

Printer interface specification

1 Generality.....	3
2 Function declaration.....	4
Port Function.....	4
Port_OpenCom.....	4
Port_OpenTcp.....	5
Port_OpenUsb.....	6
Port_OpenLpt.....	7
Port_OpenPrn	8
Port_CloseCom	9
Port_CloseTcp	10
Port_CloseUsb	11
Port_CloseLpt	12
Port_ClosePrn	13
Port_SetPort	14
Port_EnumCom	15
Port_EnumLpt	16
Port_EnumUsb	17
Port_EnumPrn	18
Page Function.....	19
PAGE_PageEnter.....	19
PAGE_PagePrint.....	20
PAGE_PageExit.....	21
PAGE_SetPrintArea.....	22
PAGE_DrawString.....	24
PAGE_DrawRect.....	26
PAGE_DrawBarcode.....	27
PAGE_DrawQRCode.....	29
PAGE_DrawBitmap.....	30
Pos Function.....	31
POS_TextOut.....	31
POS_SetBarcode.....	33
POS_SetQRCode.....	35
POS_PrintPicture.....	36
POS_SelfTest.....	37
POS_Query.....	38
POS_RTQuery.....	39
POS_TicketSucceed.....	41
POS_FeedLine.....	42
POS_FeedNLine.....	43
POS_FeedNDot.....	44

POS_SetMotionUnit.....	45
POS_SetLineHeight.....	46
POS_SetRightSpacing.....	47
POS_SetAlign.....	48
POS_Reset.....	49
POS_KickOutDrawer.....	50
POS_CutPaper.....	51
POS_FeedAndCut.....	52
POS_Beep.....	53

1 Generality

- 1 PrinterLibs is DLL which use C++ to compile at windows platform, DLL export C style functions.
- 2 PrinterLibs functions have the following category
 - A Port_XXX

Function start with Port, mainly is for open the port, close port, enumerate port.
Support printing through serial, parallel, USB and Lan.
Remark: Port_SetPort: This function can specify communication port for POS-XXX series functions use.
 - B PAGE_XXX

Function start with PAGE, Encapsulate page mode command, it can control print to print by page mode.

 - ① PAGE_PageEnter
 - ② PAGE_SetPrintArea
 - ③ PAGE_DrawXXX
 - ④ PAGE_PagePrint
 - ⑤ PAGE_PageExit

Remark:
②③ can repeatedly call
Only the printer support page mode can use those functions.
 - C POS_XXX

Function start with POS, mainly encapsulate ESC/POS commands, can control printer to print.

 - ① Feed series function can control paper feeding
 - ② Setting series functions can set the printer format
 - ③ Printing series functions can print text, barcode, QR code, pictures
 - ④ Query series functions can check the status of printer.
 - ⑤ Other functions can control cash drawer, cutter, buzzer and so on.

2 Function declaration

Port Function

Port_OpenCom

Syntax

BOOL Port_OpenCom(TCHAR * pName, DWORD dwBaudrate, DWORD dwParity)

Parameters

pName

port name

for example COM1,COM2,COM3...COM11...

dwBaudrate

baudrate

Normally choose 9600,19200,38400,57600,115200.

Need to keep the same as printer baudrate, suggest using high baudrate to get better printing speed.

dwParity

Parity bit

Chose value as following:

#define PARITY_NONE ((WORD)0x0100)

#define PARITY_ODD ((WORD)0x0200)

#define PARITY_EVEN ((WORD)0x0400)

#define PARITY_MARK ((WORD)0x0800)

#define PARITY_SPACE ((WORD)0x1000)

Return value

If open success, return TRUE, OR return FALSE

Remarks

If serial port was occupied, open serial will fail

If baud rate don't match with printer baud rate, it won't print.

Port_OpenTcp

Syntax

BOOL Port_OpenTcp(TCHAR * szIp, USHORT nPort)

Parameters

szIp

IP Address

For example: 192.168.1.87

nPort

Port Number

Fixed value: 9100

Return value

If open success, return TRUE, OR return FALSE

Remarks

PC and printer need in the same network segment, so they can connect

Port_OpenUsb

Syntax

BOOL Port_OpenUsb(TCHAR * pName)

Parameters

pName

port name

Can get printer name through Port_EnumUSB

Can use any other strings, at this time, if find USB printer, will open directly.

Return value

If open successfully, return TRUE, OR return FALSE

Remarks

USB printer connect to computer, if device manager appear USB Printing Support, then can open USE this function.

If Prolific USB-to-Serial Com Port appear, it say this is USB virtual com, then please use Port_OpenCom

Port_OpenLpt

Syntax

BOOL Port_OpenLpt(TCHAR * pName)

Parameters

pName

port name

for example: LPT1,LPT2,LPT3...

Return value

If open successfully, return TRUE, or return FALSE.

Remarks

Parallel port just support one way communication, just can write, but can't read

All the query status function, are invalid for parallel port.

Port_OpenPrn

Syntax

BOOL Port_OpenPrn(TCHAR * pName)

Parameters

pName

printer name。

for example: KP80 Printer

Return value

If open successfully, return TRUE, OR return FALSE.

Remarks

Open printer port, must use our printer driver.

Port_CloseCom

Syntax

VOID Port_CloseCom()

Parameters

Return value

Remarks

Close port

Port_CloseTcp

Syntax

VOID Port_CloseTcp()

Parameters

Return value

Remarks

Close port

Port_CloseUsb

Syntax

VOID Port_CloseUsb()

Parameters

Return value

Remarks

Close port

Port_CloseLpt

Syntax

VOID Port_CloseLpt()

Parameters

Return value

Remarks

Close port

Port_ClosePrn

Syntax

VOID Port_ClosePrn()

Parameters

Return value

Remarks

Close port

Port_SetPort

Syntax

VOID Port_SetPort(DWORD dwPortType)

Parameters

dwPortType

Port Type

#define KCPORTYPE_COM	0x1
#define KCPORTYPE_ETH	0x2
#define KCPORTYPE_USB	0x4
#define KCPORTYPE_LPT	0x8
#define KCPORTYPE_PRN	0x10

Return value

Remarks

Need to set port type, so POS_XXX series functions can normally use.

Port_EnumCom

Enumerate serial port

Syntax

```
VOID Port_EnumCom(TCHAR * pBuf, int cbBuf, int * pcbNeeded, int * pcnReturned)
```

Parameters

pBuf

Enumerated port list

cbBuf

pBuf buffer bytes numbers

pcbNeeded

Needed butter area size

pcnReturned

Returned port numbers.

Return value

Remarks

Use sample code as following: C++

```
int cbNeeded = 0;
int cnReturned = 0;
Port_EnumCom(NULL, 0, &cbNeeded, &cnReturned);
if (cbNeeded)
{
    TCHAR * pBuf = (TCHAR *)malloc(cbNeeded);
    if (pBuf)
    {
        Port_EnumCom(pBuf, cbNeeded, &cbNeeded, &cnReturned);
        TCHAR * pDevice = pBuf;
        for (int i = 0; i < cnReturned; ++i)
        {
            ComboBox_AddString(hCbx, pDevice);
            pDevice += lstrlen(pDevice) + 1;
        }
        free(pBuf);
    }
}
```

Port_EnumLpt

Enumerate parallel

```
VOID Port_EnumLpt(TCHAR * pBuf, int cbBuf, int * pcbNeeded, int * pcnReturned)
```

Parameters

pBuf

Enumerated port list

cbBuf

pBuf buffer bytes numbers

pcbNeeded

Needed size of buffer area

pcnReturned

returned port numbe

Return value

Remarks

Use the demo as following:C++

```
int cbNeeded = 0;
int cnReturned = 0;
Port_EnumLpt(NULL, 0, &cbNeeded, &cnReturned);
if (cbNeeded)
{
    TCHAR * pBuf = (TCHAR *)malloc(cbNeeded);
    if (pBuf)
    {
        Port_EnumLpt(pBuf, cbNeeded, &cbNeeded, &cnReturned);
        TCHAR * pDevice = pBuf;
        for (int i = 0; i < cnReturned; ++i)
        {
            ComboBox_AddString(hCbx, pDevice);
            pDevice += lstrlen(pDevice) + 1;
        }
        free(pBuf);
    }
}
```


Port_EnumUsb

Enumerate USB port

```
VOID Port_EnumUsb(TCHAR * pBuf, int cbBuf, int * pcbNeeded, int * pcnReturned)
```

Parameters

pBuf

Enumerated port list

cbBuf

pBuf buffer bytes numbers

pcbNeeded

needed size of buffer area

pcnReturned

returned port number

Return value

Remarks

Use the following demo code: C++

```
int cbNeeded = 0;
int cnReturned = 0;
Port_EnumUsb(NULL, 0, &cbNeeded, &cnReturned);
if (cbNeeded)
{
    TCHAR * pBuf = (TCHAR *)malloc(cbNeeded);
    if (pBuf)
    {
        Port_EnumUsb(pBuf, cbNeeded, &cbNeeded, &cnReturned);
        TCHAR * pDevice = pBuf;
        for (int i = 0; i < cnReturned; ++i)
        {
            ComboBox_AddString(hCbx, pDevice);
            pDevice += lstrlen(pDevice) + 1;
        }
        free(pBuf);
    }
}
```

Port_EnumPrn

Enumerate printer list

```
VOID Port_EnumPrn(TCHAR * pBuf, int cbBuf, int * pcbNeeded, int * pcnReturned)
```

Parameters

pBuf

Enumerated port list

cbBuf

pBuf buffer bytes numbers

pcbNeeded

needed size of buffer area

pcnReturned

returned port number

Return value

Remarks

Use the following demo code: C++

```
int cbNeeded = 0;
int cnReturned = 0;
Port_EnumPrn(NULL, 0, &cbNeeded, &cnReturned);
if (cbNeeded)
{
    TCHAR * pBuf = (TCHAR *)malloc(cbNeeded);
    if (pBuf)
    {
        Port_EnumPrn(pBuf, cbNeeded, &cbNeeded, &cnReturned);
        TCHAR * pDevice = pBuf;
        for (int i = 0; i < cnReturned; ++i)
        {
            ComboBox_AddString(hCbx, pDevice);
            pDevice += lstrlen(pDevice) + 1;
        }
        free(pBuf);
    }
}
```

Page Function

PAGE_PageEnter

Choose page mode

Syntax

```
BOOL PAGE_PageEnter()
```

Parameters

Return value

If the write command successfully, return TURE, or return FALSE

Remarks

PAGE_PagePrint

Printing content at Page mode

Syntax

```
BOOL PAGE_PagePrint()
```

Parameters

Return value

If the write command successfully, return TURE, or return FALSE

Remarks

PAGE_PageExit

Exit page mode

Syntax

```
BOOL PAGE_PageExit()
```

Parameters

Return value

If the write command successfully, return TURE, or return FALSE

Remarks

PAGE_SetPrintArea

Setting printing area at page mode

Syntax

BOOL PAGE_SetPrintArea(int left, int top, int right, int bottom, int direction)

Parameters

left

x coordinate of left corner printing area

top

Y coordinate of top left corner printing area

right

X coordinate of bottom right corner

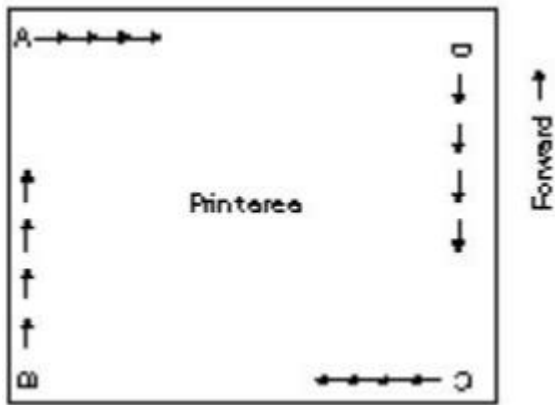
bottom

Y coordinate of bottom right corner

direction

Direction of printing area

direction	Printing direction	Starting position
0	from left to right	Top left corner(A in picture)
1	from bottom to top	Bottom left corner (B in picture)
2	from right to left	Bottom right corner (C in picture)
3	from top to bottom	Top right corner (D in picture)

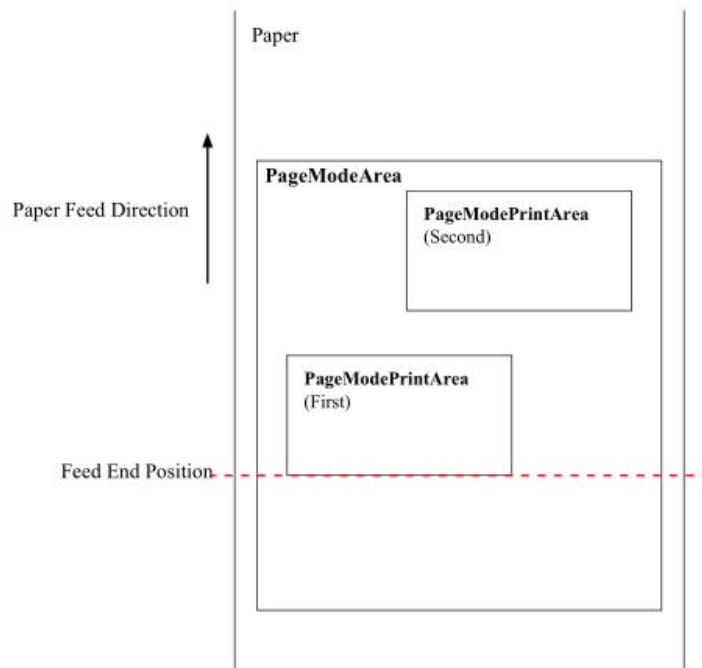


Return value

If the command was wrote successfully, return value TRUE, or FALSE

Remarks

Following picture show the concept of printing area



PAGE_DrawString

drawing the text

Syntax

```
BOOL PAGE_DrawString(TCHAR * pszString, int x, int y, int nWidthScale, int nHeightScale, int nFontType, int nFontStyle)
```

Parameters

pszString

content need to print。 UNICODE code strings

x

specify number of dots from starting area of horizontal direction to left margin of printing area. (horizontal axis)

support left center right alignment

incoming x see table

x	implication
-1	Left alignment
-2	Center alignment
-3	Right alignment
Great than or equal to 0	Horizontal direction

y

specify number of dots from starting area of vertical direction to upper edge of printing area.(vertical direction)

nWidthScale

specified width magnification [0,7]

nHeightScale

specified height magnification [0,7]

nFontType

font type

0 standard font

1 compressed font

nFontStyle

specify font styles, can be one or more of following table,(add together)

Value	Meaning
0x00	normal
0x08	Bold
0x80	1 dot bold underline
0x100	2 dot bold underline
0x200	invert (only valid at the head of first line)
0x400	reverse (black and white)
0x1000	rotate each character 90 degrees clockwise

Return value

If the command was wrote successfully, return value TRUE, or FALSE

Remarks

Page mode Draw functions won' t print at once, just fill in the page, till call PAGE_PagePrint , it will print.

PAGE_DrawRect

draw rectangle

Syntax

```
BOOL PAGE_DrawRect(int x, int y, int nWidth, int nHeight, int nColor)
```

Parameters

x

specify number of dots from starting area of horizontal direction to left margin of printing area.(horizontal direction)

y

specify number of dots from starting area of vertical direction to upper edge of printing area.(vertical direction)

nWidth

specify the width of the rectangle

nHeight

specify the height of the rectangle

nColor

specify the color of the rectangle

0 white

1 black

Return value

If the command was wrote successfully, return value TRUE, or FALSE

Remarks

If want to draw lines, just need to set the width to 1, (if want to draw bolder line, set the number larger) will be OK.

Notes: please don't draw too large area rectangle, or the printer will reset as the power problem.

PAGE_DrawBarcode

Draw barcode

Syntax

```
BOOL PAGE_DrawBarcode(TCHAR * pszBarcodeContent, int x, int y, int nBarcodeUnitWidth, int nBarcodeHeight, int nHriFontType, int nHriFontPosition, int nBarcodeType)
```

Parameters

pszBarcodeContent

Barcode contents

x

It assigns the dots from the start point position to printing area margin in horizontal direction.(Horizontal coordinates)

It supports left justifying, mediate, right justifying

Introduce x as below chart

x	implication
-1	Left alignment
-2	Center alignment
-3	Right alignment
Great than or equal to 0	Horizontal direction

y

It assigns the dots from start point position to printing area margin in vertical direction(vertical coordinates)

nBarcodeUnitWidth

It assigns the basic element width.

It can be any value(n) in the chart as below

n	Single basic module width(successive type)	Double basic module width(discrete type)	
		Narrow element width	Wide element width
2	0. 25mm	0. 25mm	0. 625mm
3	0. 375mm	0. 375mm	1. 0mm
4	0. 5mm	0. 5mm	1. 25mm
5	0. 625mm	0. 625mm	1. 625mm
6	0. 75mm	0. 75mm	1.875mm

nBarcodeHeight

Barcode height

nHriFontType

It assigns HRI (Human Readable Interpretation) character font types.

It can be any value (n) of the chart as below.

Value	Meaning
0x00	standard ASCII
0x01	compressed ASCII

nHriFontPosition

It assigns HRI (Human Readable Interpretation) character position

It can be any value (n) of the chart as below.

Value	Meaning
0x00	Doesn't print
0x01	Only print in upward side of barcode
0x02	Only print in downward side of barcode
0x03	Print both in upward side and downward

nBarcodeType

It can be any value as below chart

Value	Meaning
0x41	UPC-A
0x42	UPC-C
0x43	JAN13(EAN13)
0x44	JAN8(EAN8)
0x45	CODE39
0x46	ITF
0x47	CODEBAR
0x48	CODE93
0x49	CODE 128

Return value

If the command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks

PAGE_DrawQRCode

Draw QR code

Syntax

```
BOOL PAGE_DrawQRCode(TCHAR * pszContent, int x, int y, int nQRCodeUnitWidth, int nVersion,  
int nEcLevel)
```

Parameters

pszContent

QR code text

x

It assigns the dots from the start point position to printing area margin in horizontal direction.(Horizontal coordinates)

It supports left justifying, mediate, right justifying

Introduce x as below chart

x	implication
-1	Left alignment
-2	Center alignment
-3	Right alignment
Great than or equal to 0	Horizontal direction

y

It assigns the dots from start point position to printing area margin in vertical direction(vertical coordinates)

nQRCodeUnitWidth

QR code unit width, the range is [1,16].

QR code unit is wider, QR code is bigger.

nVersion

QR code version. 0 means to calculate version automatically.

QR code version is bigger, the edited characters are more, and the QR code is bigger.

nEcLevel

QR code error correction level.[1,4]

Return value

If the command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks

PAGE_DrawBitmap

Draw BMP bitmap

Syntax

```
BOOL PAGE_DrawBitmap(TCHAR * FileName, int x, int y, int dwWidth, int dwHeight)
```

Parameters

FileName

Bitmap file route

x

It assigns the dots from the start point position to printing area margin in horizontal direction.(Horizontal coordinates)

It only supports left justifying, mediate, right justifying

Introduce x as below chart

x	implication
-1	Left alignment
-2	Center alignment
-3	Right alignment
Great than or equal to 0	Horizontal direction

y

It assigns the dots from start point position to printing area margin in vertical direction(vertical coordinates)

dwWidth

The printed width

dwHeight

The printed height

Return value

If the command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks

Pos Function

POS_TextOut

Print text

Syntax

BOOL POS_TextOut(TCHAR * pszString, int x, int nWidthScale, int nHeightScale, int nFontType, int nFontStyle)

Parameters

pszString

The printed contents. UNICODE coded string.

x

It assigns the dots from the start point to left margin in horizontal direction.

nWidthScale

It assigns the width magnification times [0,7]

nHeightScale

It assigns the height magnification times [0,7]

nFontType

Font type

0 standard font

1 constrigent font

nFontStyle

It assigns the font style, and can be any one or several meanings as below(add together)

Value	Meaning
0x00	Normal
0x08	Bold
0x80	1 dot bold underline
0x100	2 dots bold underline
0x200	Convert(only ca be efficient in the start of the line)
0x400	Invert(Black in white)
0x1000	Every character rotates clockwise 90°

Return value

If the command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks

POS_TextOut doesn't print immediately, and need to call POS_FeedXXX series function, and can print the contents in the buffer area.

POS_SetBarcode

Print barcode

Syntax

```
BOOL POS_SetBarcode(TCHAR * pszBarcodeContent, int nBarcodeUnitWidth, int nBarcodeHeight,  
int nHriFontType, int nHriFontPosition, int nBarcodeType)
```

Parameters

pszBarcodeContent

Barcode contents

nBarcodeUnitWidth

It assigns the code basic element width.

It can be any value (n) of the chart as below.

n	Single basic module width(succe ssive type)	Double basic module width(discrete type)	
		Narrow element width	Wide element width
2	0. 25mm	0. 25mm	0. 625mm
3	0. 375mm	0. 375mm	1. 0mm
4	0. 5mm	0. 5mm	1. 25mm
5	0. 625mm	0. 625mm	1. 625mm
6	0. 75mm	0. 75mm	1.875mm

nBarcodeHeight

Barcode height

nHriFontType

It assigns HRI (Human Readable Interpretation) character font types.

It can be any value (n) of the chart as below

Value	Meaning
0x00	standard ASCII
0x01	compressed ASCII

nHriFontPosition

It assigns HRI (Human Readable Interpretation) character position

It can be any value (n) of the chart as below.

Value	Meaning
0x00	Doesn' t print
0x01	Only print in upward side of barcode
0x02	Only print in downward side of barcode
0x03	Print both in upward side and downward

nBarcodeType

It can be any value (n) of the chart as below.

Value	Meaning
0x41	UPC-A
0x42	UPC-C
0x43	JAN13(EAN13)
0x44	JAN8(EAN8)
0x45	CODE39
0x46	ITF
0x47	CODEBAR
0x48	CODE93
0x49	CODE 128

Return value

If the command is written successfully, it returns TRUE. Or it returns FALSE

Remarks

POS_SetQRCode

Print QR code

Syntax

```
BOOL POS_SetQRCode(TCHAR * pszContent, int nQRCodeUnitWidth, int nVersion, int nEcLevel)
```

Parameters

pszContent

QR code text

nQRCodeUnitWidth

QR code unit width, the range is [1,16]。

QR code unit width is wider, then QR code is bigger.

nVersion

QR code version. 0 means to calculate version automatically.

QR code version is bigger, the edited characters are more, and QR code is bigger.

nEcLevel

QR code error correcting level. [1,4]

Return value

If the command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks

POS_PrintPicture

Print BMP bitmap

Syntax

```
BOOL POS_PrintPicture(TCHAR * FileName, DWORD dwWidth, DWORD dwHeight)
```

Parameters

FileName

Full route of bitmap file

dwWidth

The printed width

dwHeight

The printed height

Return value

If the command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks

POS_SelfTest

Print self-test page

Syntax

BOOL POS_SelfTest()

Parameters

Return value

If the command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks

POS_Query

Query printer status

Syntax

```
BOOL POS_Query(unsigned char status[1])
```

Parameters

status

Printer status

The value has no meaning at the moment

Return value

If the query status is successful, it returns TRUE. Or it returns to FALSE

Remarks

This command returns TRUE to show the printer is under printable status.

POS_RTQuery

real-time query printer status

Syntax

BOOL POS_RTQuery(unsigned char status[4])

Parameters

status

Printer status

The bit meanings are as below chart:

status[0]: Printer status

Bit	0/ 1	Hex.	Dec.	Function
0	0	00	0	Fixed is 0
1	1	02	2	Fixed is 1
2	0	00	0	One or two cashboxes are open(Fixed is 0 if this machine is without cashbox)
	1	04	4	Two cashboxes are closed
3	0	00	0	Online
	1	08	8	Offline
4	1	10	16	Fixed is 1
5, 6		--	--	Undefined
7	0	00	00	Paper has been tore off
	1	80	96	Paper hasn't been tore off

status[1]: Transmit offline status

Bit	0/ 1	Hex.	Dec.	Function
0	0	00	0	Fixed is 0
1	1	02	2	Fixed is 1
2	0	00	0	Upper cover is closed
	1	04	4	Upper cover is open
3	0	00	0	Paper feed key is un-pressed
	1	08	8	Paper feed key is pressed
4	1	10	16	Fixed is 1
5	0	00	0	Paper is not out

	1	20	32	Paper out
6	0	00	00	No error
	1	40	64	Error
7	0	00	0	Fixed is 0

status[2]: Transmit error status

Bit	0/ 1	Hex.	Dec.	Function
0	0	00	0	Fixed is 0
1	1	02	2	Fixed is 1
2		--	--	Undefined
3	0	00	0	Cutter has no error
	1	08	8	Cutter has error
4	1	10	16	Fixed is 1
5	0	00	0	No recoverable error
	1	20	32	Has recoverable error
6	0	00	00	Printer head temp. and voltage are normal
	1	40	64	Printer head temp. or voltage is out of range
7	0	00	0	Fixed is 0

status[3]: Transmit paper sensor status

Bit	0/ 1	Hex.	Dec	Function
0	0	00	0	Fixed is 0
1	1	02	2	Fixed is 1
2, 3	0	00	0	With paper
	1	0C	12	Paper will be out
4	1	10	16	Fixed is 1
5, 6	0	00	0	With paper
	1	60	96	Paper out
7	0	00	0	Fixed is 0

Return value

If the status query is successful, it returns TRUE. Or it returns FALSE

Remarks

The printer returns immediately when it receives this command anytime

POS_TicketSucceed

Receipt printing result inquiry.

Syntax

```
BOOL POS_TicketSucceed(int dwSendIndex)
```

Parameters

dwSendIndex

Receipt index

It can increase from 1 orderly, and has no actual meaning

Return value

The receipt finishes the printing, and breaks off as paper out. Then it returns TRUE.

Or, it doesn't detect the status, or returns to finish printing as paper out or other errors. Then it returns FALSE.

Remarks

Ensure the reality of the receipt printing, pls call the receipt printing result the function to make sure the receipt once after you finish the print every time.

POS_FeedLine

Syntax

BOOL POS_FeedLine()

Parameters

Return value

If command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks

Printer feeds paper one line

POS_FeedNLine

Syntax

BOOL POS_FeedNLine(int nLine)

Parameters

nLine

Paper feed lines

Return value

If command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks

Feed paper nLine lines

POS_FeedNDot

Syntax

```
BOOL POS_FeedNDot(int nDot)
```

Parameters

nDot

Paper feed dots

Return value

If command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks

Feed paper nDot dots. Generally, 1mm has 8 dots.

POS_SetMotionUnit

Set horizontal and vertical movement unit

Syntax

```
BOOL POS_SetMotionUnit(int nHorizontal, int nVertical)
```

Parameters

nHorizontal

Horizontal movement unit

nVertical

Vertical movement unit

Return value

If command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks

When nHorizontal and nVertical are both set as 200,1 point is 0.125mm

POS_SetLineHeight

Set line height

Syntax

```
BOOL POS_SetLineHeight(int nDistance)
```

Parameters

nDistance

Line height

Return value

If command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks

POS_SetRightSpacing

Set character right space

Syntax

```
BOOL POS_SetRightSpacing(int nDistance)
```

Parameters

nDistance

Character right space

Return value

If command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks

POS_SetAlign

Set alignment

Syntax

```
BOOL POS_SetAlign(int nAlign)
```

Parameters

nAlign

Alignment method

- 0 Left justifying
- 1 Middle justifying
- 2 Right justifyin

Return value

If command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks

POS_Reset

Reset printer will empty setting.

Syntax

```
BOOL POS_Reset()
```

Parameters

Return value

If command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks

POS_KickOutDrawer

Turn on cashbox

Syntax

```
BOOL POS_KickOutDrawer(int nID, int nOnTimes, int nOffTimes);
```

Parameters

nID

Cashbox serial no.

0 Cashbox pin 2

1 Cashbox pin 5

nOnTimes

Cashbox pulse high potential ms time

nOffTimes

Cashbox pulse low potential ms time

Return value

If command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks

POS_CutPaper

Cut paper directly

Syntax

```
BOOL POS_CutPaper(int nMode)
```

Parameters

nMode

Paper cut mode

0 Full cut

1 Half cut

Return value

If command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks

POS_FeedAndCut

Printer feeds paper [the distance from the printing position to cutter + nDistance X (horizontal movement unit)], then cut paper.

Syntax

```
BOOL POS_FeedAndCut(int nDistance)
```

Parameters

nDistance

Extra paper feed distance

Return value

If command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks

POS_Beep

Buzzer call

Syntax

```
BOOL POS_Beep(int nBeepCount, int nBeepMillis)
```

Parameters

nBeepCount

Calling times

nBeepMillis

Calling time:100ms is the unit

Return value

If command is written successfully, it returns TRUE. Or it returns FALSE.

Remarks