	43	ISO-8859-9[Turkish]
20	Iran II	44	ISO-8859-15 [Latin 3]
21	Latvian	45	Thai2
22	CP864 [Arabic]	46	CP856
23	ISO-8859-1 [West Europe]	47	Cp874

[Note] • Only font 0 and font 1 have character code table and this command is ineffective with other fonts.
[Default] Default character code table 437.

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] Sets 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 N with double bytes and signless integer.

[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] = 0x24;
SendStr[2] = 0x18;//3×8＝24
SendStr[3] = 0x00;
PrtSendData(SendStr, 4); 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] • Tap 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 × 24) and character spacing 8 (list9, 17,25,…….).
[Reference] **HT**
[Program example] char SendStr[7];

```
char NextPos = 0x09;
SendStr[0] = 0x1B;
SendStr[1] = 0x44;
SendStr[2] = 0x0B;// 10 character spacing  from first list
SendStr[3] = 0x11;// 16 character spacing  from first list
SendStr[4] = 0x17;// 22 character spacing  from first list
SendStr[5] = 0x1D;// 28 character spacing  from first lsit
SendStr[6] = 0x00; // 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] = 0x32;
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] = 0x33;

SendStr[2] = 0x10;

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] Aligns 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 processesd 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] = 0x61;

SendStr[2] = 0x01;
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] = 0x1D;
SendStr[1] = 0x4C;
SendStr[2] = 0x10;
SendStr[3] = 0x00;
PrtSendData(SendStr,4);// Set left margin to 16 horizontal motion units (2mm).

3.3.8 GS W

[Name] Set print area width

[Format] ASCII GS W nL nH
Hex 1D 57 nL nH
Decimal 29 87 nL nH

[Range] 0 ≤ (nL + nH × 256) ≤ 65535 (0 ≤ nL ≤ 255, 0 ≤ nH ≤ 255)

[Description] In standard mode, sets the prin t area width to (nL + nH×256)×(horizontal motion unit).

[Notes] • When standard mode is selected, this command is enabled only when processed at the beginning of the line.
- The print area width has no effect in page mode. If this command is processed in page mode, the print area width is set and it is enabled when the printe r returns to standard mode.
- If the [left margin + prin t area width] exceeds the printable area, the print area width is automatically set to [printable area – left margin].
- If this command and GS L set the print area width to less than the widt h of one character, the print area width is extended to accommod ate one character for the line.
- Horizontal motion unit is used.
- If horizontal motion unit is changed after setting the printable area width, the printable area width setting will not be changed.
- Printable area width setting is effective until ESC @ is executed, the

printer is reset, or the power is turned off.

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 exceeds printable area of black & white bitmap. Parameter definition as follow :

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

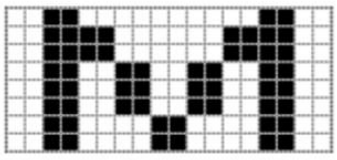
m	Vertical dots (Heights)	Double width mode
0	8	double width
1	8	Single width
32	24	double width
33	24	Single width

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



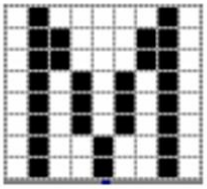
Amplification print figure

Print Image showed, procession code as follow:

```
char SendStr[15];
SendStr[0] = 0x1B;
SendStr[1] = 0x2A;
```

SendStr[2] = 0x00;//m=0 (height 8 dots, double width)
SendStr[3] = 0x09;// (image width 9 dotds)
SendStr[4] = 0x00;
SendStr[5] = 0x00;// (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] = 0x00;
SendStr[14] = 0x0a; // paper feed
PrtSendData(SendStr,15);// 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

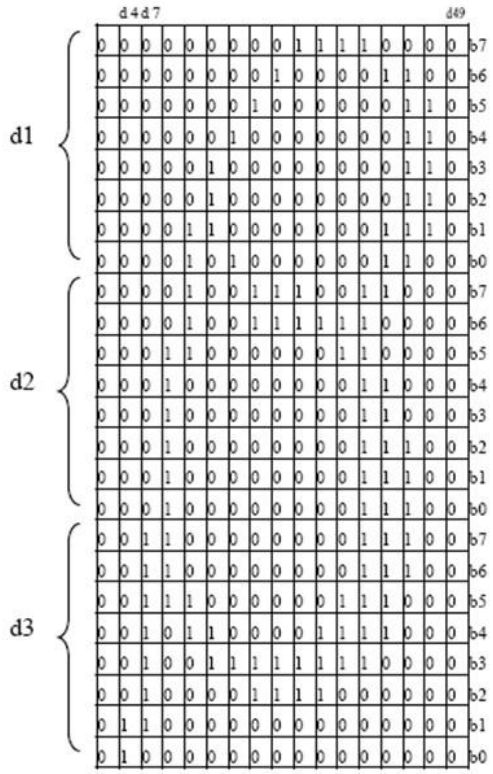
d1	d2	d3	d4	d5	d6	d7	d8	d9	
0	1	0	0	0	0	0	1	0	b7
0	1	1	0	0	0	1	1	0	b6
0	1	1	0	0	0	1	1	0	b5
0	1	0	1	0	1	0	1	0	b4
0	1	0	1	0	1	0	1	0	b3
0	1	0	1	0	1	0	1	0	b2
0	1	0	0	1	0	0	1	0	b1
0	1	0	0	1	0	0	1	0	b0

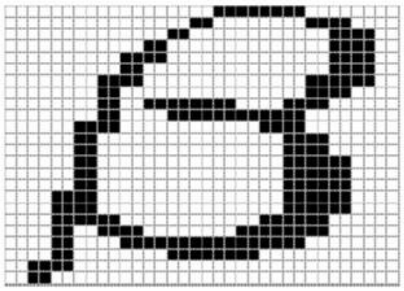


Amplification print figure

Print Image showed, procession code as follow:
char SendStr[15]；
SendStr[0] = 0x1B；
SendStr[1] = 0x2A；
SendStr[2] = 0x01； //m=1 (height 8 dots, no enlargement)
SendStr[3] = 0x09； // image width 9 dots
SendStr[4] = 0x00；
SendStr[5] = 0x00； // image dot line bitmap
SendStr[6] = 0xFF；
SendStr[7] = 0x60；
SendStr[8] = 0x1C；
SendStr[9] = 0x03；

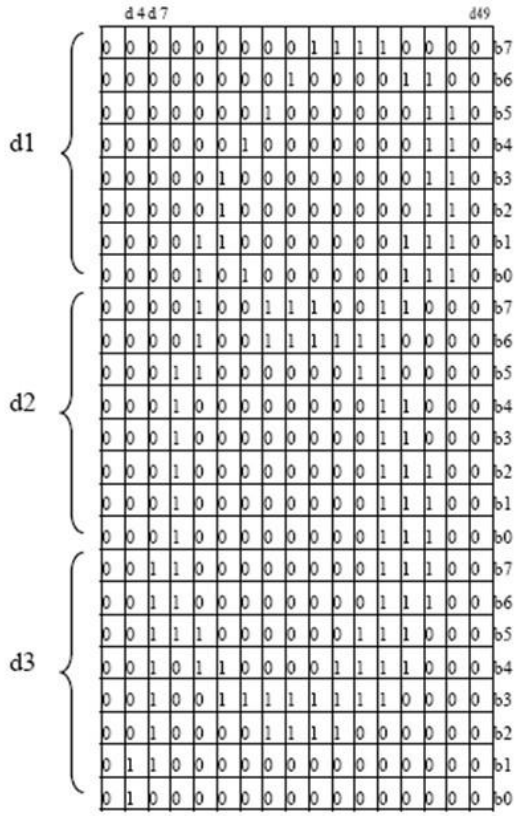
SendStr[10] = 0x1C；
SendStr[11] = 0x60；
SendStr[12] = 0xFF；
SendStr[13] = 0x00；
SendStr[14] = 0x0a； // paper feed
PrtSendData(SendStr,15)；// print image
Example3: m=32 (24 dots, double width) d1,d2,d3 indicate the first, second and third list data, dk indicates the k^{th} list data, bn indicates the n^{th} of byte

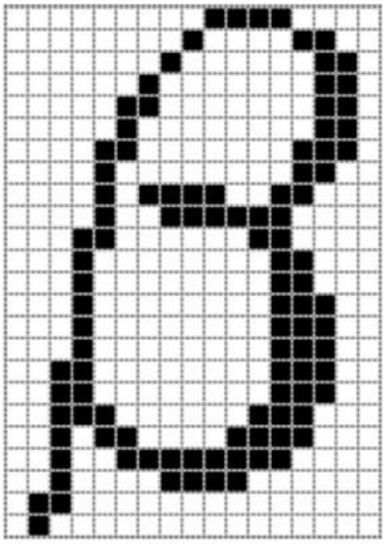



Amplification print figure

Print Image showed, procession code as follow:
char SendStr[64];
SendStr[0] = 0x1B;
SendStr[1] = 0x2A;
SendStr[2] = 0x20;//m=32 (Height 24 dots, double width)
SendStr[3] = 0x11;// 17dots image width 17 dots
SendStr[4] = 0x00;
// 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





Amplification print figure

Print Image showed, procession code as follow:
char SendStr[64];
SendStr[0] = 0x1B;
SendStr[1] = 0x2A;
SendStr[2] = 0x21;// m=33 (height 24 dots, no enlargement)
SendStr[3] = 0x11;// 17dots image width 17 dots
SendStr[4] = 0x00;
// 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] $1 \le x \le 255$

$1 \le y \le 48$

$x * y \le 576$

$0 \le d \le 255$

k=x*y*8

[Description] Using 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.**

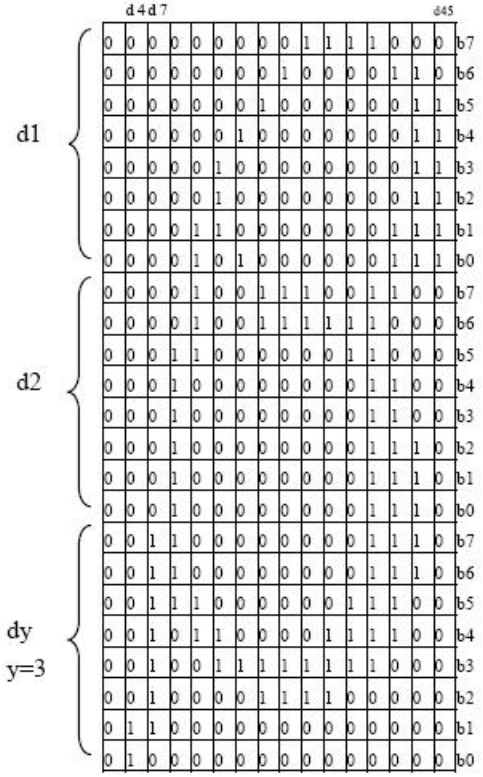
• Clear the downloaded bit image under following cases;

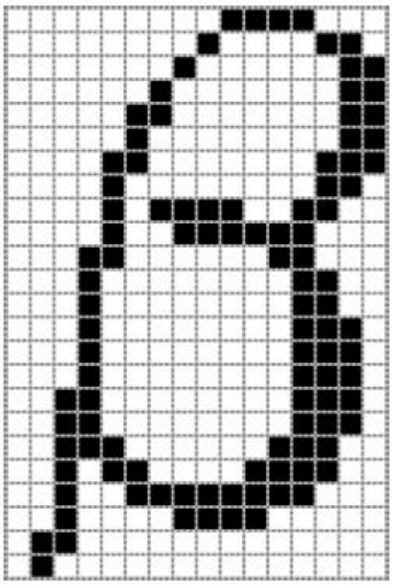
1 .Execute **ESC@**.

2. The printer is reset and the poser is turned off.

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

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





Print Image showed, procession code as follow:

char SendStr[64];

SendStr[0] = 0x1D;

SendStr[1] = 0x2A;

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 **GS L** 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 **3.4.2 GS *** program example

3.4.4 FS p *n*

[Name] Print bit image in NV memory

[Format] ASCII FS p *n*

Hex 1C 70 *n*

Decimal 28 112 *n*

[Range] 0 ≤ *n* ≤ 7

[Description] This command prints the pre-stored non-volatile memory in the printer 2 values in the

bitmap. Printer non-volatile memory bitmap on a PC via a special tool to generate and write

the bitmap width of up to 128, the maximum height of 64.

n is the number of the NV bit image .

[Note] • This command is not effective when the specified NV bit image has not been defined.

• NV bit image must be 2 value bit image.

• This command is not affected by print modes (emphasized, double-strike, underline, character size, white/black reverse printing, or 90° rotated characters,etc.), except upside-down printing mode.

• The printing area width is extended to the right in NV bit image mode up to one line vertically. In this case, printing does not exceed the printable area.

•Need to use special tools to upload print bitmaps, see [three, MPT-II printer tool]. In this way upload bitmap is not lost, unless the re-upload to another overlay bitmap.

[Reference] **ESC *, GS /**

[Program example] char SendStr[4]；

SendStr[0] = 0x1C；

SendStr[1] = 0x50；

SendStr[2] = 0x01；

PrtSendData(SendStr,3)；// Print number 1 of the NV bit image

3.4.5 FS q n [xL xH yL yH d1...dk]1...[xL xH yL yH d1...dk]n

[Name] Define NV bit image

[Format]

ASCII	FS	q	n [xL xH yL yH d1...dk]...[xL xH yL yH d1...dk]
Hex	1C	71	n [xL xH yL yH d1...dk]...[xL xH yL yH d1...dk]
Decimal	28	113	n [xL xH yL yH d1...dk]...[xL xH yL yH d1...dk]

[Range] 1 ≤ n ≤ 255

0 ≤ xL ≤ 255

1 ≤ (xL + xH × 256) ≤ 1023

1 ≤ (yL + yH × 256) ≤ 800

0 ≤ d ≤ 255

k = (xL + xH × 256) × (yL + yH × 256) × 8

Total defined data area =64K bytes

[Description] Define the NV bit image specified by n.

- n specifies the number of the defined NV bit image.
- xL, xH specifies (xL + xH × 256) × 8 dots in the horizontal direction for the NV bit image you are defining.
- yL, yH specifies (yL + yH × 256) × 8 dots in the vertical direction for the NV bit image you are defining.

[Notes]

- Frequent write command executions may damage the NV memory. Therefore, it is recommended to write the NV memory 10 times or less a day.
- This command cancels all NV bit images that have already been defined by this command. The printer cannot redefine only one of several data definitions previously defined. In this case, all data needs to be sent again.
- During processing of this command, the printer is BUSY when writing data to the user NV memory and stops receiving data. Therefore it is prohibited to transmit the data, including real-time commands, during the execution of this command.
- NV bit image is a bit image defined in non-volatile memory by **FS q** and printed by **FS p**.
- In standard mode, this command is effective only when processed at the beginning of

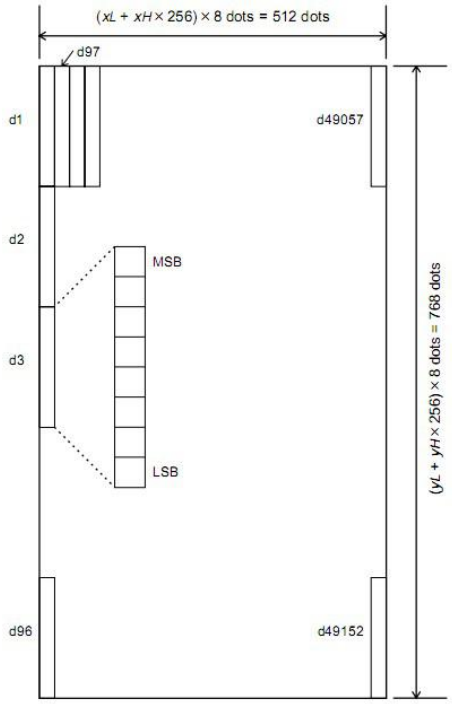
the line.

- This command is effective when 7 bytes <FS yH> of the command are processed normally.
- When the amount of data exceeds the capacity left in the range defined by xL, xH, yL, yH, the printer processes xL, xH, yL, yH out of the defined range.
- In the first group of NV bit images, when any of the parameters xL, xH, yL, yH is out of the definition range, this command is disabled.
- In groups of NV bit images other than the first one, when the printer encounters xL, xH, yL, yH out of the defined range, it stops processing this command and starts writing into the NV images. At this time, NV bit images that haven't been defined are disabled (undefined), but any NV bit images before that are enabled.
- The d indicates the definition data. In data (d) a 1 bit specifies a dot to be printed and a 0 bit specifies a dot not to be printed.
- This command defines n as the number of a NV bit image. Numbers rise in order from NV bit image 01H. Therefore, the first data group [xL xH yL yH d1...dk] is NV bit image 01H, and the last data group [xL xH yL yH d1...dk] is NV bit image n. The total agrees with the number of NV bit images specified by the command **FS p**.
- The definition data for an NV bit image consists of [xL xH yL yH d1...dk].

Therefore, when only one NV bit image is defined n=1, the printer processes a data group [xL xH yL yH d1...dk] once. The printer uses ([data: (xL + xH × 256) × (yL + yH × 256) × 8] + [header :4]) bytes of NV memory.

- The definition area in this printer is a maximum of 64K bytes. This command can define several NV bit images, but cannot define bit image data whose total capacity [bit image data + header] exceeds 40K bytes.
- The printer is busy immediately before writing into NV memory
- The printer does not transmit ASB status or perform status detection during processing of this command even when ASB is specified.
- When this command is received during macro definition, the printer ends macro definition, and begins performing this command.
- Once an NV bit image is defined, it is not erased by performing **ESC @**, reset, and power off.
- This command performs only definition of an NV bit image and does not perform printing. Printing of the NV bit image is performed by the **FS p** command.
- NV bit image of each piece of space in NV memory is equal to the size of the NV bit image data plus 4 bytes.

[Example] When xL = 64, xH = 0, yL = 96, yH = 0



3.4.6 GS v 0 m xL xH yL Yh d1....dk

[Name] Print raster bit image

[Format] ASCII GS v 0 m xL xH yL yH d1...dk

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

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

[Range] 0 ≤ m ≤ 3,
48 ≤ m ≤ 51
0 ≤ xL ≤ 255
0 ≤ xH ≤ 255
0 ≤ yL ≤ 255
0 ≤ d ≤ 255

[Description] k = (xL + xH × 256) × (yL + yH × 256) (k ≠ 0)

print raster bit image, using m to select raster bit image mode:

m	mode	Vertical Dot Density (DPI)	Horizontal Dot Density(DPI)
0, 48	normal	203 DPI	203 DPI
1, 49	double width	203 DPI	101 DPI
2, 50	double heigh	101DPI	203 DPI
3, 51	quadruple	101DPI	101 DPI

• xL、xH indicates the number of bit image bytes in horizontal direction（xL+ xH × 256）

• yL、yH indicates the number of bit image bytes in vertical direction （yL+ yH × 256）.

[Note] • In standard mode, this command is effective only when there is no data in the print buffer.

• Printing modes, such as characters amplification/ emphasized/ double-strike/ underline/ white/ black reverse/ upside-down printing, etc., are effective to this command.

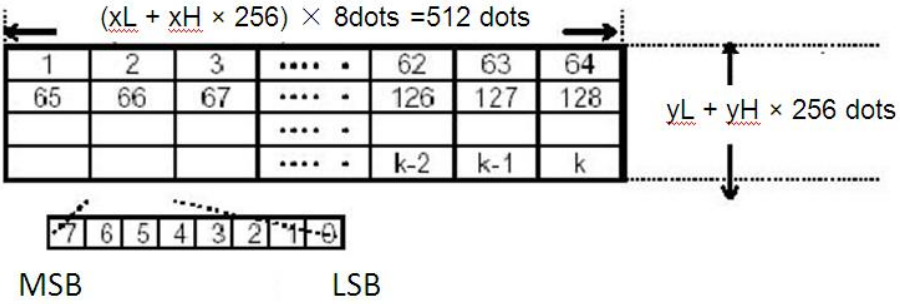
• The part exceeds printing area is not to be printed.

• **ESC a (**select justification) is effective to raster bit image.

• If this command is received while a macro is being defined, the printer ends macro definition mode and execute it. This command is not part of macro definition.

• d indicates the bit image data. Set a bit to 1 to print a dot, or set a bit to 0 to not print a dot.

[Example] When xL+ xH × 256 = 64,



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 **3.5.5 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 **3.5.5 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 **3.5.5. GS k** program example

3.5.4 GS f *n*

[Name] Select font for Human Readable Interpretation (HRI) characters

[Format] ASCII GS f *n*

Hex 1D 66 *n*

Decimal 29 102 *n*

[Range] *n* = 0, 1

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

n selects a font from the following table:

n	Font
0	Font A (12 * 24)
1	Font B (8 * 16)

[Note] • HRI characters are printed at the position specified by GS H.

• This command is not affected by print modes (emphasized, double-strike, underline, character size, white/black reverse printing, or 90° rotated characters,etc.) .

[Default] *n* = 0

[Reference] **GS H, GS k**

[Program example] Please refer to **3.5.5. GS k** program example

3.5.5 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 use command in form1, if barcode type set the data length, K(the data length printer received)should equal to the regulation of data length, if not equal to it , the command is invalid. Concerning barcode data length see【appendix B】。

•The barcode data character (the printer receive) should include in barcode type regulation character collection, if have character beyond in barcode data character, the command is invalid. Concerning barcode data length see【appendix A】。

•When use command in form 2, the n value should equal to the regulation of data length (if the barcode type rule the data length), if not equal to it, the command is invalid, concerning barcode data length see 【appendix B】。

• INTERLEAVED 25(ITF)data length should be even，if use form 1 to print ITF, the k value should be even, if it is odd number, the last number will be ignore. If use form 2 to print ITF, the same as form 1.

• 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.

• Print barcode should meet the barcode regulation , otherwise it may cause no scanning.

• The printer is not calculated check code, if barcode need check code, the check code should include in barcode data, the printer does not check correct errors, wrong check code will cause barcode can not scan.

• If identifiable character is set to print, the invisible character can not be print in CODE93 and CODE128, with '□' replace this time.

• CODE39 not include extend CODE39（EXTERN CODE 39）。

• CODE93 not include extend CODE93 (EXTERN CODE 93）。

• CODE128 barcode data string head should be coding of collection selection character(CODE A, CODE B, CODE C), it can also switch coding collection in a bar code. With character '{' and a character assembly to define special function, through continuously transmit twice '{' to define ASCII character '{'. As the chart

ASCII	HEX	function
{A	7B, 41	Select CodeA
{B	7B, 42	Select CodeB
{C	7B, 43	Select CodeC
{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] = 0x68；

SendStr[2] = 0x28；

PrtSendData(SendStr,3)；// set barcode height of 5mm

SendStr[0] = 0x1D；

SendStr[1] = 0x77；

SendStr[2] = 0x02；

PrtSendData(SendStr,3)；// set barcode width is 2

SendStr[0] = 0x1D；

SendStr[1] = 0x48；

SendStr[2] = 0x02；

PrtSendData(SendStr,3)；// set barcode recognition position below the barcode

SendStr[0] = 0x1D；

SendStr[1] = 0x66；

SendStr[2] = 0x00；

PrtSendData(SendStr,3)；// set barcode recognition is 12*24 dot character

SendStr[0] = 0x1D；

SendStr[1] = 0x6B；

SendStr[2] = 0x04；

SendStr[3] = 'T'； SendStr[4] = 'E' ；

SendStr[5] = 'S'； SendStr[6] = 'T'； SendStr[7] = '8' ；

SendStr[8] = '0'； SendStr[9] = '5'； SendStr[10] = '2' ；

SendStr[11] = 0；

PrtSendData(SendStr,12)；// Use form 1 to print CODE39"TEST8052"

3.5.6 GS k m v r d1……dk [NULL]

[Name] Print QRCODE

[Format]

① m=32

ASCII	GS	k	m v r d1…dk NUL
Hex	1D	6B	m v r d1…dk 00
Decimal	29	107	m v r d1…dk 0

② m=97

ASCII	GS	k	m v r nL nH d1…dn
Hex	1D	6B	m v r nL nH d1…dn
Decimal	29	107	m v r nL nH d1…dn

[Range] m=32 or 97

1 ≤ v ≤ 17 1 ≤ r ≤ 4

[Description]

v is DQCODE version number

r=1 Error correction level is L

r=2 Error correction level is M

r=3 Error correction level is Q

r=4 Error correction level is H

nL, nH is the low and high of integer N,N is the printing bar code data length,Unit is bytes.

When using the first kind of format, the command to 00 at the end, d1 ... dk is the bar code data.

When using the second kind of format,printer to set N characters (d1…dn) behind nH as Bar code data.

[Note] •Because the paper width is limited, the version number of QRCODE maximum is 20.

• Coding standard for QR CODE more information, see China National Standard GB / T 18284-2000 or ISO standard.

ISO/IEC 18004:2000

[Program example]] char SendStr[16];

SendStr[0] = 0x1D; SendStr[1] = 'k'; SendStr[2] = 32;

SendStr[3] = 1;// version is 1

SendStr[4] = 2;// Error correction level is M

PrtSendData(SendStr, 5);

strcpy(SendStr,"123456789");

PrtSendData(SendStr, 10);

3.5.7GS (k <Function 167>

[Name] QR Code: Set the size of module

[Format] ASCII GS (k p L p H cn fn n

Hex 1D 28 6B p L p H cn fn n

Decimal 29 40 107 p L p H cn fn n

[Range] (p L + p H × 256) = 3 (p L = 3, p H = 0)

cn = 49

fn = 67

[Description] • Sets the size of the module for QR Code to n dots.

[Notes] • Settings of this function affect the processing of Functions 181 and 182.

• The unit depends on the printer model.

• Settings of this function are effective until ESC @ is executed, the printer is reset, or the power is turned off.

•n = width of a module = height of a module. (Becaus e the QR code modul es are square.)

3.5.8GS (k <Function 169>

[Name] QR Code: Select the error correction level

[Format] ASCII GS (k p L p H cn fn n

Hex 1D 28 6B p L p H cn fn n

Decimal 29 40 107 p L p H cn fn n

[Range] (p L + p H × 256) = 3 (p L = 3, p H = 0)

cn = 49

fn = 69

48 ≤ n ≤ 51

[Default] n = 48

[Description] • Selects the error correction level for QR Code.

n	Function	Reference: Approx. figure of recovery
48	Select error correction level L	7 %
49	Select error correction level M	15 %
50	Select error correction level Q	25 %
51	Select error correction level H	30 %

[Notes] • Settings of this function affect the processing of Functions 181 and 182.

• QR Code employs Reed-Solom on error correction to generate a series of error correction codewords.

• Settings of this function are effective until ESC @ is executed, the printer is reset, or the power is turned off.

3.5.9GS (k <Function 180>

[Name] QR Code: Store the dat a in the symbol storage area

[Format] ASCII GS (k p L p H cn fn m d1…dk

Hex 1D 28 6B p L p H cn fn m d1…dk

Decimal 29 40 107 p L p H cn fn m d1…dk

[Range] 4 ≤ (p L + p H × 256) ≤ 7092 (0 ≤ p L ≤ 255, 0 ≤ p H ≤ 27)

cn = 49

fn = 80

m = 48

0 ≤ d ≤ 255

k = (p L + p H × 256) – 3

[Description] • Stores the QR Code symbol data (d1...dk) into the symbol storage area.

[Notes] • The symbol data saved in the symbol archive area by this function is encoded by <Function 181> and <Function 182> of this command. After <Function 181> and <Function 182> are executed, the symbol archive area symbol data is kept.

•k bytes of d1...dk are processed as symbol data.

• It is possible to encode to a QR Code as follows. Be sure not to include anything except the following data in the data d1...dk .

Category of data	Characters it is possible to specify
Numerical Mode data	"0" ~ "9"
Alphanumeric Mode data	"0" ~ "9", "A" ~ "Z", SP, $, %, *, +, –, . , /, :
Kanji Mode data	Shift JIS value (Shift value from JISX0208)
8-Bit Byte Mode data	00H ~ FFH

• Settings of this function are effective until the following processing is performed:

• Function 080 or 180 or 280 is executed

• ESC @ is executed

3.5.10 GS (k <Function 181>

[Name] QR Code: Pr int the symbol data in the symbol storage area

[Format] ASCII GS (k p L p H cn fn m

Hex 1D 28 6B 03 00 31 51 m

Decimal 2940107 3 0 49 81 m

[Range] (p L + pH × 256) = 3 (p L = 3, pH = 0)

cn = 49

fn = 81

m = 48

[Description] Encodes and prints the QR Code symbol data in the symbol storage area using t he process of <Function 180>.

[Notes] • In standard mode, use this function when printer is "at the beginning of a line," or "there is no data in the print buffer."

• The symbol size that exceeds the print area cannot be printed.

• If there is any error described below in the data of the symbol storage area, it cannot be printed.

• There is no data (Function 180 is not processed).

• If the data of the sym bol storage area is more than the data allowed by specified model and data compaction mode. (Thi s case is an abnorma l number of data.)

• The four data compaction modes are listed below (in order of compaction rate). Automatically selects best compaction mode by the data of the symbol storage area.

– Numerical mode
– Alphanumeric mode
– Kanji mode
– 8-Bit Byte Mode

• The following data are added automatically by the encode processing.
• Position Detection Patterns
• Separators for Position Detection Patterns
• Timing Patterns
• Format Information
• Version Information
• Error Correction codewords (emp loys the Reed-Solomon Error Detection and Correction algorithm)
• Pad codeword
• Number of bits in Cha racter Count Indicator
• Mode Indicator
• Terminator
• Alignment Patterns (when model 2 is selected)
• Extension Patterns (when model 1 is selected)
• Printing of symbol is not affected by print mode (emphasized, double-strike, underline, white/ black reverse printing, or 90° clockwis e-rotated), except fo r character size and ups ide-down print mode.
• In standard mode, this command exec utes paper feeding for the amount needed for printing the symbol, regardless of the paper feed amount set by the paper feed setting command. The print position returns to the left side of the printabl e area after printing the symbol, and printer is in the status “beginning of the line,” or “there is no data in the print buffer.”
• In page mode, the pr inter stores th e symbol data in the print buffer without executing actual printing. The printer moves print positi on to the next dot of the last data of the symbol.

• The quiet zone is not incl uded in the printing data. Be sure to include the quiet zone when using this function.

3.6 Curve print commands

3.6.1 GS ‘

[Name] Print line section on a horizontal

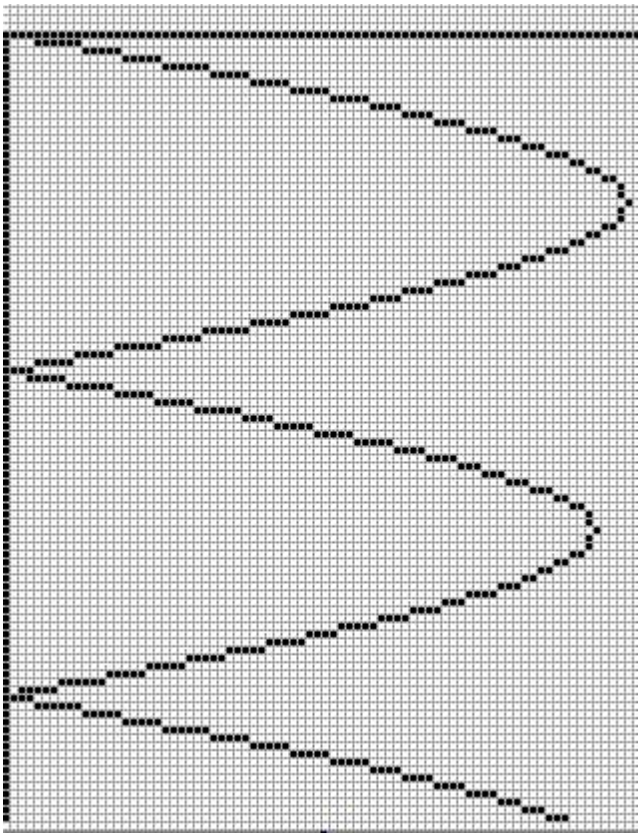
[Format] ASCII GS ‘ *n x1sL x1sH x1eL x1eH … xnsL xnsH xneL xneH*

Hex 1D 27 *n x1sL x1sH x1eL x1eH … xnsL xnsH xneL xneH*

Decimal 29 39 *n x1sL x1sH x1eL x1eH … xnsL xnsH xneL xneH*

[Range] 0 ≤ *n* ≤ 8

[Description] Print amplification figure as shown below: The level of each curve segment by many (points can be regarded as segments of length 1) composition. The instructions for printing a line of n horizontal line segments, continuous use of the command the user can print out the required segments.



xksL ： The K line starting point is the low order of horizontal coordinate;

xksH ： The K line starting point is the high order of horizontal coordinate;

xkeL : The K line end point is the low order of horizontal coordinate;

xkeH : The K line end point is the high order of horizontal coordinate;

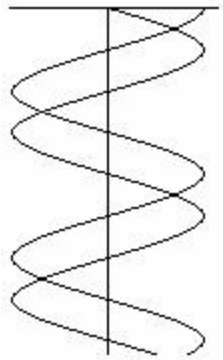
Coordinates starting from the most left of printing area.The minimum is 0, maximum is 383, that xkeL + xkeH * 256 maximum is 383.

The data of line does not need to according to arrange in sequential order;

[Note] • When printing a point，xkeL=xksL, xkeH=xksH。

[Reference] **GS “**

[Program example] Print sin, cos functions continuous curve, as the chart:



Print figure

```
char SendStr[8]；
char SendStr2[16]；
float i；
short y1,y2,y1s,y2s；
// Print the y-axis axis (a line)
SendStr[0] = 0x1D；
SendStr[1] = 0x27；
SendStr[2] = 1； // A line
SendStr[3] = 30；
SendStr[4] = 0； // Starting point
SendStr[5] = 104；
SendStr[6] = 1； // End point
PrtSendData(SendStr,7)；

// Print curve
SendStr[0] = 0x1D；
SendStr[1] = 0x27；
SendStr[2] = 3； // Three lines:X-axis, sin and cos function curve
SendStr[3] = 180； SendStr[4] = 0； // x-axis position
SendStr[5] = 180； SendStr[6] = 0；
for(i=1；i<1200；i++){
    y1 = sin(i/180*3.1416)*(380-30)/2+180； // Calculation of the sin function coordinates
    y2 = cos(i/180*3.1416)*(380-30)/2+180；// Calculation of the cos function coordinates
    if(i==1) { y1s = y1； y2s = y2； }
    PrtSendData( SendStr, 7 )；

    if(y1s < y1)
    {
        PrtSendData( &y1s, 2 )； // Sin function curve at the starting point of the line
        PrtSendData( &y1, 2 )； // Sin function curve at the end point of the line
    }
    else
    {
            PrtSendData( &y1, 2 )； // Sin function curve at the starting point of the line
            PrtSendData( &y1s, 2 )； // Sin function curve at the end point of the line
        }
    if(y2s < y2)
    {
```

```
            PrtSendData( &y2s, 2 );    // Cos function curve at the starting point of the line
            PrtSendData( &y2, 2 );    // Cos function curve at the end point of the line
    }
    else
    {
            PrtSendData( &y2, 2 );    // Cos function curve at the starting point of the line

            PrtSendData( &y2s, 2 );    // Cos function curve at the end point of the line

    }

        y1s = y1;    //when print the next line, sin function curve at the starting point of the line coordinate
        y2s = y2;  // when print the next line, cos function curve at the starting point of the line coordinate
}
```

3.7 User-defined character commands

3.7.1 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* N is only enable when n = LSB

[Default] *n* = 0

[Reference] **ESC &**, **ESC ?**

[Program example] char SendStr[4];

SendStr[0] = 0x1B;

SendStr[1] = 0x25;

SendStr[2] = 0x01;

PrtSendData(SendStr,3);// select user-defined character set

3.7.2 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.

•Can be defined several continuously character code. when it need only a character，

c1should equal to 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.

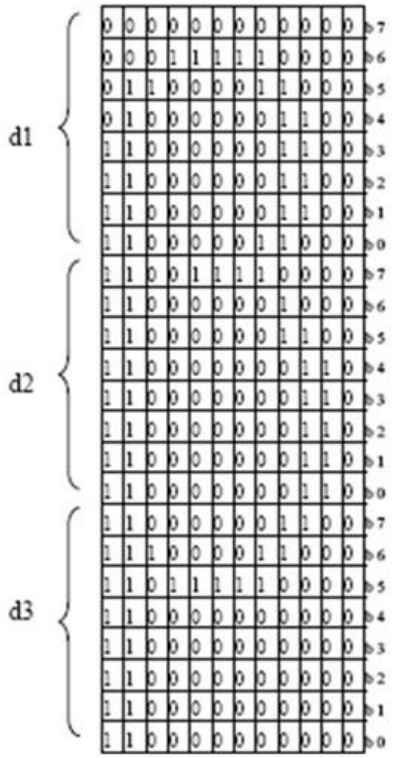
- User-defined character and bitmap is not more than 32K.
- User-defined characters will be cleared under following cases:

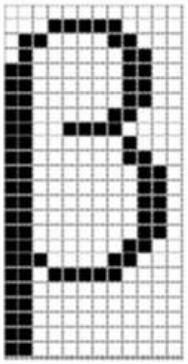
1.Execute **ESC @**

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





Amplification print figure

```
char SendStr[42];
SendStr[0] = 0x1B;
SendStr[1] = 0x26;
SendStr[2] = 0x03;  // y=3   character height 24
SendStr[3] = 0x42;  // character 'B' (code 0x42) redefined
SendStr[4] = 0x42;
SendStr[5] = 0x0C;  //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**

3.7.3 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] = 0x3F;
SendStr[2] = 0x42;
PrtSendData(SendStr,3);// Clear characters defined by 0x42 .

3.8 Kanji character commands.

3.8.1 FS &

[Function] Select Chinese character mode

[Format] ASCII FS &

Hex 1C 26

Decimal 28 38

[Description] Select Kanji character mode.

[Notes] • Chinese character mode is selected autom atically when the power is turned on.

[Reference] FS, FS C

[Example] char SendStr[4];

SendStr[0] = 0x1C;

SendStr[1] = 0x26;

PrtSendData(SendStr,2);

3.8.2 FS 2 c1 c2 d1…dk

[Name] Define user-defined Kanji characters

[Format] ASCII FS 2 *c1 c2 d1…dk*

Hex 1C 32 *c1 c2 d1…dk*

Decimal 28 50 *c1 c2 d1…dk*

[Range] c1 and c2 indicate character codes for the defined characters.

c1 = FEH A1H ≤ *c2* ≤ FEH

0 ≤ *d* ≤ 255

k = 72

[Description] Defines user-defined Kanji characters for the character codes specified by c1 and c2.

[Note] • c1 and c2 indicate character codes for the defined characters. c1 specifies for the first byte, and c2 for the second byte.

•d indicates the dot data. Set a corresponding bit to 1 to print a dot or to 0 to not print a dot. Define data format is the same as ESC & command .

• When the user selects the font A, the definition should be 24 characters dot matrix characters, select font B, the definition should be 16 characters dot matrix characters

[Default] All spaces.

[Reference] **ESC &**

SendStr[0] = 0x1C；

SendStr[1] = 0x2E；

PrtSendData(SendStr,2)；//Turn Kanji characters mode off

3.8.3 FS .

[Name] Canceling chinese mode

[Format] ASCII FS .

Hex 1C 2E

Decimal 28 46

[Description] When the Chinese mode is canceledall the characters are

the same as ASCII style，and deal with one byte once.

[Notes] Selecting Chinese mode when power on.

[Reference] **FS &**, **FS C**

[Example] char SendStr[4];

SendStr[0] = 0x1C;

SendStr[1] = 0x2E;

PrtSendData(SendStr,2);// canceling chinese mode

A Bar code

UPC-A: UPC-A should accord with UCC standard (http://www.uccnet.org)
UPC-E: UPC-E should accord with UCC standard (http://www.uccnet.org)
ENA8: ENA8 should accord with EAN standard (http://www.ean-int.org)
ENA13: ENA13 should accord with EAN standard (http://www.ean-int.org)
CODE39: also called 39 code，CODE39start and stop bit character should be '*'，among start and stop bit should not include character'*' .
ITF: also called INTERLEAVED 25，INTERLEAVED 2 of 5，data bit only can be even
CODABAR: also called Codebar，start and stop bit should be among A,B,C,D.
CODE93: CODE93 start and stop bit character should be'*'，and among start and stop bit should not included'*',

A.1 Bar code length ASCII

Barcode system	Length	ASCII f
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 $ / + % *

B. Pre-printing specifications

If user wants to locate receipt by detecting mark preprinted, testing mark should meet the testing print regulation. Otherwise it may cause printer can't recognize the detecting mark.

Printing position: Testing mark should be printed on right margin of thermo sensitive side,
Width range: width ⩾7mm
Height range: 4mm⩽height⩽6mm
The reflection rate to IR is less than <10% (reflection rate to the detecting mark on paper >65%)
Hps indicates the distance from down edge of detecting mark from upper edge.
0mm⩽Hps⩽1mm