MASUNG Printers
Software Development Kit
Reference Manual

Shenzhen Masung Technology Co.,Ltd.

Software research and development department

Revising history				
Version	Explain	author	Auditor	Revision date
2.1.0.0	create	yijungan		2018-6-12
2.1.0.1	Add mutex lock to interface	yijungan		2018-07-24
2.1.0.2	Add serial port flow control,SetPrintportFlowCtrl	yijungan		2018-08-01
2.2.0.0	Add support for Bluetooth printing(SetPrintConn) Add print Data Matrix code (Print) (only DLL has this function)	yijungan		2018-12-11
	Add PrintQRcode500II	yijungan		201-03-01
	Add PrintQrcodeII	yijungan		201-03-01
	SetDevname optimizing and adding the function of automatic matching USB interface can greatly improve the printing speed.	yijungan		201-03-01

Content

1. SDK introduction

1.1 overview

SDK means Software Development Kit

This SDK manual provides technical reference to all printers manufactured from MASUNG , it will ease developers work and help them to finish terminal application software.

Its advantages:

- There is no need to understand the specific instructions of the printer.
- Packaging the communication with the printer, No need to care processing and printer specific communication protocol.
- Easy bit-image printing
- Easy QR code printing
- Access to printer status checking

1.2 To whom should concern

- Application software developer

1.3 Support printer model

- Full series thermal printer made by MASUNG(if there is without special note)

1.4 Communication interfaces

- USB
- RS-232C

1.5 SDK library

- Msprintsdk.dll

DLL means Dynamic Link Library , it could be used to different application software programs, and it is used on Windows platform,support multiple compiled languages invoking(such as VB\VC\ C#\ QT etc)

- Msprintsdk.ocx

OCX means Object Linking and Embedding (OLE) Control Extension),and it is used on Windows platform, Support VB, VC, C#, QT, JAVA and other development languages, mainly used in web development calls.

- Msprintsdk.so

SO means shared object, it is used on Linux platform.

This SDK manual explained to those three kind libraries. Most of them are same communication interface, difference part will be marked with those signs

stands for this interface is only available for "Msprintsdk.dll"

stands for this interface is only available for "Msprintsdk.ocx "

stands for this interface is only available for "Msprintsdk.so"

Note: if there is no any sign, that means this part information available for all kind SDK document.

2. interface description

2.1 Printer Initialization and close

2.1.1 Set device name (Windows)

[Interface]

int SetPrintport(char *strPort, int iBaudrate) OCX

[function]

Setup printer connection mode,name and serial interface baud rate.

[parameter]

name	type	parameter description
strPort	char *	Printer communication interface name is generally USB00X or COMX(X stand for number 1, 2, 3) Interface will automatically recognize its

		Communication method: USB or COM
iBaudrate	int	Serial baud rate 9600、38400、115200 this parameter should be pair with printer hardware baud rate configuration. When printer is in COM interface connection,this parameter is valid, when printer is in USB interface connection ,this parameter is invalid.

[return value]

0 success

1 failure

[sample]

```
int r = SetPrintport("COM1",115200);


int r = SetPrintport("USB001",0);
```

2.1.2 Set device connection(Windows)

[interface]

```
int SetPrintConn(int iConnWay,char *strName, char *strValue)
```

[function]

Set the printer device connection mode, device name and parameter value.

[parameter]

name	type	parameter description
iConnWay	int	Printer communication mode 1 Serial 2 USB 3 Bluetooth
strName	char *	Printer communication interface name is generally USB00X or COMX(X stand for number

		1, 2, 3) or Bluetooth name.
strValue	char *	Communication baud rate if the connection mode is serial port Support 9600,38400,115200,need to be consistent with the hardware settings of the printer. This parameter is invalid when the connection mode is USB. When the connection mode is Bluetooth, it is the connection password.

[return value]

0 success

1 failure

[sample]

```
int r = SetPrintConn (1,″COM1″,″115200″);

int r = SetPrintConn (2,″USB001″,″″);

int r = SetPrintConn (3,″ MS-BL58″,″1234″);
```

[note]

This upgrade to version 2.2 adds support for bluetooth printers and can replaces the SetPrintport interface.

2.1.3 Set device name(Linux)

[interface]

int SetDevname(int iDevtype, char *cDevname, int iBaudrate)🐧

[function]

Set printer connection mode,name and serial interface baud rate.

[parameter]

name	type	parameter description
iDevtype	int	Printer interfaces 1 COM (including direct COM or virtual COM) 2 USB 3 Automatically match USB
cDevname	char *	Printer name In generally, when iDevtype=1, Direct COM value as/dev/ttySX(X stands for COM number 0、1、2) Virtual COM value as/dev/ttyUSBX(X stands for COM number 0、1、2) When iDevtype=2, USB value as /dev/usb/lpX(X stands for USB number 0、1、2)
iBaudrate	int	Serial baud rate 9600、38400、115200,this parameter should be pair with printer hardware baud rate configuration. iDevtype=1 this parameter is valid iDevtype=2,this paramter is invalid

iDevValue	int	Equipment parameter value
		iDevtype=1,serial-communication baud rate, Support 9600, 38400, 115200, and printer hardware Settings should be consistent;
		iDevtype=2,invalid;
		iDevtype=3,the USB PID value;
		iDevtype=0,Use the default value 0x2013

[return value]

0 success

1 failure

[sample]

```
int r = SetDevname(1,″/dev/ttyS0″,115200);


int r = SetDevname(2,″/dev/usb/lp0″,0);
```

2.1.4 set flow control

[interface]

```
int SetPrintportFlowCtrl(int iFlowCtrlFlag)
```

[function]

Configure the flow control of serial port and USB port

[parameter]

name	type	parameter description
iFlowCtrlFlag	int	0 cancel flow control 1 set flow control

[return value]

0 success

1 failure

[sample]

```
int r1 = SetPrintport(″COM1″,115200);
int r2 = SetPrintportFlowCtrl (1);
```

[note]

It needs to be called after SetPrintport.

When using a serial port, set the flow control, printer exception (such as the printer not powered on), It’s going to keep blocking, please use with caution.

2.1.5 Automatically configure USB interface

[interface]

SetUsbportauto() OCX

[function]

Automatically search USB interface name

[parameter]

none

[return value]

0 success

1 failure

[sample]

```
int r = SetUsbportauto();
```

[note]

This interface is only suitable for printer USB communication.

2.1.6 Automatically configure Serial port interface

[interface]

SetComportauto()

[function]

Automatic search serial port

[parameter]

none

[return value]

0 success

1 failure

[sample]

```
int r = SetComportauto();
```

[note]

The interface is only suitable for printer serial port communication.

2.1.7 Printer initialization

[interface]

SetInit()

[function]

Open the device or port to connect to the printer, initial printer , clear butter memory.

[parameter]

none

[return value]

0 success

1 failure

[sample]

```
int r = SetInit();
```

[note]

Only when this interface is invoked successfully, then developer can make printer working

Please setup printer connection method , name and other parameters before invoking this interface, below are explains for different SDK libraries.

- Msprintsdk.dll

Need to invoke SetPrintinterface or SetUsbinterfaceauto configuration parameters,if developer didn’t invoke those, the dll will look up “Config.ini” from folder includes dll. If there is “Config.ini” file,then his programm will read interface information from this document. Format refers to appendix 4.1.1 Windows configuration.

- Msprintsdk.ocx

Need to invike SetPrintinterface or SetUsbinterfaceauto configuration parameters

- Msprintsdk.so

Need to invoke SetDevname configuration parameters,if developer didn’t invoke those, the dll will look up “Config.ini” from folder includes so . If there is “Config.ini” file then his programm will read interface information from this document. Format refers to appendix 4.1.2 Linux configuration.

2.1.8 Clear buffer memory

[interface]

SetClean()

[function]

Clear printer buffer memory and previous configuration parameters.

[parameter]

none

[return value]

0 success

1 failure

[sample]

```
int r = SetClean();
```

2.1.9 Printer close

[interface]

SetClose()

[function]

Close printer interface and release printer resource

[paramter]

none

[return value]

0 success

1 failure

[sample]

```
int r = SetClose();
```

2.1.10 Set chinese mode

[interface]

SetReadZKmode(int mode)

[function]

Set chinese mode

[parameter]

name	type	parameter description
mode	int	Chinese mode 0 enter into chinese mode 1 exit out chinese mode

[return value]

0 success

1 failure

[sample]

```
int r = SetReadZKmode(0);
```

2.1.11 Set country & language codepage

[interface]

SetCodepage(int country,int CPnumber)

[function]

Set country & language codepage

[parameter]

name	type	parameter description
country	int	Country Refer to below table
CPnumber	int	Codepage

		Refer to below table

Country table

parameter	country
0	US
1	france
2	germany
3	UK
4	denmark I
5	sweden
6	italy
7	spain
8	japan
9	norway
10	denmark II

Codepage table

paramter	codepage
0	PC437[USA-Europe standard]
1	Katakana
2	PC850[multiple language]
3	PC860 [portuguese]
4	PC863 [canadian-french]
5	PC865 [north Europe]
16	WPC1252
17	PC866[slavic2]
18	PC852[latin2]
19	PC858 [Europe]

[return value]

0 success

1 failure

[sample]

```
int r = SetCodepage(0,0);
```

2.2 Normal printing function

2.2.1 Print character string

[interface]

```
int PrintString(char* strData, int iImme)
```

[function]

Print character string

[parameter]

name	type	parameter description
strData	char *	Printing content
iImme	int	Print immediately or not 0 line feed and printing 1 no line feed and not print

[return value]

0 success

1 failure

[sample]

```
int r = PrintString("PrintTest",0);
```

2.2.2 Cutting

[interface]

```
int PrintCutpaper(int iMode)
```

[function]

cutting

[parameter]

name	type	parameter description
iMode	int	Cutting mode 0 full cutting 1 half cutting

[return value]

0　success

1　failure

[sample]

```
int r = PrintCutpaper(0);
```

2.2.3　Print & line feed

[interface]

int PrintChargeRow() 🪟🐧

[function]

Printing data and line feed, if without printing data, printer will feed one blank line.

[parameter]

none

[return value]

0　success

1　failure

[sample]

```
int r = PrintString("PrintTest1",1);
    r = PrintString("PrintTest2",1);
    r = PrintChargeRow();
```

2.2.4　Print & feed paper

[interface]

int PrintFeedDot(int Lnumber) 🪟🐧

[function]

Feeding paper.

[parameter]

name	type	parameter description
Lnumber	int	unit Value range: 0-250 Unit value = 0.125 mm

[return value]

0 success

1 failure

[sample]

```
int r = PrintFeedDot(10);
```

2.2.5 Print self test page

[interface]

int PrintSelfcheck()

[function]

Print self test page.

[parameter]

none

[return value]

0 success

1 failure

[sample]

```
int r = PrintSelfcheck();
```

2.3 printing function

2.3.1 Set line space

[interface]

int SetLinespace(int iLinespace)

[function]

Set line space

[parameter]

name	type	parameter description
iLinespace	int	Line space Value range 0-127 Unit value = 0.125mm

[return value]

0 success

1 failure

[sample]

int r = iLinespace(10);

2.3.2 Set character pitch

[interface]

int SetSpacechar(int iSpace)

[function]

Set character pitch

[parameter]

name	type	parameter description
iSpace	int	Character pitch Value range 0-64 Unit value = 0.125mm

[return value]

0 success

1 failure

[sample]

```
int r = SetSpacechar(10);
```

2.3.3 Set left margin

[interface]

```
int SetLeftmargin(int iLeftspace)
```

[function]

Set left margin

[parameter]

name	type	parameter description
iLeftspace	int	Left margin Value range 0-576 Unit value = 0.125mm

[return value]

0 success

1 failure

[sample]

```
int r = SetLeftmargin(10);
```

2.3.4 set character size

[interface]

```
int SetSizechar(int iHeight,int iWidth,int iUnderline,int iAsciitype)
```

[function]

Set character size

[parameter]

name	type	parameter description
iHeight	int	Double height

		0 invalid 1 valid
iWidth	int	Double width 0 invalid 1 valid
iUnderline	int	Underline 0 invalid 1 valid
iAsciitype	int	ASCII font type 0 12*24 1 9*17

[return value]

0 success

1 failure

[sample]

```
int r = SetSizechar(1,1,1,1);
```

2.3.5 Set text size

[interface]

```
int SetSizetext(int iHeight, int iWidth)
```

[function]

Set text size

[parameter]

name	type	parameter description
iHeight	int	Enlager height Value range 1-8
iWidth	int	Enlager width Value range 1-8

[return value]

0 success

1 failure

[sample]

```
int r = SetSizetext (2,2);
```

2.3.6 Set character alignment

[interface]

```
int SetAlignment(int iAlignment)
```

[function]

Set character alignment

[parameter]

name	type	parameter description
iAlignment	int	Character alignment 0 Left alignment 1 Central alignment 2 right alignment

[return value]

0 success

1 failure

[sample]

```
int r = SetAlignment(1);
```

2.3.7 Set bold character

[interface]

```
int SetBold(int iBold)
```

[function]

Set bold character

[parameter]

name	type	parameter description

iBold	int	Bold character 0 invalid 1 valid

[return value]

0 success

1 failure

[sample]

```
int r = SetBold(1);
```

2.3.8 Set character rotation

[interface]

```
int SetRotate(int iRotate)
```

[function]

Set character rotation

[parameter]

name	type	parameter description
iRotate	int	Character rotation 0 cancel character rotation 1 clockwise rotation 90°

[return value]

0 success

1 failure

[sample]

```
int r = SetRotate(1);
```

2.3.9 Set character direction

[interface]

```
int SetDirection(int iDirection)
```

[function]

Set character direction

[parameter]

name	type	parameter description
iDirection	int	Character direction 0 cancel character rotation 1 rotate 180°

[return value]

0 success

1 failure

[sample]

```
int r = SetDirection(0);
```

2.3.10 Set inverse white

[interface]

```
int SetWhitemodel(int iWhite)
```

[function]

Set inverse white

[parameter]

name	type	parameter description
iWhite	int	Inverse white 0 cancel inverse white 1 set inverse white

[return value]

0 success

1 failure

[sample]

```
int r = SetWhitemodel(1);
```

2.3.11 Set italic

[interface]

int SetItalic(int iItalic)

[function]

Set italic

[parameter]

name	type	parameter description
iItalic	int	italic 0 cancel italic 1 set italic

[return value]

0 success

1 failure

[sample]

```
int r = SetItalic (1);
```

2.3.12 Set underline

[interface]

int SetUnderline(int underline)

[function]

Set underline

[parameter]

name	type	parameter description
underline	int	underline 0 none 1 single dot underline 2 double dots underline

[return value]

0 success

1 failure

[sample]

```
int r = SetUnderline (1);
```

2.3.13 Set chinese size

[interface]

```
int SetSizechinese(int iHeight,int iWidth,int iUnderline,int iChinesetype)
```

[function]

Set chinese size

[parameter]

name	type	parameter description
iHeight	int	Double height 0 invalid 1 valid
iWidth	int	Double width 0 invalid 1 valid
iUnderline	int	underline 0 invalid 1 valid
iChinesetype	int	Chinese font type 0 24*24 1 16*16

[return value]

0 success

1 failure

[sample]

```
int r = SetSizechinese(1,1,1,1);
```

2.3.14 Set chinese pitch

[interface]

```
int SetSpacechinese(int iChsleftspace,int iChsrightspace)
```

[function]

Set chinese pitch

[parameter]

name	type	parameter description
iChsleftspace	int	Chinese left margin Value range 0-64 Unit value = 0.125mm
iChsrightspace	int	Chinese right margin Value range 0-64 Unit value = 0.125mm

[return value]

0 success

1 failure

[sample]

```
int r = SetSpacechinese(10,10);
```

2.3.15 Set horizontal table position

[interface]

```
int SetHTseat(const char* bHTseat,int iLength)
```

[function]

Set horizontal table position

[parameter]

name	type	parameter description
bHTseat	char*	HT position from small to big , unit as single ASCII character, beginning position couldn’t be

		“0”
iLength	int	HT position data quantity Value range 1-32

[return value]

0 success

1 failure

[sample]

Refer to Execure to next HT position;

2.3.16 Execute to next HT position

[interface]

int PrintNextHT()

[function]

Execure to next HT position

[parameter]

none

[return value]

0 success

1 failure

[sample]

```
char cSeat[3]={10,18,25};
SetHTseat(cSeat,3);
PrintString(″1″,1);
PrintNextHT();
PrintString(″2″,1);
PrintNextHT();
PrintString(″3″,1);
PrintNextHT();
PrintString(″4″,0);
PrintString(″1a″,1);
PrintNextHT();
PrintString(″2a″,1);
PrintNextHT();
PrintString(″3a″,1);
PrintNextHT();
PrintString(″4a″,0);
PrintString(″1b″,1);
PrintNextHT();
PrintString(″2b″,1);
PrintNextHT();
PrintString(″3b″,1);
PrintNextHT();
PrintString(″4b″,0);
```

Printing result

```
1       2       3       4
1a      2a      3a      4a
1b      2b      3b      4b
```

2.4 Bit-image & barcodes

2.4.1 Print QR code

[interface]

int PrintQrcode(const char* strData,int iLmargin,int iMside,int iRound)

[function]

Print QR code

[parameter]

name	type	parameter description
strData	char*	QR code inner content
iLmargin	int	Left margin Value range 0-27 Unit value = 1mm
iMside	int	QR code size Value range 1-8
iRound	int	Encircle mode 0 direct printing 1 encircle (synchysis,but some model printers didn’ t support this function)

[return value]

0 success

1 failure

[sample]

```
int r = PrintQrcode("QR Code",2,2,0);
```

[Note]

The printed QR code has multiple interfaces.,see the interface description of the custom class.Any question,please ask technical service personnel for details.

2.4.2 Print mixed QR code

[interface]

PrintRemainQR()

[function]

Print mixed QR code

[parameter]

none

[return value]

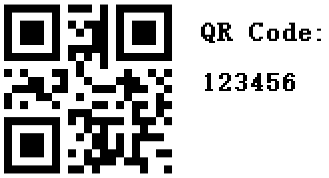
0 success

1 failure

[sample]

```
PrintQrcode(″QR Code:123456″,2,4,1);
SetLeftmargin(120);
PrintString(″QR Code:″,0);
PrintString(″123456″,0);
int r = PrintRemainQR();
```

Printing result:



QR Code:

123456

2.4.3 Print 1D barcode

[interface]

int Print1Dbar(int iWidth,int iHeight,int iHrisize,int iHriseat,int iCodetype,const char* strData)

[function]

Print 1D barcode

[parameter]

name	type	parameter description
iWidth	int	Barcode width Value range 2-6 Unit value = 0.125mm
iHeight	int	Barcode height Value range 1-255 Unit value = 0.125mm
iHrisize	int	Character font type

		0 12*24 1 9*17
iHriseat	int	Character position 0 none 1 Above 2 below 3 up and down
iCodetype	int	Barcode type Barcode type refers to below table
strData	char*	Barcode content

1D barcode type table

parameter	Barcode type
0	* UPC-A
1	* UPC-E
2	* EAN13
3	* EAN8
4	* CODE39
5	* ITF
6	* CODABAR
7	* Standard EAN13
8	* Standard EAN8
9	* CODE93
10	* CODE128

[return value]

0 success

1 failure

[sample]

```
int r = Print1Dbar(2,10,0,1,10,″1Dbar″);
```

2.4.4 Print BMP file from local disk I

[interface]

```
int PrintDiskbmpfile(const char* strPath)
```

[function]

Print BMP file from local disk

[parameter]

name	type	parameter description
strPath	char*	BMP File path If you have only the filename,use the current path;if you specify the full path,use the specified path. 1bit BMP format file.

[return value]

0 success

1 failure

[sample]

```
int r = PrintDiskbmpfile(″D:\\test.bmp″);
```

Printing result:

[note]

Only supports 1bit single color BMP format file

2.4.5 Print BMP file from local disk II

[interface]

int PrintDiskimgfile (const char* strPath)

[function]

Print BMP file from local disk

[parameter]

name	type	parameter description
strPath	char*	BMP File path If you have only the filename, use the current path; if you specify the full path, use the specified path. 1bit and 24 bit BMP format file

[return value]

0 success

1 failure

[sample]

int r = PrintDiskimgfile (″D:\\test.bmp″);

Printing result:

[note]

Supports 1bit and 24 bits single color BMP format file

2.4.6 Set NV BMP file

[interface]

int SetNvbmp(int iNums,const char* strPath)

[function]

Set NV BMP file

[parameter]

name	type	parameter description
iNums	int	BMP file quantity Single file size is within 64KB Total files size is within 192KB (BMP file quantity is without limitation)
strPath	char*	BMP file path • If the full path to any image file is not provided, then the current working folder will be assumed • If multiple image files are specified, then each entry must be separated with a “;” character • The iNums parameter must match the no. of image files specified

[return value]

0 success

1 failure

[sample]

int r = SetNvbmp(3, ″D:\\test1.bmp;D:\\BMP\\test2.bmp;test3.bmp″);

[note]

Only supports 1bit single color BMP format file

this function completely replaces any images currently stored in NV memory

BMP file will be saved at printer flash, this interface is used to fixed BMP file content, such as LOGO information.

2.4.7 Print NV BMP file

[interface]

int PrintNvbmp(int iNvindex,int iMode)

[function]

Print NV BMP file

[parameter]

name	type	parameter description
iNvindex	int	NV BMP file index,from 1 begin Value range 1-N,N depends on “SetNvbmp” setup quantity,when N is over “SetNvbmp” setup quantity, printer won’t work.
iMode	int	BMP file size 48 normal 49 double width 50 double height 51 double width and double height

[retun value]

0 success

1 failure

[sample]

```
int r = PrintNvbmp(1,48);
```

Printing result:

```
int r = PrintNvbmp(1,49);
```

Printing result:

```
int r = PrintNvbmp(1,50);
```

Printing result:

int r = PrintNvbmp(1,51);

Printing result:

2.4.8 Print Data Matrix Code

[interface]

int PrintDataMatrix(const char* strData, int iSize)

[function]

Print Data Matrix code.

[parameter]

name	type	parameter description
strData	char*	content
iSize	int	Matrix size The minimum is 2. Maximum is related to content length.

[return value]

0 success

1 failure

[sample]

```
int r = PrintDataMatrix(″DataMatrix″,6);
```

2.5 Get printer status

2.5.1 Get printer status

[interface]

```
int GetStatus()
```

[function]

Check printer status

[parameter]

none

[return value]

return value	return value description
0	Printer is ready, Power is correct and paper is enough
1	Printer is offline or no power
2	Printer called unmatched library
3	Printer head is opened
4	Cutter is not reset
5	Printer head temp is abnormal (overheating or too low)
6	Printer does not detect blackmark, Possible problems ✧ BM sensor is broken ✧ Used white paper ✧ BM is not standard Notice: This status only can be available when printer blackmark is turn on
7	Paper out
8	Paper low

[sample]

```
int r = GetStatus();
```

2.5.2 Get printer special function status

[interface]

```
int GetStatusspecial ()
```

[function]

Get printer special function status

[parameter]

none

[return value]

return value	return value description
0	Printer is ready, Power is correct and paper is enough
1	Printer is offline or no power
2	Printer called unmatched library
3	Current printer can’t support special function
4	Printer doesn’t load paper to presenter
5	Paper is blocked in printer bezel
6	Paper jams in printer mechanism
7	Unfinished ticket is dragged by outside force
8	There is ticket held on printer bezel (This is detected by ticket taken sensor)

[sample]

```
int r = GetStatusspecial();
```

[note]

Only apply to model MS-D347/EP802 series.

EP802-TM not support special status 4、7
EP802-TMP not support special status 5、7

2.5.3 Get printer information

[interface]

int GetProductinformation(int iFstype,char* bFiddata,int iFidlen)

[function]

Get printer basic information

[parameter]

name	type	parameter description
iFstype	int	Information type 1 printer head model ID 2 printer PCB board version 3 firmware version 4 manufacturer information 5 printer model 6 chinese encode format 7 check sum value
bFiddata	char*	Return information content Output value When iFstype=7 the return value is two bytes hexadecimal,the high in the front,and the low in the back.
~~iFidlen~~	~~int~~	Return information content length (This value is invalid)

[return value]

0 success

1 failure

[sample]

```
int iLen = 0;
char bFiddata[30]={0};
int r = GetProductinformation(3,bFiddata,iLen);
```

2.5.4 Get SDK version information

[interface]

GetSDKinformation(char* bInfodata)

[function]

Get SDK version information

[parameter]

name	type	parameter description
bInfodata	char*	Return information content Output value

[return value]

0 success

1 failure

[sample]

```
char cCmd[20]={0};
int r = GetSDKinformation(bInfodata);
```

2.6 Blackmark

2.6.1 Set blackmark cutting offset

[interface]

int SetMarkoffsetcut(int iOffset)

[function]

Set blackmark cutting offset

[parameter]

name	type	parameter description
iOffset	int	offset Value range 0-1600

		Unit value = 0.125mm

[return value]

0 success

1 failure

[sample]

```
int r = SetMarkoffsetcut(10);
```

2.6.2 Set blackmark printing offset

[interface]

```
int SetMarkoffsetprint(int iOffset)
```

[function]

Set blackmark printing offset.

[parameter]

name	type	parameter description
iOffset	int	offset Value range 0-1600 Unit value = 0.125mm

[return value]

0 success

1 failure

[sample]

```
int r = SetMarkoffsetprint(10);
```

2.6.3 Blackmark detection

[interface]

```
int PrintMarkposition()
```

[function]

Under blackmark mode detection, printer will feed paper and stop on blackmark position

[parameter]

none

[return value]

0 success

1 failure

[sample]

```
int r = PrintMarkposition();
```

[note]

Pinting offset will influence paper feed distance

2.6.4 Detect blackmark and feed paper to printing position

[interface]

```
int PrintMarkpositionPrint()
```

[function]

Under blackmark detection mode, printer will feed paper and stop on printing position

[parameter]

none

[return value]

0 success

1 failure

[sample]

```
int r = PrintMarkpositionPrint();
```

[note]

Printing offset will influence paper feed distance

2.6.5 Detect blackmark and feed paper to cutting position

[interface]

```
int PrintMarkpositioncut()
```

[function]

Under blackmark detection mode, printer will feed paper and stop on cutting position

[parameter]

none

[return value]

0 success

1 failure

[sample]

```
int r = PrintMarkpositioncut();
```

[note]

Cutting offset will influence paper feed distance

2.6.6 Cut blackmark

[interface]

int PrintMarkcutpaper(int iMode)

[function]

Cut blackmark

[parameter]

name	type	parameter description
iMode	int	Cutting mode 0 detect blackmark full cutting 1 not detect blackmark half cutting

[return value]

0 success

1 failure

[sample]

```
int r = PrintMarkcutpaper(0);
```

2.7 Other interface

2.7.1 Transmit command (sending)

[interface]

int PrintTransmit(const char* bCmd,int iLength)

[function]

Transmit original command to printer

[parameter]

name	type	parameter description
bCmd	char*	command
iLength	int	command length

[return value]

0 success

1 failure

[sample]

```
char bCmd[2]={0x1B,0x69};  //cutting command
int r = PrintTransmit(bCmd,2);
```

~~2.7.2 Transmit command (send and receive)~~

[interface]

int GetTransmit(const char* bCmd,int iLength,char* bRecv,int iRelen)

[function]

Transmit original command to printer and receive return value

[parameter]

name	type	parameter description
bCmd	char*	command
iLength	int	Command length

bRecv	char*	Return value
~~iRelen~~	~~int~~	Return value length (This value is invalid)

[return value]

0 success

1 failure

[sample]

```
char bCmd[3]={0x10,0x04,0x01};  //check printer status
char bRecv[3]={0};
int iReLen = 0;
int r = GetTransmit(bCmd,3,bRecv,iReLen);
```

2.8 Customize printer interface description

This interface only applys to customized printer series, eg: EP800、EP802 and so on.

Please consult technical service personnel for details.

2.8.1 Set command mode

[interface]

SetCommandmode(int iMode)

[function]

Switching printer into different command mode, in order to realize customized printing interface function

[parameter]

name	type	parameter description
iMode	int	Command mode 2 EPIC mode 3 EPOS mode

[return value]

0 success

1 failure

[sample]

int r = SetCommandmode(2);

[note]

If developer needs to use customized printing interface , he needs to switch printer into EPIC command mode. after that function is accomplished, developer needs to switch printer into EPOS command mode, then he can use common standard printing interface.

2.8.2 Set rotation printing mode

[interface]

int SetRotation_Intomode()

[function]

Set rotation printing mode

[parameter]

none

[return value]

0 success

1 failure

[sample]

int r = SetRotation_Intomode();

2.8.3 Send rotation mode data

[interface]

int PrintRotation_Sendtext(char* strData,int iImme)

[function]

Send text data in rotation mode.

[parameter]

name	type	parameter description
strData	char*	Text data
iImme	int	Line or not 0　line feed 1　no line feed

[return value]

0　success

1　failure

[sample]

```
int r = PrintRotation_Sendtext("Rotation",1);
```

2.8.4 Send rotation barcode

[interface]

```
int PrintRotation_Sendcode(int leftspace,int iWidth,int iHeight,int iCodetype,const char* strData)
```

[function]

Send barcode data in rotation mode

[parameter]

name	type	parameter description
leftspace	int	barcode left margin value range 0-36 Unit　mm
iWidth	int	Barcode width Value range 2-6 Unit value = 0.125mm
iHeight	int	Barcode height Value range 1-10 Unit value = 3mm
iCodetype	int	Barcode type

		Barcode type refer to below table
strData	char*	Barcode content

Barcode type table

parameter	Barcode type
0	* UPC-A
1	* UPC-E
2	* EAN13
3	* EAN8
4	* CODE39
5	* ITF
6	* CODABAR
~~7~~	~~* Standard EAN13~~
~~8~~	~~* Standard EAN8~~
9	* CODE93
10	* CODE128

[return value]

0 success

1 failure

[sample]

```
int r = PrintRotation_Sendcode(10,3,60,10,"1Dbar");
```

2.8.5 Send rotation line feed

[interface]

```
int PrintRotation_Changeline()
```

[function]

Send line feed in rotation

[parameter]

none

[return value]

0 success

1 failure

[sample]

```
int r = PrintRotation_Changeline();
```

2.8.6 Send rotation left margin

[interface]

```
int SetRotation_Leftspace(int iLeftspace)
```

[function]

Send rotation left margin

[parameter]

name	type	parameter description
iLeftspace	int	Left margin Value range 0-36 Unit mm

[return value]

0 success

1 failure

[sample]

```
int r = SetRotation_Leftspace(10);
```

2.8.7 Printing rotation mode data

[interface]

```
int PrintRotation_Data()
```

[function]

After rotation printing and save data, then exit rotation mode, enter into EPOS command mode

[parameter]

none

[return value]

0 success

1 failure

[sample]

```
int r = PrintRotation_Data();
```

2.8.8 Set printer ID or name

[interface]

```
int SetPrintIDorName(char* strIDorNAME)
```

🪟🐧

[function]

Set printer ID or name

[parameter]

name	type	parameter description
strIDorNAME	char*	Printer ID or name Value length range:1-14

[return value]

0 success

1 failure

[sample]

```
int r = SetPrintIDorName("EP800-000001");
```

2.8.9 Get printer ID or name

[interface]

```
int GetPrintIDorName(char* strIDorNAME)
```

🪟🐧

[function]

Get printer ID or name

[parameter]

name	type	parameter description
strIDorNAME	char*	Get printer ID or name return value

[return value]

0 success

1 failure

[sample]

```
char bRecv[30]={0};
int r = GetPrintIDorName(bRecv);
```

2.8.10 Extend set character alignment

[interface]

```
int SetAlignmentLeftRight(int iAlignment)
```

[function]

Align left and right on the same line.

[parameter]

name	type	parameter description
iAlignment	int	Character alignment 0 left alignment 2 right alignment

[return value]

0 success

1 failure

[sample]

```
SetClean();
SetAlignmentLeftRight(0);
PrintString("Left",1);
SetAlignmentLeftRight(2);
PrintString("Right",0);
```

[note]

must use this as an example, or you might cause data errors.

2.8.11 Set page mode

[interface]

int SetPagemode(int iMode,int Xrange,int Yrange)

[function]

Print data as complete page mode

[parameter]

name	type	parameter description
iMode	int	0 exit page mode 1 enter page mode
Xrange	int	Page width,max 576
Yrange	int	Page height,max 640

[return value]

0 success

1 failure

[sample]

```
int r = SetPagemode(1,576,640);
```

2.8.12 Set page mode data beginning position

[interface]

int SetPagestartposition(int Xdot,int Ydot)

[function]

set page mode data beginning position

[parameter]

name	type	parameter description
Xdot	int	X position,max 576

Ydot	int	Y position,max 640

[return value]

0　success

1　failure

[sample]

```
int r = SetPagestartposition(40,0);
```

2.8.13 Set page mode printing direction

[interface]

int SetPagedirection(int iDirection)

[function]

Set page mode printing direction

[parameter]

name	type	parameter description
iDirection	int	Printing direction 0　normal 1　rotation 90° 2　rotation 180° 3　rotation 270°

[return value]

0　success

1　failure

[sample]

```
int r = SetPagedirection(2);
```

2.8.14 Print data in page mode

[interface]

int PrintPagedata()

[function]

Print complete page data by page mode

[parameter]

none

[return value]

0 success

1 failure

[sample]

```
int r = PrintPagedata ();
```

2.8.15 Print QR Code II

[interface]

int PrintQrcodeII(const char* strData,int iLen,int iMside) 🐧

[function]

Print QR code.

[parameter]

name	type	parameter description
strData	char*	Content of QR code Support for invisible characters
iLen	int	Content length
iMside	int	Size of QR code Value range 1-16

[return value]

0 success

1 failure

[sample]

```
char cQRData[128]={0x31,0x32,0x33,0x34,0x35,0x36};
int r = PrintQrcodeII(cQRData,6,8);
```

2.8.16 Print QR Code III

[interface]

int PrintQRcode500II(int iwidth,const char* strData)

[function]

Print QR code.

[parameter]

name	type	parameter description
iwidth	int	Size of QR code Value range 1-8
strData	char*	Content of QR code

[return value]

0 success

1 failure

[sample]

```
int r = PrintQRcode500II (3,"QR Code");
```

[note]

Retain compatibility with previous SDK version 1.0.

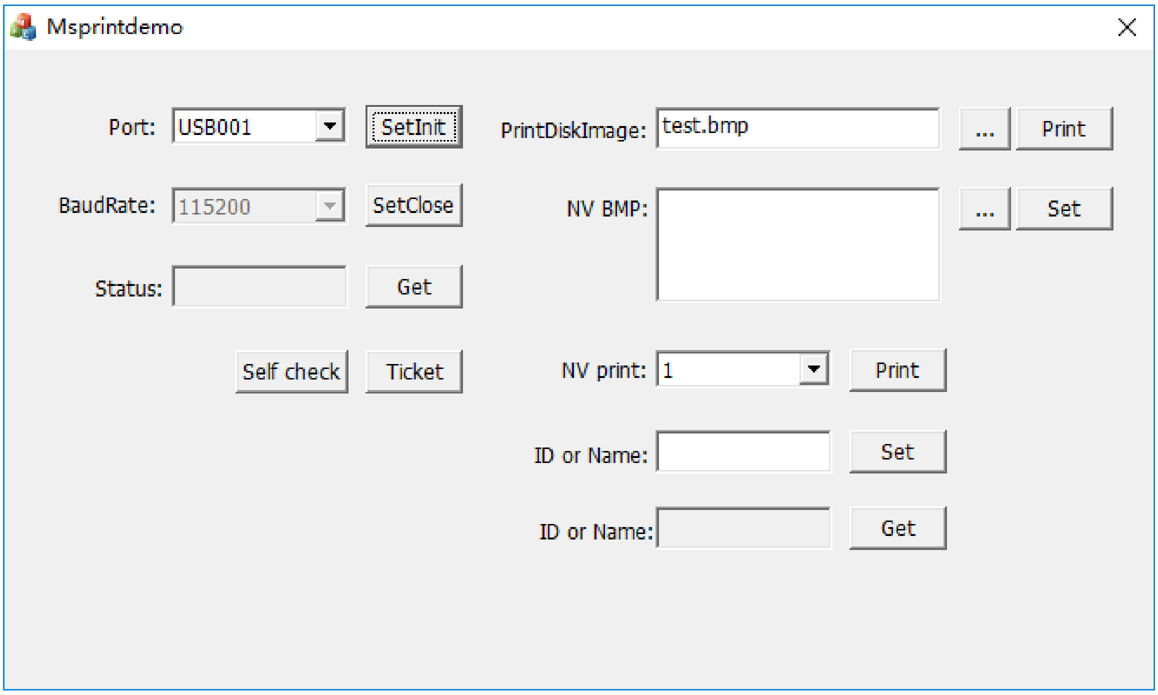
3. Invoking sample

3.1 Msprintsdk.dll

VC sample

Msprintdemo

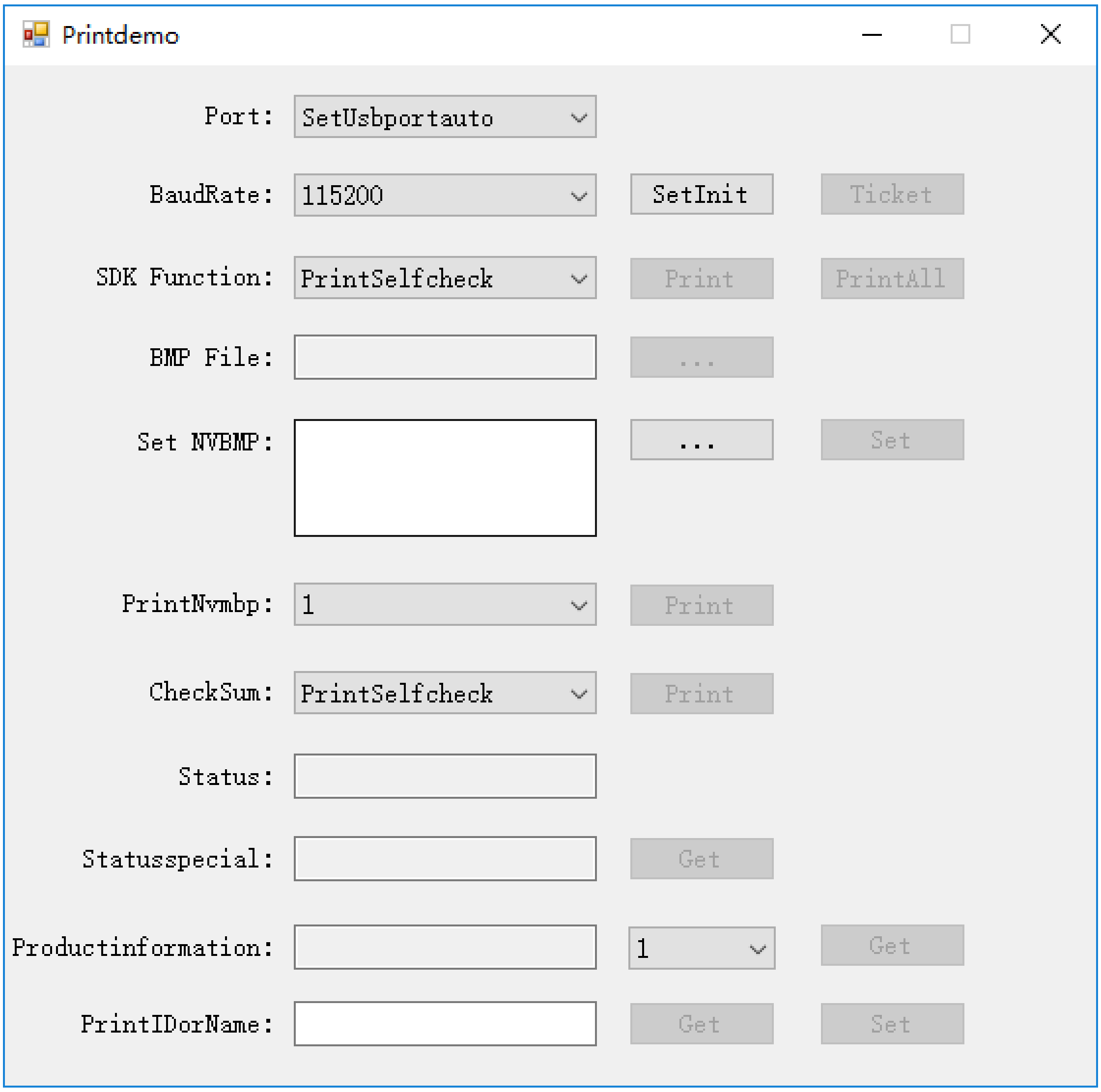
Visual Studio 2008 VC++ compile



C# sample

Msprintcsharp

Visual Studio 2008 C# compile



Printdemo

Port: SetUsbportauto

BaudRate: 115200

SDK Function: PrintSelfcheck

BMP File:

Set NVBMP:

PrintNvmbp: 1

CheckSum: PrintSelfcheck

Status:

Statusspecial:

Productinformation: 1

PrintIDorName:

Notice item attribute , creation->objective platform, configuration as X86; click on “allow unsafe code”

3.2 Msprintsdk.ocx

Needs to register before compile, XP OS can run “RegMsprintsdk1.bat” to register; win7/win8/win10 OS right click on “RegMsprintsdk2.bat” and “run it as administrator”.

Sample refers to “Msprintsdk.htm”

3.3 Msprintsdk.so

Sample msprintdemo compile

```
gcc -o msprintdemo msprintdemo.c /lib/Msprintsdk.so -lstdc++
```

4. appendix

4.1 Configuration file

4.1.1 Windows configuration

[file name]

Config.ini

[sample]

```
[PortConfig]
Portname=COM3
Baudrate=115200
```

[configuration description]

Same as manual Set device name (Windows) parameter

4.1.2 Linux configuration

[file name]

Config.ini

[sample]

```
[DevConfig]
DevType=1
DevName=/dev/ttyUSB0
Baudrate=115200
Or
[DevConfig]
DevType=3
PID=8211
```

[configuration description]

Same as manual Set device name(Linux) parameter