

POSTEK PPLE

API Manual

Version 3.0

2019

Description of API functions file

Name: CDFPSK.dll

Version number: 3. 0.0.6170522

Copyright: ©2019 by Postek Electronics CO., LTD. All rights reserved.

Purpose

This API functions provide a series of commands/functions for users of Postek Electronics CO., LTD. All the functions offer convenience for them when they compile the application programs based on the operating system of Windows9X, NT, 2000, Windows7, Windows 8 and many more.

This API functions only supports products of Postek Electronics CO., LTD.

Abbreviation comparison

PPLI: The first set of Printer Program Language of the Postek Electronics CO., LTD.

API: Application Program Interface.

Dots: Pixel is a kind dimensional unit used in computer science technology. Originally means the minimum unit of the TV imaging. The minimum imaging unit for a printer: Dots are equal to one inch divided by the maximum resolution of the printer.

- For the 203 DPI printer, 1dot = $25.4\text{mm}/203 = 0.125\text{mm}$ (1dot = $1000 / 203 = 5\text{mil}$);
- For the 300 DPI printer, 1dot = $25.4\text{mm}/300 = 0.085\text{mm}$ (1dot = $1000 / 300 = 3\text{mil}$).

TrueType Font: based on the Windows operating system, can be loaded or unloaded.

-All installed TrueType Font can be used by this function.

Notice

String

The quotation mark character (") designates the beginning and ending of a string.

The back slash character (\) designates that the character following is a literal and will encode into the data field.

Refer to the following examples:

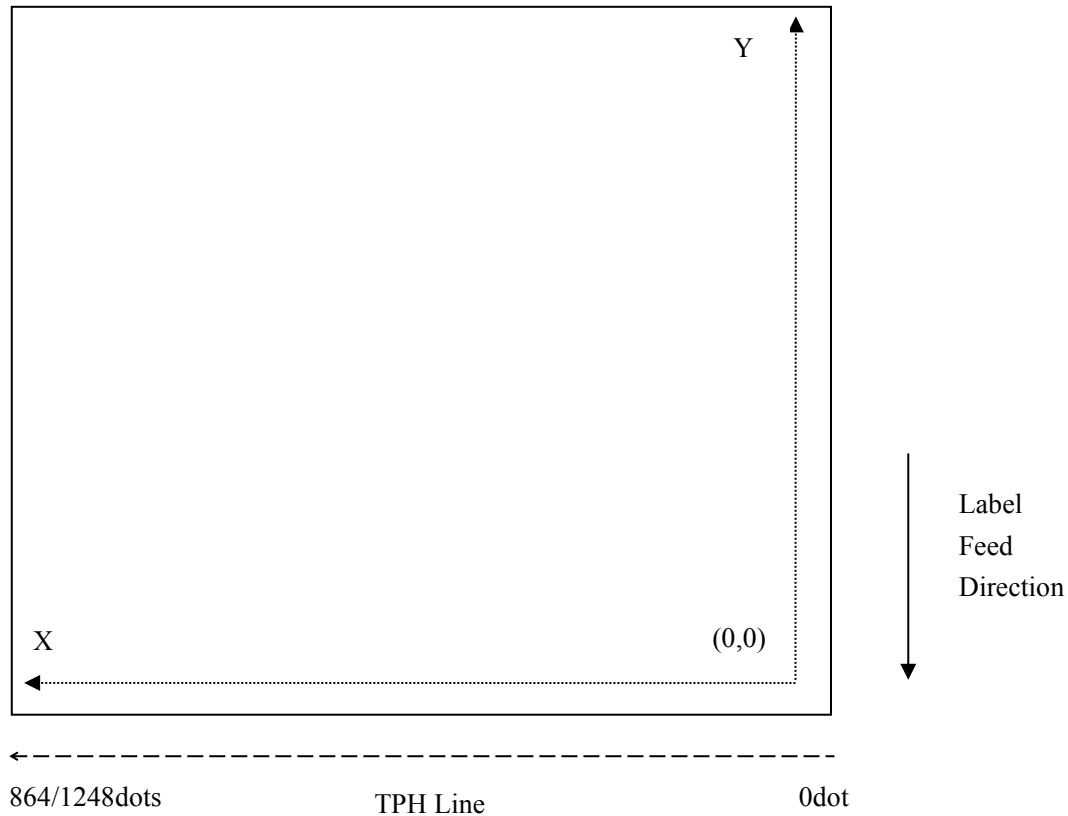
<u>character</u>	<u>Enter into Data Field</u>
"	\"
\	\\
0x00 – 0x7F	\x00 - \x7F

* All print commands and alpha character command, Parameters are case sensitive.

* <CR> is “13” of the USASCII decimal system, or “0DH” of the hexadecimal system, namely “enter” symbol.

Coordinate system

The barcode label printer’s coordinates system is depicted in following figure:



Functions

Function name	Description
OpenPort	Open the communication port.
ClosePort	Close the communication port that opened by OpenPort function.
SetPCComPort	Set the baudrate speed of the serial port in PC.
PTK_GetErrState	Check if there's any error after using other functions in CDFPSK.dll.
PTK_GetInfo	Obtain the edition information of this API functions.
PTK_ClearBuffer	Clear the contents in printer buffer.
PTK_SetDarkness	Set the darkness of the TPH.
PTK_SetPrintSpeed	Set printing speed.
PTK_SetLabelHeight	Set label's height and the height of media gap/black line/ hole.
PTK_SetLabelWidth	Set label's width.
PTK_SetDirection	Set label printing direction.
PTK_SetCoordinateOrigin	Set/change the coordinate reference point.
PTK_PrintLabel	Order the printer to execute printing work.
PTK_DrawText	Print text string.
PTK_DrawTextEx	Print text letters, the content can be constant, counter value, variable or combined string.
PTK_DrawTextTrueTypeW	Print TrueType Font, the width and height of the characters can be adjusted.
PTK_DrawBarcode	Print bar code.
PTK_DrawBarcodeEx	Print a barcode, the content can be constant, counter value, variable or combined string.
PTK_DrawBar2D_DATAMATRIX	Print a DataMatrix 2D barcode. Output in printer command.
PTK_DrawBar2D_QR	Print a QR 2D barcode. Output in printer command.
PTK_DrawBar2D_QREx	Print a QR 2D barcode. Output in Graphics.
PTK_DrawBar2D_MaxiCode	Print a MaxiCode. Output in printer command.
PTK_DrawBar2D_Pdf417	Print a PDF417 2D barcode.
PTK_DrawBar2D_HANXIN	Print a HANXIN 2D barcode.
PTK_PcxGraphicsList	Print the PCX graphics name list stored in the RAM or flash memory.
PTK_PcxGraphicsDel	Delete one or all of the graphics stored in the printer.
PTK_PcxGraphicsDownload	Store a PCX graphic to the printer.
PTK_DrawPcxGraphics	Print the designated graphics.
PTK_PrintPCX	Print a PCX graphic.

<u>PTK_BmpGraphicsDownload</u>	Convert a graphic in BMP format to PCX format, and then download the PCX graphic into the printer memory.
<u>PTK_BinGraphicsList</u>	Print the Bin and PCX graphics name list stored in the RAM or flash memory.
<u>PTK_BinGraphicsDel</u>	Delete one or all of the graphics stored in the printer.
<u>PTK_BinGraphicsDownload</u>	Store a Bin graphic to the printer.
<u>PTK_RecallBinGraphics</u>	Print a stored Bin graphic
<u>PTK_DrawBinGraphics</u>	Print a binary graphic in data format.
<u>PTK_DrawRectangle</u>	Draw a Rectangle.
<u>PTK_DrawLineXor</u>	Draw straight-line (If two straight-lines intersects, use an exclusive OR operation).
<u>PTK_DrawLineOr</u>	Draw straight-line (If two straight-lines intersects, use an OR operation).
<u>PTK_DrawDiagonal</u>	Draw diagonal.
<u>PTK_DrawWhiteLine</u>	Draw white straight-line.
<u>PTK_SoftFontList</u>	Print the soft font list stored in the RAM or flash memory.
<u>PTK_SoftFontDel</u>	Delete one or all of the soft fonts stored in the RAM or flash memory.
<u>PTK_FormList</u>	Print the form name list stored in the printer.
<u>PTK_FormDel</u>	Delete one or all of the forms stored in the printer.
<u>PTK_FormDownload</u>	Store a form to the printer, it must be used together with the PTK_FormEnd.
<u>PTK_FormEnd</u>	End form store, it must be used together with the PTK_FormDownload.
<u>PTK_ExecForm</u>	Execute the designated form.
<u>PTK_DefineCounter</u>	Define counter variable.
<u>PTK_DefineVariable</u>	Define variable.
<u>PTK_Download</u>	Download variable or counter variable.
<u>PTK_DownloadInitVar</u>	Initialize Variable or counter variable.
<u>PTK_PrintLabelAuto</u>	Auto-execute the printing work.
<u>PTK_SendFile</u>	Send a command file to the printer.
<u>PTK_GetUSBID</u>	Get the USB ID of the printer.
<u>PTK_DisableBackFeed</u>	Disable Tear-off mode for the printer.
<u>PTK_EnableBackFeed</u>	Enable Tear-off mode for the printer.
<u>PTK_PrintConfiguation</u>	The current setting for printer/printer's working state.
<u>PTK_SetPrinterState</u>	Set the printer's working state.
<u>PTK_DisableErrorReport</u>	Disable the error report function from taking effect.
<u>PTK_EnableErrorReport</u>	Enable the error report function
<u>PTK_EnableFLASH</u>	Select the flash memory.

<u>PTK_DisableFLASH</u>	Cancel Select the flash memory.
<u>PTK_FeedMedia</u>	Feed one label.
<u>PTK_MediaDetect</u>	Do Calibrate
<u>PTK_CutPage</u>	Set cutter's working cycle (namely the cutter will start to cut labels once how many labels have been printed)
<u>PTK_CutPageEx</u>	Set the cutter frequency
<u>PTK_FeedBack</u>	Require an error report from printer immediately.
<u>PTK_Reset</u>	Reset the printer.
<u>PTK_ErrorReport</u>	Send ^ee command to the printer and get an error code via the serial port.
<u>PTK_ErrorReportUSB</u>	Send ^ee command to the printer and get an error code via the USB port.
<u>PTK_RWRFLIDLabel</u>	Read/Write RFID tag.
<u>PTK_SetRFLabelPWAndLockRFLabel</u>	Lock and Set password for RFID tag.
<u>*PTK_SetRFIDLabelRetryCount</u>	Set RFID retries.
<u>PTK_SetRFID</u>	RFID setup
<u>PTK_SetFontGap</u>	Specify font gap for downloaded soft fonts.
<u>*PTK_SetBarCodeFontName</u>	Specify font name for barcode human readable text.
<u>PTK_RenameDownloadFont</u>	Specify an ID for a downloaded font, the ID can be used as Parameters "p4" of PTK_DrawTextEx().
<u>PTK_SetCharSets</u>	Set the character set for the label.
<u>PTK_ReadRFIDTagData</u>	Read EPC/TID data from RFID tag via designate Serial Port.
<u>PTK_ReadRFIDTagDataUSB</u>	Read EPC/TID data from RFID tag via designate USB Port.
<u>PTK_ReadRFIDTagDataNet</u>	Read EPC/TID data from RFID tag via designate Ethernet Port.
<u>PTK_Connect</u>	Establish internet connection.
<u>PTK_CloseConnect</u>	Close internet connection port.
<u>PTK_ErrorReportNet</u>	Get printer error report via TCP/IP port.
<u>PTK_ReadPrintConifUSB</u>	Get printer info via USB port.
<u>PTK_ReadPrintConifNet</u>	Get printer info via TCP/IP port.
<u>PTK_RFIDCalibrate</u>	Perform RFID calibration (only available for printer firmware version higher than V1.73).

The function with sign * cannot be used for the moment.

Expounding of the Function

OpenPort

Description:

This function is used to open the communication port.

Be sure the OpenPort function has been correctly executed before using other functions.

Syntax:

```
int OpenPort(int PortFlag);
```

Parameters:

xx: The communication port's name in windows;

0: Print to file PBuff.txt (Create file under the program's execution directory); (Pls do not use this port!)

1: Open LPT1;

2: Open LPT2;

3: Open LPT3;

4: Open COM1;

5: Open COM2;

6: Open COM3;

7: Open USB001;

8: Open USB002;

9: Open USB003;

...

255: When there is only one Postek printer connects to PC, the PC will open the communication port of this printer by default.

Returns:

0 -> OK;

Other returns: Please refer to chapter: CDFPSK.DLL erroneous return value specification.

Example:

```
OpenPort(1);    // symbols opening the port LPT1.
```

ClosePort

Description:

This function is used to close the communication port which opened by the OpenPort function.

It is suggested that call the ClosePort to close the communication port after completing all the operation operated by other functions. Otherwise, your program will always take up the opened communication port

until the program be closed.

Syntax:

```
int ClosePort(void);
```

Parameters: None

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
ClosePort( );
```

SetPCComPort

Description:

This function is used to set the baudrate speed of the serial port in PC.

It is only available when use the serial port.

Note:

Be sure to corresponded with the serial baudrate setting in your printer (by adjusting the DIP's 7, 8PIN on-off, please refer to the user's manual).

Syntax:

```
int SetPCComPort(DWORD BaudRate, BOOL HandShake);
```

Parameters:

BaudRate:set serial port baudrate, values:
9600,19200,38400,57600;

HandShake:whether uses HandShaking or not;
TRUE:enable HandShaking,
FALSE:disable HandShaking.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
SetPCComPort ( 9600, TRUE);
```


PTK_GetErrState

Description:

This function is used to check the correctness/validity of the other functions in CDFPSK.dll.

Please refer to the section of CDFPSK.dll description of returning errors for error codes.

This function must use before the Closeport().

Syntax:

```
int GetErrState(void);
```

Parameters: None

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
int state = 0;
OpenPort(1);
...
State = GetErrState();
...
ClosePort();
```

PTK_GetInfo

Description:

This function is used to get the edition information of this API DLL.

Syntax:

```
int PTK_GetInfo(void)
```

Parameters: None

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_GetInfo (void)
```

PTK_ClearBuffer

Description:

This function is used to clear the contents in printer buffer. It suggest that use this function to clear the contents in printer buffer before sending a new label to the printer.

Please do not use this function during editing the FORM.

Syntax:

```
int PTK_ClearBuffer (void);
```

Parameters: None.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_ClearBuffer ( );
```

PTK_SetDarkness

Description:

This function is used to set darkness of the TPH.

Syntax:

```
int PTK_SetDarkness (unsigned int id);
```

Parameters:

id: Acceptable values are from 0 to 20, the default value is 10.

This value is not in deed the temperature of the TPH, it is a relative value. The lightest print darkness is achieved with a value of 0 and the greatest print darkness is achieved with a value of 20.

Returns:

0-> OK;

Other returns: Please refer to chapter:CDFPSK.dll erroneous return value specification.

Example:

```
PTK_SetDarkness (10);
```

PTK_SetPrintSpeed

Description:

This function is used to set the print speed.

Syntax:

```
int PTK_SetPrintSpeed (unsigned int px);
```

Parameters:

px: Value range :0 – 10, or 10 – 100.

px Value	Speed	EPL2(compatible)
10	1.0 ips (25.40 mm/s)	0 or 1
15	1.5 ips (38.10 mm/s)	
20	2.0 ips (50.80 mm/s)	2
25	2.5 ips (63.50 mm/s)	
30	3.0 ips (76.20 mm/s)	3
35	3.5 ips (88.90 mm/s)	
40	4.0 ips (101.60 mm/s)	4
50	5.0 ips (127.00 mm/s)	5
60	6.0 ips (152.40 mm/s)	6
70	7.0 ips (177.80 mm/s)	7
80	8.0 ips (203.20 mm/s)	8
90	9.0 ips (228.60 mm/s)	9
100	10.0 ips (254.00 mm/s)	10

Returns:

0-> OK;

Other returns: Please make reference to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_SetPrintSpeed (5);
```

PTK_SetLabelHeight

Description:

This function is used to set the label's height and the height of media gap/black line/bore hole.

Syntax:

```
int PTK_SetLabelHeight (unsigned int lheight, unsigned int gapH);
```

Parameters:

lheight: Label height in dots. Value range: 0-65535.

gapH: The height of media gap/black line/bore hole in dots. Value range: 0-65535. The value of gapH is related to the position of labels.

Gap mode: By default, set gapH to gap length. The printer is in Gap mode and Parameters are

set with the media AutoSense.

Black Line Mode: Set gapH to B plus black line thickness in dots.

Continuous Media Mode: Set gapH to 0 (zero). The transmissive (gap) sensor will be used to detect the end of media.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_SetLabelHeight (160, 24);
```

PTK_SetLabelWidth

Description:

This function is used to set the label's width.

Syntax:

```
int PTK_SetLabelWidth (unsigned int lwidth);
```

Parameters:

lwidth: label width in dots .

Returns:

0-> OK;

Other returns: Please make reference to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_SetLabelWidth (250);
```

PTK_SetDirection

Description:

This function is used to set print orientation for all graphics, text, bar codes, lines and rectangles.

This function will change the direction of contents, such as text, barcode, straight line, and rectangle.

Syntax:

```
int PTK_SetDirection (char direct);
```

Parameters:

direct:Orientation; Acceptable values are B or T. The default value is T.

B: Print from bottom right corner.

T: Print from top left corner.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_SetDirection ('B');
```

PTK_SetCoordinateOrigin

Description:

This function is used to set/change the coordinate origin point.

Syntax:

```
int PTK_SetCoordinateOrigin (unsigned int px, unsigned int py);
```

Parameters:

px: X coordinate moved distance in dots.

py: Y coordinate moved distance in dots.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_SetCoordinateOrigin (12,23);
```

PTK_PrintLabel

Description:

This function is used to order the printer to execute the print work.

Note: Please don't use this function during editing the FORM, use the PTK_PrintLabelAuto() instead.

Syntax:

```
int PTK_PrintLabel (unsigned int number, unsigned int cpnumber);
```

Parameters:

number: Number of label sets, 1—65535.

cpnumber: Number of copies per label, 1—65535.

If the cpnumber have no set value, it will default 1.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_PrintLabel (2,3);
```

PTK_DrawText

Description:

This function is used to print text, the content can be constant, counter value, variable or combined string.

Syntax:

```
int PTK_DrawText ( unsigned int   px,  
                  unsigned int   py,  
                  unsigned int   pdirec,  
                  unsigned int   pFont,  
                  unsigned int   pHorizontal,  
                  unsigned int   pVertical,  
                  char ptext,  
                  LPTSTR pstr );
```

Parameters:

px: X coordinate in dots.

py: Y coordinate in dots.

pdirec: select printing direction. 0—No rotation; 1—90°rotation; 2—180°rotation; 3—270°rotation.

pFont: Selects internal fonts or softfonts, 1—5: internal fonts; A—Z: downloaded soft fonts.

a: 24*24 simple Chinese fonts in printer.

Value	Description
1	Foreign language fonts 1
2	Foreign language fonts 2
3	Foreign language fonts 3
4	Foreign language fonts 4
5	Foreign language fonts 5
a	24*24 dot matrix Chinese font
A~Z	Soft fonts

pHorizontal: Horizontal multiplier expands the text horizontally. Value range: 1-24

pVertical: Vertical multiplier expands the text vertically. Value range: 1-24

The acceptable values for both pHorizontal and pVertical are from 1 to 24.

ptext: Choosing 'N' prints normal text (i.e. black characters on a white background)

Choosing 'R' prints reversed text (i.e. white characters on a black background)
pstr: A character string (length is 1 to 100).

Returns:

0-> OK;

Other returns: Please make reference to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_DrawText (50, 30, 0, 2, 1, 1, 'N', "123456789");
```

PTK_DrawTextEx

Description:

This function is used to print text letters, the content can be constant, counter value, variable or combined string.

Syntax:

```
int PTK_DrawTextEx ( unsigned int  px, unsigned int  py,  
                    unsigned int  pdirec, unsigned int  pFont,  
                    unsigned int  pHorizontal, unsigned int  pVertical,  
                    char ptext, LPTSTR pstr , BOOL Variable);
```

Parameters:

px: X coordinate in dots.

py: Y coordinate in dots.

pdirec: select printing Direction. 0—No rotation;1—90°rotation; 2—180°rotation; 3—270°rotation.

pFont: Selects internal fonts or softfonts,1—5: internal fonts; A—Z: downloaded soft fonts.

a: 24*24 simple Chinese fonts in printer.

Value	Description
1	Foreign language fonts1
2	Foreign language fonts 2
3	Foreign language fonts 3
4	Foreign language fonts 4
5	Foreign language fonts 5
a	24*24 dot matrix Chinese font
A~Z	Soft fonts

pHorizontal: Horizontal multiplier expands the text horizontally. Value range: 1-24

pVertical: Vertical multiplier expands the text vertically. Value range: 1-24

The acceptable values for both pHorizontal and pVertical are from 1 to 24.

pText: Choosing 'N' prints normal text (i.e. black characters on a white background)

Choosing 'R' prints reversed text (i.e. white characters on a black background)

pStr: A character string (length is 1 to 100), using "DATA", Cn and Vn Parameters

"DATA": A data fixed string, it must begin and end with ""Example:"POSTEK Printer".

Cn: A counter value. Refer to C command.

Vn: A variable string. Refer to V command.

Example: "data1"CnVn "data2".

Variable: TRUE suggests that variable is included in this string, add '\'" to enclose the constant string.

For example: PTK_DrawTextEx (50, 30, 0, 2, 1, 1, 'N', "\'123456789\'C0", TRUE);

FALSE suggests that no variable is included in this string, and there is no need to add '\'".

For example, the following two commands have the same effect:

PTK_DrawTextEx (50, 30, 0, 2, 1, 1, 'N', "123456789", FALSE);

PTK_DrawText(50, 30, 0, 2, 1, 1, 'N', "123456789")

Returns:

0-> OK;

Other returns: Please make reference to chapter: CDFPSK.dll erroneous return value specification.

Example:

PTK_DrawTextEx (50, 30, 0, 2, 1, 1, 'N', "\'123456789\'C0", TRUE);

PTK_DrawTextEx (50, 30, 0, 2, 1, 1, 'N', "123456789", FALSE);

PTK_DrawTextEx (50, 30, 0, 2, 1, 1, 'N', "C1", TRUE);

PTK_DrawTextEx (50, 30, 0, 2, 1, 1, 'N', "V3", TRUE);

PTK_DrawTextEx (50, 30, 0, 2, 1, 1, 'N', "\'Printer\' C2V1\'is ok.\'", TRUE);

Print counter and variable string (usually used in Form):

PTK_FormDel("LSFORM"); //Delete Form "TEST"

PTK_FormDownload("LSFORM"); //Save Form "TEST"

PTK_DefineCounter(0,10,78,"+1","InputC0"); //Define a counter variable C0

PTK_DefineCounter(1,10,78,"+1","InputC1"); // Define a counter variable C1

PTK_DefineVariable(0,16,78,"InputV0"); //Define a variable string V0, and name it as :V'+
the first Parameters

PTK_DrawBarcodeEx(100,100,0,"3",2,6,30,66,"C0",true); //Print a counter barcode

PTK_DrawTextEx(100,160,0,2,1,1,78,"" contents of barcode: \'C0",true); //Print text letters, the string
consists of constant and variable C0

PTK_DrawTextEx(100,220,0,2,1,1,78,"" print counter: \'C1",true); //Print counter value C1

PTK_DrawTextEx(100,280,0,2,1,1,78,"" print variable: \'V0",true); //print variable string V0

PTK_DrawTextEx(100,340,0,2,1,1,78,"" combined function:

counter: \'C1\' variable: \'V0",true); //combined printing

PTK_FormEnd(); //End up with saving Form

PTK_ExecForm("LSFORM"); // Run the appointed Form:LSFORM


```
PTK_Download(); // Download variable or counter variable
// Initialize in order of counter and the defined order of variable( The defined order for this example is
C0,C1,V0)
PTK_DownloadInitVar("12345678"); // Initialize counter variable C0
PTK_DownloadInitVar("123456"); // Initialize counter variable C1
PTK_DownloadInitVar("1111"); // Initialize variable V0

PTK_PrintLabel(2,1); // Order the printer to execute printing work.
```

PTK_DrawTextTrueTypeW

Note:

Be sure to install the printer driver of POSTEK prior to using PTK_DrawTextTrueTypeW()

Description:

This function is used to print a group TrueType Font characters. And the character width and height can be adjusted.

Syntax:

```
int PTK_DrawTextTrueTypeW( int x, int y,
                           int FHeight, int FWidth,
                           LPCTSTR FType, int Fspin,
                           int FWeight, BOOL FItalic,
                           BOOL FUnline, BOOL FStrikeOut,
                           LPCTSTR id_name, LPCTSTR data )
```

Parameters:

px: X coordinate in dots.

py: Y coordinate in dots.

FHeight: Font type height in dots.

FWidth: Font type width in dots.

*** If you want to print the normal scale font, set FWidth to 0.**

FType: Font type name.

Fspin: Font rotation. 1—No rotation; 2—90°rotation; 3—180°rotation; 4—270°rotation.

Fweight: Font thickness.

0 and 400 -> 400 standard,

100 -> extra thin, 200 -> tiny thin,

300 -> thin, 500 -> middling,

600 -> half thick, 700 -> thick,

800 -> extra thick, 900 -> bolder.

Fitalic: Italic, 0 -> FALSE, 1 -> TRUE.

Funline: Add base line, 0 -> FALSE, 1 -> TRUE.

FstrikeOut: Add strikethrough, 0 -> FALSE, 1 -> TRUE.

id_name: Identify the name. True type characters will be transferred to PCX data and store to the printer

as the name of id_name. Users can Call id_name to print this true type characters by PTK_DrawPcxGraphics()for several times before powering off.
data: The contents of string.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

Print Chinese fonts for 3mm heigh:

For the 203 DPI models printer, FHeight is $3 / 0.125 = 24\text{dots}$;

For the 300 DPI models printer, FHeight is $3 / 0.08 = 38\text{dots}$ (round).

PTK_DrawTextTrueTypeW (30, 35, 24, 0, “Arial”, 4, 400, 0, 0, 0, “A1”, “Classified”);

PTK_DrawBarcode

Description:

This function is used to print a bar code.

Syntax:

```
int PTK_DrawBarcode ( unsigned int px, unsigned int py,  
                      unsigned int pdirec, LPTSTR pCode,  
                      nsigned int NarrowWidth, unsigned int pHorizontal,  
                      unsigned int pVertical, char ptext, LPTSTR pstr );
```

Parameters:

px: X coordinate in dots.

py: Y coordinate in dots.

pdirec: select printing Direction. 0—No rotation;1—90°rotation; 2—180°rotation; 3—270°rotation.

pCode: Bar code selection

pCode Value	Barcode Type
0	Code 128 UCC (shipping container code)
1	Code 128 AUTO
1A	Code 128 subset A
1B	Code 128 subset B
1C	Code 128 subset C
1E	UCC/EAN
1F	EAN 128 subset A
1G	EAN 128 subset B
1H	EAN 128 subset C

2D	Interleaved 2 of 5 with human readable check digit
2G	German Postcode
2M	Matrix 2 of 5
2U	UPC Interleaved 2 of 5
3	Code 3 of 9
3C	Code 3 of 9 with check sum digit
3E	Extended Code 3 of 9
3F	Extended Code 3 of 9 with check sum digit
9	Code93
E30	EAN-13
E32	EAN-13 2 digit add-on
E35	EAN-13 5 digit add-on
E80	EAN-8
E82	EAN-8 2 digit add-on
E-85	EAN-8 5 digit add-on
K	Codabar
P	Postnet
UA0	UPC-A
UA2	UPC-A 2 digit add-on
UA5	UPC-A 5 digit add-on
UE0	UPC-E
UE2	UPC-E 2 digit add-on
UE5	UPC-E 5 digit add-on

NarrowWidth: Narrow bar width in dots.

pHorizontal: Wide bar width in dots.

pVertical: Bar code height in dots.

ptext: N-no print the readable characters under barcode.

B- print the readable characters under barcode

pstr: A character string (length is 1 to 100), using “DATA”, Cn and Vn Parameters

“DATA”: A data fixed string, it must begin and end with ‘ “ ’, for example “POSTEK Printer”.

Cn: A counter value. Refer to C command.

Vn: A variable string. Refer to V command.

Example: “data1”CnVn “data2”.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_DrawBarcode (50, 30, 0, '1A', 1, 1, 10, 'N', "123456");
```

```
PTK_DrawBarcode (50, 30, 0, '1A', 1, 1, 10, 'N', C2);
```

```
PTK_DrawBarcode (50, 30, 0, '1A', 1, 1, 10, 'N', V1);
```

```
PTK_DrawBarcode (50, 30, 0, '1A', 1, 1, 10, 'N', C1 " is" V2);
```

PTK_DrawBarcodeEx

Description:

This function is used to print a barcode.

Syntax:

```
int PTK_DrawBarcodeEx( unsigned int  px, unsigned int  py,  
                      unsigned int  pdirec, LPTSTR  pCode,  
                      unsigned int  NarrowWidth,  unsigned int pHorizontal,  
                      unsigned int pVertical, TCHAR ptext, LPTSTR pstr , BOOL Variable );
```

Parameters:

px: X coordinate in dots;

py: Y coordinate in dots;

pdirec: Select printing Direction; 0—No rotation; 1—90°rotation; 2—180°rotation; 3—270°rotation;

pCode: Barcode selection.

pCode Value	Barcode Type
0	Code 128 UCC (shipping container code)
1	Code 128 AUTO
1A	Code 128 subset A
1B	Code 128 subset B
1C	Code 128 subset C
1E	UCC/EAN
1F	EAN 128 subset A
1G	EAN 128 subset B
1H	EAN 128 subset C
2D	Interleaved 2 of 5 with human readable check digit
2G	German Postcode
2M	Matrix 2 of 5
2U	UPC Interleaved 2 of 5
3	Code 3 of 9
3C	Code 3 of 9 with check sum digit
3E	Extended Code 3 of 9
3F	Extended Code 3 of 9 with check sum digit
9	Code93
E30	EAN-13
E32	EAN-13 2 digit add-on
E35	EAN-13 5 digit add-on
E80	EAN-8
E82	EAN-8 2 digit add-on

E-85	EAN-8 5 digit add-on
K	Codabar
P	Postnet
UA0	UPC-A
UA2	UPC-A 2 digit add-on
UA5	UPC-A 5 digit add-on
UE0	UPC-E
UE2	UPC-E 2 digit add-on
UE5	UPC-E 5 digit add-on

NarrowWidth: Establishes the narrow bar width, in dots;

pHorizontal: Establishes the wide bar width, in dots;

pVertical: Establishes the barcode height, in dots;

ptext: Choosing "N" prints normal text (i.e. black characters on white background);

Choosing "R" prints reserved text (i.e. white characters on a black background);

"DATA": A data fixed string, it must begin and end with "" (i.e. "Postek Printer");

Cn: A counter value. Refer to C command.

Vn: A variable string. Refer to V command.

For example: "data1"CnVn "data2".

BOOL Variable:

TRUE suggests that variable is included in this string, add '\' to enclose the constant string.

For example: PTK_DrawTextEx (50, 30, 0, 2, 1, 1, 'N', "\123456789\C0", TRUE);

FALSE suggests that no variable is included in this string, and there is no need to add "\".

For example: **PTK_DrawBarcodeEx (50, 30, 0, '1A', 1, 1, 10, 'N', "123456", FALSE);**

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

PTK_DrawBarcodeEx (50, 30, 0, '1A', 1, 1, 10, 'N', "\123456\\"", TRUE);

PTK_DrawBarcodeEx (50, 30, 0, '1A', 1, 1, 10, 'N', "123456", FALSE);

PTK_DrawBarcodeEx (50, 30, 0, '1A', 1, 1, 10, 'N', C2, TRUE);

PTK_DrawBarcodeEx (50, 30, 0, '1A', 1, 1, 10, 'N', V1, TRUE);

PTK_DrawBarcodeEx (50, 30, 0, '1A', 1, 1, 10, 'N', C1 "is" V2, TRUE);

PTK_DrawBar2D_DATAMATRIX

Description:

This function is used to print a Datamatrix barcode.

Syntax:

PTK_DrawBar2D_DATAMATRIX (unsigned int x, unsigned int y,
 unsigned int w, unsigned int v,
 unsigned int o, unsigned int m,
 LPTSTR pstr)

* Parameters:

int	x;	●X coordinates.
int	y;	●Y coordinates. Note:1 dot = 0.125 mm.
int	w;	●Maximum printing width, in dots.
int	v;	●Maximum printing height, in dots.
int	o;	●Set rotate directions, value range: 0~3.
int	m;	●Set the multiplier values, in dots, value range: (1 - 9).
LPCTSTR	pstr;	●A character string.

Return value: 0 -> OK.

Reference Error.txt file.

PTK_DrawBar2D_QR

This function is used to print a QR barcode.

Syntax:

PTK_DrawBar2D_QR(unsigned int x, unsigned int y,
 unsigned int w, unsigned int v,
 unsigned int o, unsigned int r,
 unsigned int m, unsigned int g,
 unsigned int s, LPTSTR pstr)

* Parameters:	int	x;	●X coordinates.
	int	y;	●Y coordinates. Note:1 dot = 0.125 mm.
	int	w;	●Maximum printing width, in dots.
	int	v;	●Maximum printing height, in dots.
	int	o;	●Set the rotate direction, value range: 0~3.
	int	r;	●Set the multiplier values, in dots, value range: (1 - 9).
	int	m;	●Select the coding mode of QR, value range:(0 - 4).
	int	g;	●Select the error correction level of QR, value range: (0 - 3).
	int	s;	●Select the masking pattern of QR, value range: (0 - 8).
	LPCTSTR	pstr;	●A character string.

Return Value: 0 -> OK.

Reference Error.txt file.

LPTSTR pstr)

*Parameters:

int x;	●X coordinates.
int y;	●Y coordinates. Note: 1 dot = 0.125 mm.
int m;	●Mode [2 - 4];
int u;	●Data format: 0-None UPS format; 1-UPS format.
LPCTSTR pstr;	●A character string.

Return Value: 0 -> OK.

Reference Error.txt file.

PTK_DrawBar2D_Pdf417**Description:**

This function is used to print a PDF417 barcode.

Syntax:

```
int PTK_DrawBar2D_Pdf417(unsigned int x, unsigned int y,
                          unsigned int w, unsigned int v,
                          unsigned int s, unsigned int c,
                          unsigned int px, unsigned int py,
                          unsigned int r, unsigned int l,
                          unsigned int t, unsigned int o,
                          LPCTSTR pstr)
```

Parameters:

unsigned int x;	●X coordinates;
unsigned int y;	●Y coordinates; Note:1 dot = 0.125 mm.
unsigned int w;	●Maximum printing width, in dots;
unsigned int v;	●Maximum printing height, in dots;
unsigned int s;	●Error correction level, range:0~8.
unsigned int c;	●Compression mode, range:0 or 1.
unsigned int px;	●Module width, range:2~9 dots.
unsigned int py;	●Module height, range:4~99 dots.
unsigned int r;	●Max row count.
unsigned int l;	●Max column count.
unsigned int t;	●Truncation flag, '0' means normal and '1' means truncated.
unsigned int o;	●print direction positioning, '0' means 0°, '1' means 90°, '2' means 180°, '3' means 270°
LPCTSTR pstr;	●A character string (length is 1 to 100), using "DATA", Cn and Vn Parameters.

"DATA": A data fixed string, it must begin and end with "". Example: "POSTEK Printer".

Cn: A counter value. Refer to C command.

Vn: A variable string. Refer to V command.

Example: "data1"CnVn "data2".

Return:

0 -> OK.

Reference Error.txt file.

Example:

```
unsigned int x, y, w, v, s, c, px, py, r, l, t, o;  
LPCTSTR pstr = "POSTEKINFO";  
x=10;y=10;w=400;v=300;s=0;c=0;px=3;py=7;r=10;l=2;t=0;o=0;  
PTK_DrawBar2D_Pdf417 (x, y, w, v, s, c, px, py, r, l, t, o, pstr);
```

PTK_DrawBar2D_HANXIN

Description:

This function is used to print a HANXIN 2D barcode.

Syntax:

```
PTK_DrawBar2D_HANXIN( unsigned int x, unsigned int y,  
                      unsigned int w, unsigned int v,  
                      unsigned int o, unsigned int r,  
                      unsigned int m, unsigned int g,  
                      unsigned int s, LPCTSTR pstr );
```

Parameters:

x;	●X coordinates;
y;	●Y coordinates; Note:1 dot = 0.125 mm.
w;	●Maximum printing width, in dots;
v;	●Maximum printing height, in dots;
o;	●Print direction positioning, '0'means 0°, '1'means 90°, '2' means 180°, '3'means 270°
r;	●Set the multiplier values, in dots, value range: (1 - 9).
m;	●Select the coding mode of HANXIN, value range:(0 - 6).
g;	●Select the correcting ranks of HANXIN, value range: (0 - 3)
s;	●Select the masking pattern of HANXIN, value range: (0 - 3).
pstr:	●The contents of string.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_DrawBar2D_ HANXIN (50, 30, 0, 0, 0, 5, 1, 3, 2, "POSTEK");
```

PTK_PcxGraphicsList

Description:

This function causes the printer to print the list of graphics that stored to RAM or flash memory from host.

Syntax:

```
int PTK_PcxGraphicsList (void );
```

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_PcxGraphicsList ( );
```

PTK_PcxGraphicsDel

Description:

This function causes the printer to delete graphics currently stored in RAM or flash memory.

Syntax:

```
int PTK_PcxGraphicsDel (LPTSTR pid);
```

Parameters:

pid: Graphics name with a maximum of 16 characters.

if pid = "*", all graphics will be deleted from RAM or flash memory.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_PcxGraphicsDel ("PCX2");
```

PTK_PcxGraphicsDownload

Description:

This function causes the printer to store a PCK graphics to printer.

Syntax:

```
int PTK_PcxGraphicsDownload (char* pcxname, char* pcxpath);
```

Parameters:

pcxname: User-defined graphics name with a maximum of 16 characters. graphics can only be printed by using this name in PTK_DrawPcxGraphics() after the graphics being stored to the printer

pcxpath: The path of the PCX graphics in memory.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_PcxGraphicsDownload ("PCXA", "c:\\test1111.pcx");
```

PTK_DrawPcxGraphics**Description:**

This function is used to print the designated graphics.

Note: The graphics must store in the printer by using PTK_PcxGraphicsDownload before it to be printed.

Syntax:

```
int PTK_DrawPcxGraphics (unsigned int px, unsigned int py, LPTSTR gname);
```

Parameters:

px: X coordinate in dots.

py: Y coordinate in dots.

game: Graphics name with a maximum of 16 characters, it must be user-defined name in the PTK_PcxGraphicsDownload().

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_DrawPcxGraphics (100, 50, "PCX1");
```

PTK_PrintPCX**Description:**

This function is used to print a PCK graphics.

Using this function is equal to use PTK_PcxGraphicsDownload() and PTK_DrawPcxGraphics() at the same

time.

Syntax:

```
int PTK_PrintPCX (unsigned int px, unsigned int py, char* filename);
```

Parameters:

px:X coordinate in dots.

py:Y coordinate in dots.

filename:PCX Graphics name with a maximum of 16 characters, include the path of the file.

Format: "XXXXXXXXX.XXX" or "X:\\XXX\\XXX.PCX".

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_PrintPCX(10,100, "c:\\phone.pcx");
```

PTK_BmpGraphicsDownload

Description:

This function is used to convert a graphic in BMP format to PCX format, and then download the PCX graphic into the printer memory.

Syntax:

```
int PTK_BmpGraphicsDownload(LPTSTR pcxname,LPTSTR pcxpath, int iDire);
```

Parameters:

pcxname: User-defined graphics name with a maximum of 16 characters. Graphics can only be printed by using this name in PTK_DrawPcxGraphics () after the graphics being stored to the printer.

pcxpath: The path of the BMP graphics in memory.

iDire: print direction positioning, '0' means 0°, '1' means 90°, '2' means 180°, '3' means 270°, others means 0°.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_BmpGraphicsDownload ("PCXA", "c:\\test1111.bmp", 0);
```

PTK_BinGraphicsList

Description:

This function is used to print the Bin and PCX graphics name list stored in the RAM or flash memory.

Syntax:

```
int PTK_BinGraphicsList (void );
```

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_BinGraphicsList ();
```

PTK_BinGraphicsDel

Description:

This function is used to delete one or all of Bin and PCX graphics stored in the printer.

Syntax:

```
int PTK_BinGraphicsDel (LPTSTR pid);
```

Parameters:

pid: Form name with a maximum of 16 characters.

If pid = "*", all forms will be deleted from RAM or flash memory.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_BinGraphics Del ("Bin2");
```

PTK_BinGraphicsDownload

Description:

This function is used to store a Bin graphic to the printer.

Syntax:

```
int PTK_BinGraphicsDownload (char* name,unsigned int pbyte,unsigned int pH,UCHAR * Gdata);
```

Parameters:

name: User-defined graphics name with a maximum of 16 characters. Graphics can only be printed by using this name in PTK_DrawPcxGraphics () after the graphics being stored to the printer.

pbyte: Bytes for one line data. If 8 cannot divide the bits of one line data, then the bytes equal to the result add 1. Example: the bytes of one line data is 2(14 bits data).

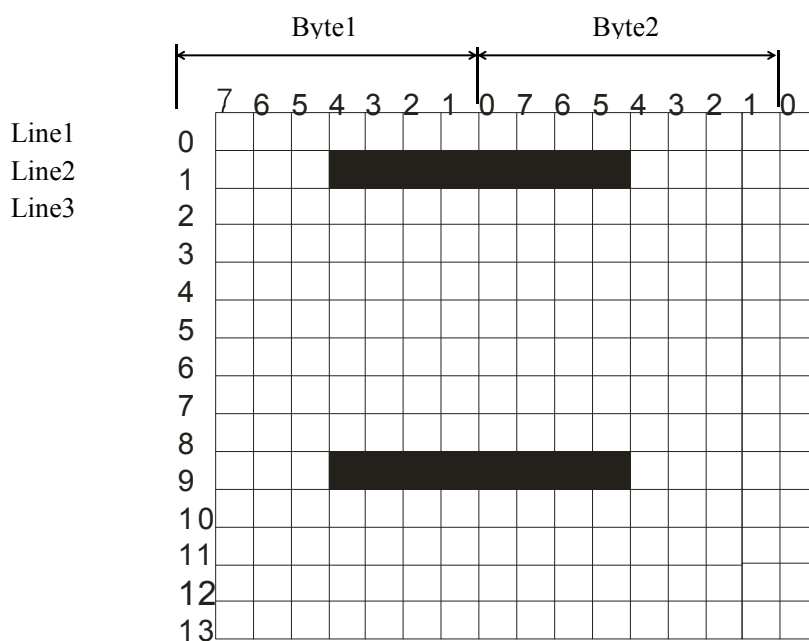
pH: Height of graphic, in dots.

Gdata([...raster data...]): Binary graphic data; data size = pbyte * pH (Bytes).

Binary data transmission sequence is left to right, up to down, example:

data transmission sequence: Byte1(0xff) of Line1, Byte2(0xff) of line1, Byte1(0xe0) of Line2, Byte2(0x1f) of Line2, Byte1(0xff) of Line3, Byte2(0xff) of Line3,...

The part of the broken line is non-graphic area, and then the corresponding bit is 0.



Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
char buf[] = {0xff,0xff,0xe0,0x1f,0xff,0xff...};
PTK_BinGraphicsDownload ("BinA", 3, 24, buf);
```

PTK_RecallBinGraphics

Description:

This function is used to print a stored Bin graphic

Syntax:

```
int PTK_RecallBinGraphics (unsigned int px, unsigned int py, char* name);
```

Parameters:

px: X coordinate in dots.

py: Y coordinate in dots.

name: User-defined graphics name with a maximum of 16 characters. Graphics can only be printed by using this name in PTK_DrawPcxGraphics () after the graphics being stored to the printer.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_RecallBinGraphics(10,100, "BinA");
```

PTK_DrawBinGraphics

Description:

This function causes the printer to print binary graphics.

Binary graphics is non-compressed graphics data. Each bit represents a dot, a value of "0" prints a dot; a value of "1" does not print a dot.

Syntax:

```
int PTK_DrawBinGraphics ( unsigned int  px, unsigned int  py,  
                          unsigned int  pbyte, unsigned int  pH,  
                          UCHAR* Gdata );
```

Parameters:

px: X coordinate in dots.

py: Y coordinate in dots.

pbyte: Bytes for one line data. If 8 cannot divide the bits of one line data, then the bytes equal to the result add 1. Example: the bytes of one line data is 2(14 bits data),

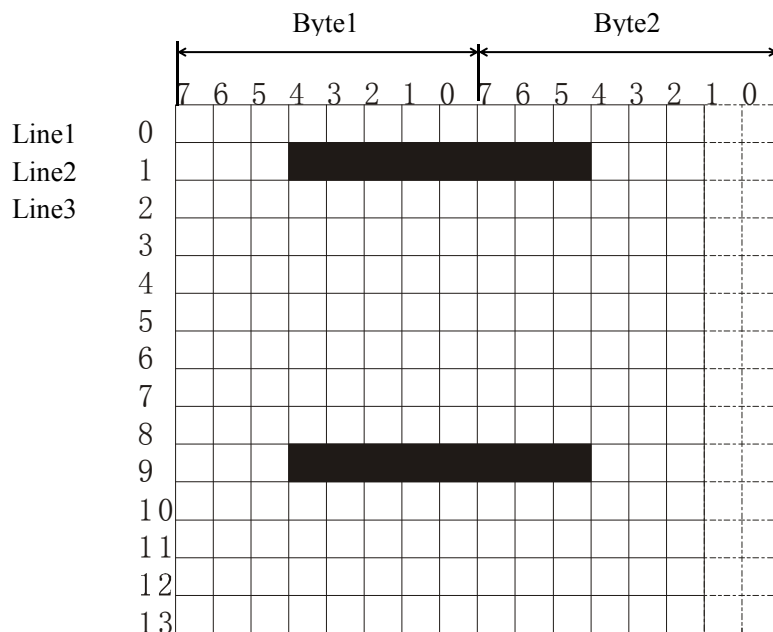
Ph: Height of graphic, in dots.

Gdata ([...raster data...]): Binary graphic data; data size = pbyte * pH (Bytes).

Binary data transmission sequence is left to right, up to down, example:

data transmission sequence: Byte1(0xff) of Line1, Byte2(0xff) of line1, Byte1(0xe0) of Line2, Byte2(0x1f) of Line2, Byte1(0xff) of Line3, Byte2(0xff) of Line3,...

The part of the broken line is non-graphic area, and then the corresponding bit is 0.



Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
char buf[] = {0xff, 0xff, 0xe0, 0x1f, 0xff, 0xff...};
PTK_DrawBinGraphics (20, 30, 4, 14, buf);
```

PTK_DrawRectangle

Description:

This function is used to draw rectangle.

Syntax:

```
int PTK_DrawRectangle (unsigned int px, unsigned int py,
                       unsigned int thickness, unsigned int pEx,
                       unsigned int pEy);
```

Parameters:

- px: Horizontal start position (X) in dots.
- py: Vertical start position (Y) in dots.
- thickness: Line thickness in dots.
- pEx: Horizontal end position (X) in dots.
- pEy: Vertical end position (Y) in dots.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_DrawRectangle (50, 120, 5, 250, 150);
```

PTK_DrawLineXor

Description:

This function causes the printer to draw straight-line (If two straight-lines intersects, use an exclusive or operation).

Syntax:

```
int PTK_DrawLineXor (unsigned int px, unsigned int py,  
                    unsigned int pbyte, unsigned int pH);
```

Parameters:

px: X coordinate in dots.

py: Y coordinate in dots.

pbyte: Horizontal length in dots.

pH: Vertical length in dots.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_DrawLineXor (100, 20, 5, 110);
```

PTK_DrawLineOr

Description:

This function causes the printer to draw straight-line (If two straight-lines intersects, use Or operation).

Syntax:

```
int PTK_DrawLineXor (unsigned int px, unsigned int py,  
                    unsigned int pbyte, unsigned int pH);
```

Parameters:

px:X coordinate in dots.

py:Y coordinate in dots.

pbyte:Horizontal length in dots.

pH:Vertical length in dots.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_DrawLineOr (100, 20, 5, 110);
```

PTK_DrawDiagonal

Description:

This function is to draw diagonal.

Syntax:

```
int PTK_DrawDiagonal (unsigned int  px, unsigned int  py,  
                      unsigned int  thickness, unsigned int  pEx,  
                      unsigned int  pEy);
```

Parameters:

px: X coordinate in dots.

py: Y coordinate in dots.

thickness: set Thickness in dots.

pEx: End X coordinate in dots.

pEy: End Y coordinate in dots.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_DrawDiagonal (50, 30, 10, 100, 80);
```

PTK_DrawWhiteLine

Description:

This function is used to draw a whit line.

Syntax:

```
int PTK_DrawWhiteLine (unsigned int  px, unsigned int  py,  
                      unsigned int  plength, unsigned int  pH);
```

Parameters:

px: X coordinate in dots.
py: Y coordinate in dots.
plength: Horizontal length in dots.
pH: Vertical length in dots.

Returns:

0-> OK;
Other returns: Please make reference to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_DrawWhiteLine (100, 20, 5, 110);
```

PTK_SoftFontList**Description:**

This function will cause the printer to print a list of all soft fonts that are stored in RAM or FLASH memory.

Syntax:

```
int PTK_SoftFontList (void);
```

Parameters: None

Returns:

0-> OK;
Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_SoftFontList ();
```

PTK_SoftFontDel**Description:**

This function causes the printer to remove one or all, soft fonts stored in RAM and/or Flash memory.

Syntax:

```
int PTK_SoftFontDel (char pid);
```

Parameters:

pid: Soft font ID, A—Z,*
If pid = '*',all fonts will be deleted from RAM or flash memory.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_SoftFontDel ('A');
```

PTK_FormList

Description:

This function causes the printer to print a list of forms that have been stored.

Syntax:

```
int PTK_FormList (void );
```

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_FormList ( );
```

PTK_FormDel

Description:

This function causes the printer to delete forms currently stored.

Syntax:

```
int PTK_FormDel (LPTSTR pid);
```

Parameters:

pid:Form name with a maximum of 16 characters.

If pid = "*", all forms will be deleted from RAM or flash memory.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_FormDel ("FORMNAME");
```

PTK_FormDownload

Description:

This function is used to store a form to the printer, it must be used together with the PTK_FormEnd.

If this function is used after EnableFLASH(), the form will be save to FLASH memory.

If this function is used under default state or after DisableFLASH(), the form will be saved to flash memory, otherwise it is saved to RAM.

Syntax:

```
int PTK_FormDownload (LPTSTR pid);
```

Parameters:

pid: User-defined form name with a maximum of 16 characters. Use this form prior to execute this function after storing the form to the printer

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_FormDownload ("FORMNAME");
```

PTK_FormEnd

Description:

This function is used to end a form store, it must be used together with the PTK_FormDownload.

Syntax:

```
int PTK_FormEnd (void );
```

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_FormDownload("Form1");
```

```
...
```

```
PTK_FormEnd ();
```

PTK_ExecForm

Description:

This function is used to execute the designated form.

Syntax:

```
int PTK_ExecForm (LPTSTR pid);
```

Parameters:

pid: Form name with a maximum of 16 characters.

Returns:

0-> OK;

Other returns: Please make reference to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_ExecForm ("FORM1");
```

PTK_DefineCounter

Description:

This function is used to define a counter variable.

Syntax:

```
int PTK_DefineCounter ( unsigned int id, unsigned int maxNum,  
                        char ptext, LPTSTR pstr, LPTSTR pMsg );
```

Parameters:

id: Counter ID, acceptable values are from 0 to 9.

maxNum: Maximum digit number, acceptable values are from 1 to 40.

ptext: Justification code. L for left justification, R for right justification, C for centralization and N for no justification.

Pstr: Counter change rule : "+" or "-" sign followed by a single digit of 1 – 9, then followed by a change symbol (i.e. D - decimal base, B - binary system, O - octonary number system, H - hexadecimal system, X - user defined pattern, to a maximum of 64 characters.)

Step values:

" +1 " = Increases each time by 1, according to Decimal base computation. Example: 1234, 1235, 1236...

" +3D " = Increases each time by 3, according to Decimal base computation. Example: 1234, 1235, 1236...

" -1B " = Decreases each time by 1, according to Binary computation. Example: 1111, 1110,

1101...

“-4O= Decreases each time by 4, according to Octonary number system computation. Example: 1234, 1230, 1224...

“-6H”= Decreases each time by 1, according to hexadecimal base computation. Example: 1234, 122E, 1228...

“+3X”= Increase each time by 3, according to a user-defined pattern. Example: In user-defined rule: TE2DOKLU046MNY37, the starting value is “T062”, followed by T062, T06K, T060...

pMsg: A text string that will be sent to LCD or KDU.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_DefineCounter (0, 6, 'N', "+1", "\nEnter\ Code :");
```

PTK_DefineVariable

Description:

This function is used to define variable in the FORM.

Syntax:

```
int PTK_DefineVariable (unsigned int pid, unsigned int pmax,  
                        char porder, LPTSTR pmsg);
```

Parameters:

pid: Variable ID number, value range: 00—99;

pmax: Maximum number of characters, value range: 1—99.

If you use KDU, the length should limited to under 16 characters.

porder: Field Justification; L-left justification, R- right justification, C-center, N-no justification.

pmsg: Remind contents; Will be sent to KDU or displayed on LCD display of the printer.

Returns:

0-> OK;

Other returns: Please make reference to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_DefineVariable (0,16,L, "\nEnter Title:");
```

PTK_Download

Description:

This function is used to download variable or counter variable to the printer.

Please refer to the command “?” of the PPL I.

Syntax:

```
int PTK_Download(void);
```

Parameters: None

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_Download( );
```

PTK_DownloadInitVar

Description:

This function is used to initialize variable or counter variable.

This function must use before the Closeport().

Syntax:

```
int PTK_DownloadInitVar(LPTSTR pstr);
```

Parameters: None

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_DownloadInitVar("123456");
```

PTK_PrintLabelAuto

Description:

This function causes the printer to execute to printing job automatically. It suggests that use this function when variables or counter values exist. The printer prints the form, as soon as all variable data have been input.

Notes: Only use in FORM.

Syntax:

```
int PTK_PrintLabelAuto (unsigned int  number,  unsigned int  cpnumber);
```

Parameters:

number: Number of label, 1—65535.

cpnumber: Number of copies for per label, 1—65535.

If the cpnumber haven't been set, it will default 1.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_PrintLabelAuto (2, 3);
```

PTK_SendFile

Description:

This function is used to send a command file to the printer.

Syntax:

```
int PTK_SendFile(LPTSTR FilePath);
```

Parameters:

FilePath: The path of the command file in memory.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_SendFile("cmdfile.txt");
```

PTK_GetUSBID

Description:

This function is used to get the USB ID of the printer.

Syntax:

```
int PTK_GetUSBID (LPTSTR USBDeviceSerial);
```

Parameters:

USBDeviceSerial: Variable defined to save the USB ID.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Note: USB ID can be only got from USB port.

Example:

```
TCHAR USBIDString [10] = {_T('0')};  
PTK_GetUSBID (USBIDString);
```

PTK_DisableBackFeed

Description:

This function is used to disable Tear-off mode for the printer.

Syntax:

```
int PTK_DisableBackFeed(void);
```

Parameters: None

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_DisableBackFeed ( );
```

PTK_EnableBackFeed

Description:

This function is used to enable Tear-off mode for the printer.

Syntax:

```
int PTK_EnableBackFeed (unsigned int distance);
```

Parameters:

distance: Back feed distance, unit in dots.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_EnableBackFeed (140);
```

PTK_PrintConfiguration

Description:

This function is used to print the current printer configuration.

Syntax:

```
int PTK_PrintConfiguration ( );
```

Parameters: None.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_PrintConfiguration ( );
```

PTK_SetPrinterState

Description:

This function is used to set the printer's working state.

Syntax:

```
int PTK_SetPrinterState (char state);
```

Parameters:

State:

D: Enable direct thermal printing (without ribbon).

P: Enable continuous printout.(default)

L: After printing a label the printer will stop, requiring user input to print the next label. Input determined by: 1. By pressing the “feed” button for each label to be printed. 2. Will continue automatically after previously printed label is removed (with peeler kit installed)

C: Enable Cutting mode. (Only with cutter kit installed)

N: Enable Peeler mode. (Only with peeler kit installed)

Notes:

1. The cutter and dispenser cannot be enabled at the same time.
2. Once the options are incorrectly selected, the READY light in front panel will flicker. Please refer to

the trouble-shooting section to correct the errors.

Returns:

0-> OK;

Other returns: Please make reference to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_SetPrinterState ('D');
```

PTK_DisableErrorReport

Description:

This function is used to disable the error report function from taking effect.

Syntax:

```
int PTK_DisableErrorReport(void);
```

Parameters: None

Returns:

0-> OK;

Other returns: Please make reference to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_DisableErrorReport( );
```

PTK_EnableErrorReport

Description:

This function is used to enable the error report function.

The printer sends its feedback through the RS232 port.

If an error occurs, the printer will send a NACK (0x15), followed by the error number, to the host. If no errors occur, the printer will echo ACK (0x6) after each P command.

Error Code	Error/Status Description
0x00	No Error
0x01	Object Exceeded Label Border
0x02	Bar Code Data Length Error
0x03	Insufficient Memory to Store Data
0x04	Memory Configuration Error

0x05	RS-232 Interface Error
0x06	Paper or Ribbon Empty
0x07	Duplicate Name: Form, Graphic or Soft Font
0x08	Name Not Found: Form, Graphic or Soft Font
0x09	Not in Data Entry Mode
0x0a	Print Head Up (Open)
0x0b	Pause Mode or Paused in Peel mode
0x0c	Does not fit in area specified
0x0d	Data length to long
0x0c	PDF-417 coded data to large to fit in bar code
0x0d	
0x0e	

PTK_EnableFLASH

Description:

This function is used to enable Flash memory.

The data sent to the printer will be stored to the flash memory when use this function.

Syntax:

```
int PTK_EnableFLASH ( void);
```

Parameters: None

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_EnableFLASH ();
```

PTK_DisableFLASH

This function is used to disable Flash memory.

The data sent to the printer will be stored to the SDRAM when use this function.

Syntax:

```
int PTK_DisableFLASH (void);
```

Parameters: None

Returns:

0-> OK;

Other returns: Please make reference to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_DisableFLASH ( );
```

PTK_FeedMedia

Description:

This function is used to feed one label.

Syntax:

```
int PTK_FeedMedia (void);
```

Parameters: None

Returns:

0-> OK;

Other returns: Please make reference to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_FeedMedia ( );
```

PTK_MediaDetect

Description:

This function is used to do media calibration.

Syntax:

```
int PTK_MediaDetect (void);
```

Parameters: None

Returns:

0-> OK;

Other returns: Please make reference to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_MediaDetect ( );
```

PTK_CutPage

Description:

This function is used to set cutter's working circle (namely the cutter will start to cut labels once after how many labels have been printed).

After this function is executed, the printer will enter settings mode, the change shall take effect after cycling the printer power.

Syntax:

```
int PTK_CutPage (UINT page);
```

Parameters:

page: pages, value range: 1-999, the default value is 1.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_CutPage(1);
```

PTK_CutPageEx

Description:

This function is used to set the cutter frequency immediately (namely the cutter will start to cut labels once after how many labels have been printed). The setting will take effect without cycling the printer power.

Syntax:

```
int PTK_CutPageEx (unsigned int page);
```

Parameters:

page: pages, value range: 1-999, the default value is 1.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_CutPage(1);
```

PTK_FeedBack

Description:

This function is used to require an error report from printer immediately.

Use this command to get printer error and status reports immediately. The printer will report 4 bytes back to the host in the following format.

0xXX XX 0x0d 0x0a: Error/Status code <CR><LF>

Error/Status code	Error/Status Description
00	No Error
01	Syntax Error
82	Ribbon Error
83	Media Error
86	Cutter Error
87	Print Head Up (Open)
88	Pause
99	Other Error

Syntax:

```
int PTK_FeedBack (void);
```

Parameters: None

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_FeedBack ( );
```

PTK_Reset

Description:

This function is used to reset the printer.

This function is equal to the power off, then power on, thus reinitializing the printer.

Notes:

The reset function is not available during the download of PCX graphics, soft fonts or while the printer is in dump mode.

The reset function cannot be used within a stored form.

The reset function can be sent to the printer during all other printing operations.

The printer will ignore all commands sent while the reset command is executing, up to 2 seconds.

Syntax:

```
int PTK_Reset(void);
```

Parameters: None

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_Reset();
```

PTK_ErrorReport

Description:

This function is used to send ^ee command to the printer and get an error code via the serial port.

Use this command to get printer error and status reports immediately. The printer will report 4 bytes back to the host in the following format.

0xXX XX 0x0d 0x0a : Error/Status code <CR><LF>

Chart 1

Error/Status code	Error/Status Description	PTK_ErrorReport()Return Value	ErrorCode
00	No Error	0	“00”
01	Syntax Error	1	“01”
82	Ribbon Error	82	“82”
83	Media Error	83	“83”
86	Cutter Error	86	“86”
87	Print Head Up (Open)	87	“87”
88	Pause	88	“88”
99	Other Error	99	“99”

Syntax:

```
int PTK_ErrorReport (int wPort, int rPort, DWORD BaudRate, BOOL HandShake, int TimeOut);
```

Parameters:

wPort: The port used to send datas; (This Parameters is mainly reserved for programming extension, default it to zero, no function.)

0: Print to filw PBuffi.txt (Create file under the program's execution directory); (Pls do not use this port!)

- 1: Open LPT1;
- 2: Open LPT2;
- 3: Open LPT3;
- 4: Open COM1;
- 5: Open COM2;
- 6: Open COM3.

rPort: The port used to receive current error status code; (This Parameters can support COM1 to COM255)

- 1: Open COM1 as the receive port;
- 2: Open COM2 as the receive port;
- 3: Open COM3 as the receive port;

BaudRate: Set serial port's baud rate, options:
9600, 19200, 38400, 57600;

HandShake: Use Handshaking or not;
TRUE: HandShaking enabled;
FALSE: HandShaking disabled.

TimeOut: Serial receiving timeout, unit: 100ms

Returns:

>=0, please refer to the Error/Status code value in chart 1.

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

1. Read current error status code from COM1 by setting the printing port in C168 200DPI driver.
char buff[5] = {0};
PTK_ErrorReportEx (1, 1,38400, FALSE,20);

OpenPort("POSTEK C168/200s ");
PTK_ClearBuffer();
.....

.....
ClosePort();

PTK_ErrorReportUSB

Description:

This function is used to send ^ee command to the printer and get an error code via the USB port.

This function must used after the PTK_ClearBuffer().

Use this command to get printer error and status reports immediately. The printer will report 4 bytes back to the host in the following format.

0xXX XX 0x0d 0x0a : Error/Status code <CR><LF>

Chart 1

Error/Status code	Error/Status Description	PTK_ErrorReport()Return Value	ErrorCode
00	No Error	0	“00”
01	Syntax Error	1	“01”
82	Ribbon Error	82	“82”
83	Media Error	83	“83”
86	Cutter Error	86	“86”
87	Print Head Up (Open)	87	“87”
88	Pause	88	“88”
99	Other Error	99	“99”

Syntax:

```
int PTK_ErrorReportUSB();
```

Returns:

>=0, please refer to the Error/Status code value in chart 1.

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

1. Read current error status code through USB port:

```
OpenUSBPort(255);
....
PTK_ClearBuffer();
Int nerrorStatusCode;
nerrorStatusCode = PTK_ErrorReportUSB() ;

if (nerrorStatusCode ==0)
{
.....
print
.....
}
ClosePort();
```

PTK_RWRFIDLabel

Description:

This function is used to Read/Write RFID tag.

Syntax:

```
int PTK_RWRFIDLabel (int nRWMode, int nWForm,  
                     int nStartBlock, int nWDataNum,  
                     int nWArea,LPTSTR pstr)
```

Parameters:

nRWMode: RFID operation. 0-Read; 1-Write;
nWForm: RFID data format. 0-HEX; 1-ASCII;
nStartBlock: Starting block number;
nWDataNum: Byte size of RFID data;
nWArea: Write-in Block. 0-Reserved; 1-EPC; 2-TID; 3-USER;
pstr: A string. (Format determined by p2.)

Returns:

0-> OK;
Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example 1:

```
int return = PTK_RWRFIDLabel(1,0,2,6,1, "313233343536");
```

Result

Read EPC Block (Starting block number=2, size= 3word) 313233343536

Example 2:

```
int return = PTK_RWRFIDLabel(1,1,0,6,3,"POSTEK");
```

Result:

Read USER Block (Starting block number=0, size= 3word) 504F5354454B

PTK_SetRFLabelPWAndLockRFLabel

Description:

This function is used to lock and set password for RFID tag.

Syntax:

```
int PTK_SetRFLabelPWAndLockRFLabel(int nOperationMode, int OperationnArea,LPTSTR pstr)
```

Parameters:

nOperationMode: Action; Options: 0-Unlock; 1-Lock; 2-Unlock Permanently; 3-Lock Permanently; 4- Set Password;
OperationnArea: Block; Options: 0- Kill Password; 1- Access Password; 2-EPC; 3-TID; 4-USER;
pstr: A constant string consists of 8 HEX characters.

Returns:

0-> OK;
Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example 1:

```
int return = PTK_SetRFLabelPWAndLockRFLabel(1,1, "73BE115B");
```

Result:

Set the password for Access Password Block to "73BE115B"
Once password is set, it will be not allowed to do Write/ Lock/ Unlock operations to the certain Block unless you know the password.

Example 2:

```
int return = PTK_SetRFLabelPWAndLockRFLabel(4,0,"5462EF21");
```

Result;

Kill Password is set to "5462EF21"

PTK_SetRFIDLabelRetryCount*Description:**

This function is used to set RFID retries.

Syntax:

```
int PTK_SetRFIDLabelRetryCount(int nRetryCount)
```

Parameters:

nRetryCount: Retry count, Range: 0-9.

Returns:

0-> OK;
Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
int return = PTK_SetRFIDLabelRetryCount(7); Retry count = 7.
```

PTK_SetRFID

Description:

This function is used to set RFID.

Syntax:

```
int PTK_SetRFID(int nReservationParameters, int nReadWriteLocation,  
               int ReadWriteArea, int nMaxErrNum, int nErrProcessingMethod)
```

Parameters:

nReservationParameters: Reserved, Default value: 0;

nReadWriteLocation: Distance between top of label and the optimal RFID chip writing position;
Range: 0-999, Default value: 0. Unit: mm.

ReadWriteArea: Reserved, Default value: 0;

nMaxErrNum: Maximum number of retries, if exceed, the printer will stop and report a RFID error;
Range: 0-9, Default: value: 2;

nErrProcessingMethod: Reserved, Default value: 0.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
int return = PTK_SetRFID(0, 0, 0, 3, 0);
```

PTK_SetFontGap

Description;

This function is used to specify font gap for downloaded soft fonts.

Syntax:

```
int PTK_SetFontGap(int gap);
```

Parameters:

gap: Length of the spacing between neighboring characters, unit in dots, Range: -99~99.

Note: Fonts usually have initial spacing, value set after the “g” command does not overwrite the initial value, but is added to the initial value.

Final spacing = initial spacing + gap.

Remark: Followings are unit conversion relations between different resolutions:

203DPI: 1mm = 8 dot;

300DPI: 1mm = 11.8dot;

600DPI: 1mm = 23.6 dot;

Example: If users need to set the spacing between characters to 1 mm, then gap should be set to 8 in 203DPI, 12 in 300DPI, and 24 in 600DPI.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
int return = PTK_SetFontGap (10);
```

***PTK_SetBarCodeFontName**

Description:

This function is used to specify font name for barcode human readable text.

Syntax:

```
int PTK_SetBarCodeFontName (UCHAR Name,unsigned int FontW,unsigned int FontH);
```

Parameters:

Name: Specify the stored font ID, Range: A-Z

This Parameters should be the same with Fontname in PTKRenameDownloadFont

FontW: the font width, unit in dots.

FontH: the font height, unit in dots.

Note: This command is used to print downloaded TrueType fonts as barcode's stored font, the size of font will be automatically adjusted with the width of barcode.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
int return = PTK_SetBarCodeFontName ('A',32,32);
```

PTK_RenameDownloadFont

Description:

This function is used to specify an ID for a downloaded font, the ID can be used as Parameters “p4” of PTK_DrawTextEx().

Syntax:

```
int PTK_RenameDownloadFont (unsigned int StoreType,UCHAR Fontname,LPTSTR DownloadFontName);
```

Parameters:

StoreType: Where to find the font, options: “1” or “0”, meaning “Flash” or “SDRAM”.

Note: The downloaded fonts in SDRAM will be erased after printer turned off, while downloaded fonts in FLASH will be reserved.

Fontname: Font ID you want to use, from “A”~“Z”.

DownloadFontName: Name of the font stored, for example, “arial”.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
int return = PTK_RenameDownloadFont (1, 'A', “arial”);
```

PTK_SetCharSets

Description:

This function is used to set the character set for the label.

Syntax:

```
int PTK_SetCharSets (unsigned int BitValue,UCHAR CharSets,LPTSTR CountryCode);
```

Parameters:

BitValue: Bit value; 8 = 8 bit character, 7 = 7 bit character.

CharSets: Character set.

CountryCode: Country code of KDU.

8 Bit Character (p1 = 8)	Code Page	7 Bit Character (p1 = 7)	Character Set
0	English (437)	0	USASCII
1	Latin 1 (850)	1	British
2	Slavic (852)	2	German
3	Portugal (860)	3	French
4	Canadian/French (863)	4	Danish

5	Nordic (865)	5	Italian
		6	Spanish
		7	Swedish
		8	Swiss

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
Int return = PTK_SetCharSets (8,'N',"001");
```

PTK_ReadRFTagData

Description:

This function is used for the printer to read and get the EPC / TID data from the RFID tags via the designated serial port.

This function must be used before OpenPort().

Syntax:

```
int PTK_ReadRFTagData(unsigned int wPort, unsigned int rPort, DWORD BaudRate, BOOL HandShake,  
unsigned int TimeOut, unsigned int nDataBlock, unsigned int nRFPower, BOOL bFeed, LPTSTR strRFData);
```

Parameters:

wPort: The port through which the commands are sent to printer.

- 0: Print to file PBufFi.txt (Create a "PBufFi.txt" file under the program's execution directory); (For data checking and verifying purpose, please do not use this port while normal printing!)
- 1: LPT1;
- 2: LPT2;
- 3: LPT3;
- 4: COM1;
- 5: COM2;
- 6: COM3.

rPort: The port used for receiving data and error code;

- 1: Open COM1 as the receive port;
- 2: Open COM2 as the receive port;
- 3: Open COM3 as the receive port.

BaudRate: Set baud rate, value range: 9600, 19200, 38400, 57600;

HandShake: Enable/Disable HandShaking;

Parameters:

usbPort: The port used for sending data;

1: Open port USB001.

2: Open port USB002.

3: Open port USB003.

.....

255: When there is only one Postek printer connected to PC, the PC will open the communication port of this printer by default.

nDataBlock: Select data block; 0: TID , 1: EPC , 2: TID+EPC.

nRFPower: Set the power of reading; Range: 0 ~ 30dBm;

When set to 0, the power of reading is 23dBm by default;

bFeed: Feed one label after reading RFID label data.

TRUE: Feed one label,

FALSE: Read only.

strRFData: Store the obtained RFID label data.

Returns:

0-> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
LPTSTR strRFData;
```

```
PTK_ReadRFTagDataUSB (255, 1,0,TRUE, strRFData);
```

```
OpenPort (255);
```

```
....
```

```
ClosePort ();
```

PTK_ReadRFTagDataNet**Description:**

This function is used for the printer to read and get the EPC / TID data from the RFID tags via the designated TCP/IP port.

Syntax:

```
int PTK_ReadRFTagDataNet(LPTSTR IPAddress, unsigned int Port, unsigned int nFeedbackPort,  
                          unsigned int nRFPower, BOOL bFeed, LPTSTR strRFData);
```

Parameters:

IPAddress: IP address of the printer;
Port: Ethernet port of the printer;
nDataBlock: Select data block; 0: TID, 1: EPC, 2: TID+EPC.
nRFPower: Set the power of reading; Range: 0 ~ 30dBm;
When set to 0, the power of reading is 23dBm by default;
bFeed: Feed one label after reading RFID label data;
TRUE: Feed one label.
FALSE: Read only.
strRFData: Store the obtained RFID label data.

Returns:

0-> OK;
Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
LPTSTR strRFData;  
PTK_ReadRFDataNet ("199.9.10.230", 9100, 1, 0, TRUE, strRFData);
```

PTK_Connect**Description:**

This function is used to establish internet connection.

Syntax:

```
int PTK_Connect(LPTSTR IPAddress, unsigned int Port)
```

Parameters:

IPAddress: IP address of the printer.
Port: Network port of the printer.

Returns:

0 -> OK;
Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

```
PTK_Connect("199.9.10.2", 9100); //Establish internet connection//below example is for your reference only,  
please set parameters according to your actual label size and printing needs.  
PTK_SetLabelHeight(376, 20, 0, FALSE);  
PTK_SetLabelWidth(800);  
PTK_SetDarkness(10);
```

```
PTK_SetPrintSpeed(5);  
.....  
PTK_DrawBarcode(300,23,0,_T("1"),2,2,50,'B',_T("123456789"))); //Print a barcode  
PTK_DrawTextEx(80,168,0,3,1,1,'N',"Internal Soft Font",0); //Print a text  
.....  
PTK_PrintLabel(1,1);  
PTK_CloseConnect();
```

PTK_CloseConnect

Description:

This function can be used to close the Internet connection established by “PTK_Connect” function.

Syntax:

```
void PTK_CloseConnect()
```

Parameters:

None

Returns:

None

Example:

```
PTK_CloseConnect( );
```

PTK_ErrorReportNet

Description:

This function is used to get printer error report via TCP/IP port.

Error/Status code	Explanation	Return Value	Error Code
00	No Error	0	“00”
01	Syntax Error	1	“01”
4	Printing		
82	Ribbon Error	82	“82”
83	Media Error	83	“83”
86	Cutter Error	86	“86”
87	Print Head Up (Open)	87	“87”
88	Pause	88	“88”
99	Other Error	99	“99”

Syntax:

Int PTK_ErrorReportNet (LPTSTR PrintIPAddress, unsigned int PrinterNetPort);

Parameters:

PrinterAddress: Printer IP Address

PrinterNetPort: Printer Net Port

Returns:

Error/Status Code
00
01
82
83
86
87
88
99

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

int nPrintStatus

nPrintStatus-PTK_ErrorReportNet ("199.9.10.230",9100);

PTK_ReadPrintConifUSB

Description:

PTK_ReadPrintConifUSB is used to feeding back the printer information to the HOST via USB port.

Syntax:

intPTK_ReadPrintConifUSB(unsigned int usbPort, unsigned int nInfoType,LPTSTR strData)

Parameters:

usbPort: Set USB Port;

nInfoType: Set feedback contents, see below chart:

p2	Feedback Information Type
1	Printer model, Firmware Version, Firmware Part Number, Printing Resolution
2	Print Method(0 Direct Thermal/1 Thermal Transfer), Media sensor type (0 Transmissive/1 Down Reflective/2 Up Reflective), Ribbon Sensor(1 Effective/0 Void), Tear Mode(1 Enable/0 Disable), Cutter Mode(1 Enable/0 Disable), Printing Darkness(1~20)
3	IP Address, Subnet Mask, Gateway, Base Raw Port, Mac Address
4	Distance between printhead heating line to media sensor/cutter/tear off position, distance between the starting position of the current label to the next label.
5	RFID(1 Effective/0 Void), RFID Power, RFID Offset (unit in mm), current RFID tag's height (unit in dots)

strData: used for receiving printer feedback information.

Returns:

0 -> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Examples:

LPTSTR strData:

```
PTK_ReadPrintConifUSB(255,1, strData);
```

PTK_ReadPrintConifNet

Description:

PTK_ReadPrintConifNet is used to feeding back the printer information to the HOST via TCP/IP port.

Syntax:

```
IntPTK_ReadPrintConifNet(LPTSTR PrintIPAddress, unsigned int PrinterNetPort, unsigned int nInfoType, LPTSTR strData)
```

Parameters:

PrintIPAddress: Printer IP Address.

PrinterNetPort: Printer TCP/IP port.

nInfoType: Set feedback contents, see below chart:

p2	Feedback Information Type
1	Printer model, Firmware Version, Firmware Part Number, Printing Resolution
2	Print Method(0 Direct Thermal/1 Thermal Transfer), Media sensor type (0 Transmissive/1 Down Reflective/2 Up Reflective), Ribbon Sensor(1 Effective/0 Void), Tear Mode(1 Enable/0 Disable), Cutter Mode(1 Enable/0 Disable), Printing Darkness(1~20)
3	IP Address, Subnet Mask, Gateway, Base Raw Port, Mac Address
4	Distance between printhead heating line to media sensor/cutter/tear off position, distance between the starting position of the current label to the next label.
5	RFID(1 Effective/0 Void), RFID Power, RFID Offset (unit in mm), current RFID tag's height (unit in dots)

Note: If adopt this function, the HOST IP address would be set as the IP address of the currently connected HOST by default.

Returns:

0 -> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Examples:

LPTSTR strData;

PTK_ReadPrintConifNet("199.9.10.196",1,strData);

PTK_RFIDCalibrate

Description:

This function is used to calibrate the optimal encode/read position for the RFID chips. (Only available on printer firmware version V1.73 or higher).

Syntax:

Int PTK_RFIDCalibrate();

Parameters:

None

Returns:

0 -> OK;

Other returns: Please refer to chapter: CDFPSK.dll erroneous return value specification.

Example:

Support Parallel, USB and Serial Port	TCP/IP Port
OpenPort(255); PTK_RFIDCalibrate(); ClosePort();	PTK_Connect("199.9.10.2", 9100) PTK_RFIDCalibrate(); PTK_CloseConnect();

CDFPSK.dll Erroneous return value specification

- 1000 to -1011 : OpenPort execution error;
- 1012 to -1025 : Serial port read error;
- 1026 to -1028 : SetPCComPort serial port setting error;
- 1029 to -1030 : Write data error;

- 2000 : PTK_GetInfo execution error;
- 2001 : PTK_ClearBuffer execution error;
- 2002 : PTK_SetDarkness execution error;
- 2003 : PTK_SetPrintSpeed execution error;
- 2004 : PTK_SetPrintSpeed Parameters error;
- 2005 : PTK_SetLabelHeight execution error;
- 2006 : PTK_SetLabelWidth execution error;
- 2007 : PTK_SetDirection execution error;
- 2008 : PTK_SetDirection Parameters error;
- 2009 : PTK_SetCoordinateOrigin execution error;
- 2010 : PTK_PrintLabel execution error;
- 2011 : PTK_PrintLabel Parameters error;
- 2012 : PTK_PrintLabelAuto execution error;
- 2013 : PTK_PrintLabelAuto Parameters error;
- 2014 : PTK_DrawText execution error;
- 2015 : PTK_DrawText Parameters error;
- 2016 : PTK_DrawTextEx execution error;
- 2017 : PTK_DrawTextEx Parameters error;
- 2018 : PTK_DrawTextTrueTypeW build PrinterDC fail, deal with the error;
- 2019 : PTK_DrawTextTrueTypeW distribution save bitmap memory error;
- 2020 : distribution save contemporary file path memory error;
- 2021 : distribution save PCX HEAD file framework memory error;
- 2022 : build PCX file error;
- 2023 : distribution save PCX data memory error;
- 2024 : save PCX data erro;
- 2025 : PTK_DrawBarcode execution error;
- 2026 : PTK_DrawBarcode Parameters error;
- 2027 : PTK_DrawBarcodeEx execution error;
- 2028 : PTK_DrawBarcodeEx Parameters error;
- 2029 : PTK_DrawBar2D_DATAMATRIX execution error;
- 2030 : PTK_DrawBar2D_DATAMATRIX execution error;
- 2031 : PTK_DrawBar2D_QR execution error;
- 2032 : PTK_DrawBar2D_QR execution error;
- 2033 : PTK_DrawBar2D_QREx execution error;
- 2034 : PTK_DrawBar2D_MaxiCode execution error;
- 2035 : PTK_DrawBar2D_MaxiCode execution error;

- 2036 : PTK_DrawBar2D_Pdf417 execution error;
- 2037 : PTK_DrawBar2D_Pdf417 execution error;
- 2038 : PTK_DrawBar2D_HANXIN execution error;
- 2039 : PTK_DrawBar2D_HANXIN execution error;
- 2040 : PTK_PcxGraphicsList execution error;
- 2041 : PTK_PcxGraphicsDel distribution memory error;
- 2042 : PTK_PcxGraphicsDel Parameters error;
- 2043 : PTK_PcxGraphicsDel execution error;
- 2044 : PTK_PcxGraphicsDownload distribution memory error;
- 2045 : PTK_PcxGraphicsDownload distribution memory error;
- 2046 : PTK_PcxGraphicsDownload open file error;
- 2047 : PTK_PcxGraphicsDownload Parameters error;
- 2048 : PTK_PcxGraphicsDownload execution send file information error;
- 2049 : PTK_PcxGraphicsDownload execution send file content error;
- 2050 : PTK_DrawPcxGraphics distribution memory error;
- 2051 : PTK_DrawPcxGraphics Parameters error;
- 2052 : PTK_DrawPcxGraphics execution error;
- 2053 : PTK_PrintPCX execution error;
- 2054 : PTK_BmpGraphicsDownload distribution memory error;
- 2055 : PTK_BmpGraphicsDownload download BMP bit depth error;
- 2056 : PTK_BmpGraphicsDownload open file error;
- 2057 : PTK_BmpGraphicsDownload Parameters error;
- 2058 : PTK_BmpGraphicsDownload execution send file information error;
- 2059 : PTK_BmpGraphicsDownload execution send file content error;
- 2060 : PTK_BinGraphicsList execution error;
- 2061 : PTK_BinGraphicsDel distribution memory error;
- 2062 : PTK_BinGraphicsDel Parameters error;
- 2063 : PTK_BinGraphicsDelexecution error;
- 2064 : PTK_BinGraphicsDownload execution send binary system format information error;
- 2065 : PTK_BinGraphicsDownload execution send binary system graphics content error;
- 2066 : PTK_RecallBinGraphics distribution memory error;
- 2067 : PTK_RecallBinGraphics execution error;
- 2068 : PTK_RecallBinGraphics Parameters error;
- 2069 : PTK_DrawBinGraphics execution send binary system format information error;
- 2070 : PTK_DrawBinGraphics execution send binary system graphics content error;
- 2071 : PTK_DrawRectangle execution error;
- 2072 : PTK_DrawLineXor execution error;
- 2073 : PTK_DrawLineOr execution error;
- 2074 : PTK_DrawDiagonal execution error;
- 2075 : PTK_DrawWhiteLine execution error;
- 2076 : PTK_SoftFontList execution error;
- 2077 : PTK_SoftFontDel Parameters error;
- 2078 : PTK_SoftFontDel execution error;

- 2079 : PTK_FormList execution error;
- 2080 : PTK_FormDel distribution memory error;
- 2081 : PTK_FormDel Parameters error;
- 2082 : PTK_FormDel execution error;
- 2083 : PTK_FormDownload distribution memory error;
- 2084 : PTK_FormDownload Parameters error;
- 2085 : PTK_FormDownload execution error;
- 2086 : PTK_FormEnd execution error;
- 2087 : PTK_ExecForm distribution memory error;
- 2088 : PTK_ExecForm Parameters error;
- 2089 : PTK_ExecForm execution error;
- 2090 : PTK_DefineCounter distribution memory error;
- 2091 : PTK_DefineCounter distribution memory error;
- 2092 : PTK_DefineCounter execution error;
- 2093 : PTK_DefineCounter Parameters error;
- 2094 : PTK_DefineVariable execution error;
- 2095 : PTK_DefineVariable symbols content parameters error;
- 2096 : PTK_DefineVariable othersParameters error, please refer to function declaration;
- 2097 : PTK_Download execution error;
- 2098 : PTK_DownloadInitVar distribution memory error;
- 2099 : PTK_DownloadInitVar execution error;
- 2100 : PTK_SendFile distribution memory fail;
- 2101 : PTK_SendFile open file error;
- 2102 : PTK_SendFile execution write data error;
- 2103 : PTK_GetUSBID execution error;
- 2104 : PTK_DisableBackFeed execution error;
- 2105 : PTK_EnableBackFeed execution error;
- 2106 : PTK_PrintConfiguation execution error;
- 2107 : PTK_SetPrinterState execution error;
- 2108 : PTK_SetPrinterState Parameters error;
- 2109 : PTK_DisableErrorReport execution error;
- 2110 : PTK_EnableErrorReport execution error;
- 2111 : PTK_EnableFLASH execution error;
- 2112 : PTK_DisableFLASH execution error;
- 2113 : PTK_FeedMedia execution error;
- 2114 : PTK_MediaDetect execution error;
- 2115 : PTK_CutPage execution error;
- 2116 : PTK_CutPageEx execution error;
- 2117 : PTK_Reset execution error;
- 2118 : PTK_FeedBack execution error;
- 2119 : PTK_ErrorReport get feedback timeout;
- 2120 : PTK_ErrorReportUSB open USB port fail;
- 2121 : PTK_ErrorReportUSB get printer feedback fail;
- 2122 : PTK_StringDownload distribution memory fail;

- 2123 : PTK_StringDownload execution error;
- 2124 : PTK_RWRFIDLabel distribution memory fail;
- 2125 : PTK_RWRFIDLabel execution error;
- 2126 : PTK_SetRFLabelPWAndLockRFLabel distribution memory fail;
- 2127 : PTK_SetRFLabelPWAndLockRFLabel execution error;
- 2128 : PTK_SetRFIDLabelRetryCount execution error;
- 2129 : PTK_SetRFIDk execution error;
- 2130 : PTK_SetFontGap execution error;
- 2131 : PTK_SetBarCodeFontName execution error;
- 2132 : PTK_SetCharSets execution error;
- 2133 : PTK_RenameDownloadFont execution error;
- 2134 : PTK_DrawBar2D_DATAMATRIX distribution memory fail;
- 2135 : PTK_DrawBar2D_QR distribution memory fail;
- 2136 : PTK_DrawBar2D_QREx distribution memory fail;
- 2137 : PTK_DrawBar2D_MaxiCode distribution memory fail;
- 2138 : PTK_DrawBar2D_Pdf417 distribution memory fail;
- 2139 : PTK_DrawBar2D_HANXIN distribution memory fail;
- 2140 : PTK_BmpGraphicsDownloado pen file fail;
- 2141 : PTK_ErrorReport open port error;
- 2142 : PTK_BinGraphicsDownload distribution memory error;
- 2143 : PTK_Connect network connection error;
- 2144 : PTK_Connect Printer TCP/IP port parameters error;
- 2145 : PTK_Connect Printer IP parameter error;
- 2146 : PTK_Connect Printer connection fail;
- 2147 to -2148 : PTK_Connect space allocation fail;
- 2149 to -2151 : Write data via TCP/TP port error;
- 2152 : PTK_ReadRFTagDataNet Network printer connection error;
- 2153 : PTK_ReadRFTagDataNet Write data error;
- 2154 : PTK_ReadRFTagData Wrong port for sending Get RFID info command;
- 2155 : PTK_ReadRFTagData Sending Get RFID info command error;
- 2156 : PTK_ReadRFTagData Read RFID timeout;
- 2157 : PTK_ReadRFTagDataUSB Open port fail;
- 2158 : PTK_ReadRFTagDataUSB Sending Get RFID info command error;
- 2159 : PTK_ReadDateUsb Reading space allocation fail;
- 2160 : PTK_ReadDateUsb Reading data timeout;
- 2161 : PTK_SendFile Reading file data error;
- 2162 : PTK_ReadSerialNumberNet Network printer connection error;
- 2163 : PTK_ReadSerialNumberNet Write data error;
- 2164 : PTK_ErrorReportNet Network printer connection error;
- 2165 : PTK_ErrorReportNet Write data error;
- 2166 : PTK_RFIDCalibrate Write data error;
- 2169 : PTK_ReadPrintConifUSB Open port fail;
- 2170 : PTK_ReadPrintConifUSB Write data error;
- 2173 : PTK_ReadPrintConifNet Network printer connection error;

- 2174 : PTK_ReadPrintConifNet Write data error;
- 2177 : PTK_ErrorReportNet Failed to get printer status;

- 3000 to -3084 : Port not opened or has been closed

- 3072 : PTK_SendFile read data file error.