BIXOLON

API Reference Guide

Linux Label SDK

Ver. 2.13

http://www.bixolon.com

Table of Contents

Copyright **4**

1. Manual guide **5**
1-1 Supported Kernel & Platform 5
1-2 Supported Interfaces 5
1-3 Supported Printers 6

2. Property **8**
2-1 CharacterSet & International CharacterSet(int W) 8
2-2 State (int R) 10

3. Method **11**
3-1 ConnectToPrinter 11
3-2 DisconnectPrinter 13
3-3 InitializePrinter 14
3-4 FeedOneLabel 15
3-5 SetSpeed 16
3-6 SetDensity 17
3-7 SetOrientation 18
3-8 SetCutter 19
3-9 SetBackFeed 20
3-10 SetPaper 21
3-11 SetMargin 22
3-12 SetOffset 23
3-13 PrintDeviceFont 24
3-14 PrintDeviceFontW 26
3-15 PrintVectorFont 29
3-16 PrintVectorFontW 32
3-17 Prints 35
3-18 Print1DBarcode 36
3-19 PrintMaxiCode 39
3-20 PrintPDF417 40
3-21 PrintQRCode 43
3-22 PrintDataMatrix 45
3-23 PrintAztec 47
3-24 PrintCode49 49
3-25 PrintCODABLOCK 51
3-26 PrintMicroPDF 53
3-27 PrintGS1DataBar 55
3-28 PrintBlock 57
3-29 PrintCircle 59
3-30 PrinteBitmap 60

3-31 DirectIO 61
3-32 CalibrateMedia 62
3-33 ResetPrinter 63
3-34 ClearBuffer 64
3-35 SetRewinder 65
3-36 SetupRFID 66
3-37 SetRFIDPosition 67
3-38 SetEPCDataStructure 68
3-39 WriteRFID 69
3-40 SetRFIDPassword 70
3-41 LockRFID 71

Copyright

© BIXOLON Co., Ltd. All rights reserved.

This user manual and all property of the product are protected under copyright law.
It is strictly prohibited to copy, store, and transmit the whole or any part of the manual and any property of the product without the prior written approval of BIXOLON Co., Ltd.
The information contained herein is designed only for use with this BIXOLON product.
BIXOLON is not responsible for any direct or indirect damages, arising from or related to use of this information.

• The BIXOLON logo is the registered trademark of BIXOLON Co., Ltd.
• All other brand or product names are trademarks of their respective companies or organizations.

BIXOLON Co., Ltd. maintains ongoing efforts to enhance and upgrade the functions and quality of all our products. In the following, product specifications and/or user manual content may be changed without prior notice.

Caution

Some semiconductor devices are easily damaged by static electricity. You should turn the printer “OFF”, before you connect or remove the cables on the rear side, in order to guard the printer against the static electricity. If the printer is damaged by the static electricity, you

1. Manual guide

This SDK manual describes the contents of the library required to develop Linux OS application programs.

1-1 Supported Kernel & Platform

• Kernel
  - Kernel 2.6.32 or higher

• Platform
  - Linux 32bit / 64bit
  - Raspberry PI

• OS
  - openSUSE 11.3 32bit / 64bit
  - Red Hat Enterprise Linux 7.3 64bit
  - CentOS 6.6 32bit / 64bit
  - Ubuntu 10.04 LTS 32bit / 64bit

1-2 Supported Interfaces

• USB, Serial, Parallel, Bluetooth, Ethernet, WLAN

1-3 Supported Printers

Model	DPI	Max Printable Width	Speed Support
SLP-DX220	203 dpi	432 dots	3,4,5,6 ips
SLP-DX223	300 dpi	672 dots	2.5,3,4 ips
SLP-TX220	203 dpi	432 dots	3,4,5,6 ips
SLP-TX223	300 dpi	672 dots	2.5,3,4 ips
SLP-DX420	203 dpi	864 dots	3,4,5,6,7 ips
SLP-DX423	300 dpi	1248 dots	3,4,5 ips
SLP-TX420	203 dpi	864 dots	3,4,5,6,7 ips
SLP-TX423	300 dpi	1248 dots	3,4,5 ips
SLP-TX400	203 dpi	864 dots	3,4,5,6,7 ips
SLP-TX403	300 dpi	1248 dots	3,4,5 ips
SLP-DL410	203 dpi	864 dots	3,4,5 ips
SLP-DL413	300 dpi	1248 dots	3,4,5 ips
SPP-L3000	203 dpi	576 dots	1,2,3,4,5 ips
SPP-L310	203 dpi	576 dots	1,2,3,4,5 ips
SPP-L410	203 dpi	832 dots	1,2,3,4 ips
SRP-770III	203 dpi	832 dots	3,4,5 ips
SRP-E770III	203 dpi	832 dots	3,4,5 ips
XT5-40	203 dpi	832 dots	2 ~ 14 ips
XT5-43	300 dpi	1248 dots	2 ~ 10 ips
XT5-46	600 dpi	2496 dots	2,3,4,5 ips
XD3-40d	203 dpi	832 dots	3,4,5 ips
XD3-40t	203 dpi	832 dots	3,4,5 ips
XD5-40d	203 dpi	864 dots	3,4,5,6,7 ips
XD5-40t	203 dpi	864 dots	3,4,5,6 ips
XD5-43d	300 dpi	1248 dots	2,3,4,5 ips
XD5-43t	300 dpi	1248 dots	2,3,4 ips
XL5-40CT	203 dpi	864 dots	3,4,5,6 ips
XL5-43CT	300 dpi	1248 dots	2,3,4,5 ips
XM7-40	203 dpi	832 dots	1,2,3,4,5 ips
XM7-30	203 dpi	576 dots	1,2,3,4,5 ips
XM7-20	203 dpi	384 dots	1,2,3,4,5,6 ips
XD5-40tR	203 dpi	864 dots	3,4,5,6 ips
XD5-43tR	300 dpi	1248 dots	2,3,4 ips
XT5-40NR	203 dpi	832 dots	2 ~ 14 ips
XT5-43NR	300 dpi	1248 dots	2 ~ 10 ips
XT5-46NR	600 dpi	2496 dots	2,3,4 ips
SRP-S3000	203 dpi	576 dots	3,4,5,6,7,8 ips
XT3-40	203 dpi	864 dots	3,4,5,6,7,8 ips

XT3-43	300 dpi	1248 dots	2,3,4,5,6 ips
XM7-40R	203dpi	832 dots	1,2,3,4,5 ips
XF3-40	203dpi	864 dots	3,4,5,6 ips
XD5-40IIdt	203 dpi	864 dots	3 ~ 8ips
XD5-43IIdt	300 dpi	1248 dots	2 ~ 6ips
BD5-40d	203 dpi	864 dots	3,4,5,6,7ips
BD5-40t	203 dpi	864 dots	3,4,5,6ips
BD5-43d	300 dpi	1248 dots	2,3,4,5ips
BD5-43t	300 dpi	1248 dots	2,3,4ips
BT5-40	203 dpi	832 dots	2 ~ 14ips
BT5-43	300 dpi	1248 dots	2 ~ 10ips
BT5-46	600 dpi	2496 dots	2,3,4,5ips
BT3-40	203 dpi	832 dots	3 ~ 8ips
BT3-43	300 dpi	1248 dots	2 ~ 6ips

2. Property

The constants used by the library are declared in BxlLabelConst.h. The development environment is based on C.

2-1 CharacterSet & International CharacterSet(int W)

• This is the property for defining the printer's code page and International character Set and set to CS_CP437 and ICS_USA by default. The values can be set using SetCharacterSet().

CharacterSet settings may need to be verified in the following cases
1. When character strings other than the one you tried to print are printed
2. When a broken string is printed in the same form as hieroglyphic characters
3. When characters are printed in the form of '?' (question mark)

The following code pages can be used:

Constant	Value	Description
CS_CP437	0	U.S.A
CS_CP850	1	Latin 1
CS_CP852	2	Latin 2
CS_CP860	3	Portuguese
CS_CP863	4	Canadian French
CS_CP865	5	Nordic
CS_WPC1252	6	Latin I
CS_CP865_WCP1252	7	European Combined
CS_CP857	8	Turkish
CS_CP737	9	Greek
CS_WCP1250	10	Latin 2
CS_WCP1253	11	Greek
CS_WCP1254	12	Turkish
CS_CP855	13	Cyrillic
CS_CP862	14	Hebrew
CS_CP866	15	Cyrillic
CS_WCP1251	16	Cyrillic
CS_WCP1255	17	Hebrew
CS_CP928	18	Greek
CS_CP864	19	Arabic
CS_CP775	20	Baltic
CS_WCP1257	21	Baltic
CS_CP858	22	Latin 1 + Euro

The following International character Set can be used:

Constant	Value	Description
ICS_USA	0	USA code
ICS_FRANCE	1	FRANCE code
ICS_GERMANY	2	GERMANY code
ICS_UK	3	UK code
ICS_DENMARK_I	4	DENMARK1 code
ICS_SWEDEN	5	SWEDEN code
ICS_ITALY	6	ITALY code
ICS_SPAIN_I	7	SPAIN code
ICS_NORWAY	8	NORWAY code
ICS_DENMARK_II	9	DENMARK 2 code
ICS_JAPAN	10	JAPAN code
ICS_SPAIN_II	11	SPAIN 2 code
ICS_LATIN	12	LATIN code
ICS_KOREA	13	KOREA code
ICS_SLOVENIA	14	SLOVENIA code
ICS_CHINA	15	CHINA code

* Example

```
int ret;

ret = ConnectPrinter (“portinfo…”);

……

SetCharacterSet(CS_PC850, ICS_UK);

……
```

2-2 State (int R)

• This is the property that sets the printer status and calls the CheckPrinterStatus function to check the printer status and receive the data. The status value can be set in duplicate and each value can be checked using bitwise operation.

These are the printer status values.

Constant	Value	Description
STS_NORMAL	0	Printer ready
STS_RIBONEND	4	Ribbon end error
STS_GAPERROR	8	Unable to recognize gap(auto sensing failure)
STS_TPHOVERHEAT	16	TPH overheat
STS_CUTTERJAM	32	Cutter jammed
STS_COVEROPEN	64	Cover open
STS_PAPEREMPTY	128	No paper

* Example

```
int ret;

ret = ConnectToPrinter (“portinfo…”);
……

int state;

state = CheckPrinterStatus();

if ((state & STS_RIBONEND) == STS_RIBONEND)
    ……

if ((state & STS_GAPERROR) == STS_GAPERROR)
    ……
…….
```

3. Method

The functions provided by Linux SDK are declared in BxlLabelAPI.h.
The development environment is based on C.

3-1 ConnectToPrinter

• Set the connection for communication with the printer.

int ConnectToPrinter(const char *port)

[Parameters]

* const char *port
[in] Interface to be connected to the printer

Interface	Input Data	Example
USB	USB:	ConnectToPrinter("USB:")
Serial	serial:(baudrate) /dev/ttySX:(baudrate)	ConnectToPrinter("serial:115200") ConnectToPrinter("/dev/ttyS0:115200")
Parallel	parallel /dev/lpX	ConnectToPrinter("parallel") ConnectToPrinter("/dev/lp0")
Bluetooth	Device MAC address	ConnectToPrinter("7d:f0:7d:e4:e0:78")
Ethernet, Wifi	IP address, port no.	ConnectToPrinter("192.168.0.10:9100")

[Return Values]

Constant	Value	Description
SUCCESS	0	The operation is successful.
PORT_OPEN_ERROR	-99	The communication port cannot be opened.
NO_CONNECTED_PRINTER	-100	The printer is not connected.
NO_BIXOLON_PRINTER	-101	It is not a BIXOLON printer.

* Example

```
int ret;

// USB
ret = ConnectToPrinter("USB:");

// Serial
ret = ConnectToPrinter("/dev/ttyS0:115200");

// Parallel
ret = ConnectToPrinter("parallel");

// Bluetooth
ret = ConnectToPrinter("7d:f0:7d:e4:e0:78");

// Ethernet or WiFi
ret = ConnectToPrinter("192.168.0.10:9100");
.....
```

3-2 DisconnectPrinter

• Disconnect the printer.

int DisconnectPrinter();

[Parameters]

None

[Return Values]

None

* Example

```
ConnectToPrinter("portinfo...");

.....

DisconnectPrinter();
```

3-3 InitializePrinter

• Initialize the printer.

int InitializePrinter();

[Parameters]

None

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

InitializePrinter();

.....
```

3-4 FeedOneLabel

• Feed a label.

int FeedOneLabel ();

[Parameters]

None

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

ret = FeedOneLabel();

.....
```

3-5 SetSpeed

• Set the speed.

int SetSpeed(int speed);

[Parameters]

* int speed
  [in] Printing speed

Constant	Value	Description
SPEED_25	0	2.5 ips
SPEED_30	1	3.0 ips
SPEED_40	2	4.0 ips
SPEED_50	3	5.0 ips
SPEED_60	4	6.0 ips
SPEED_70	5	7.0 ips
SPEED_80	6	8.0 ips
SPEED_90	7	9.0 ips
SPEED_100	8	10.0 ips
SPEED_110	9	11.0 ips
SPEED_120	10	12.0 ips
SPEED_130	11	13.0 ips
SPEED_140	12	14.0 ips

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

SetSpeed(SPEED_60);

.....
```

3-6 SetDensity

• Set the density.

int SetDensity(int density);

[Parameters]

* int density
  [in] Printing density (0 ~ 20)

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

SetDensity(20);

.....
```

3-7 SetOrientation

• Set the orientation.

int SetOrientation(int orientation);

[Parameters]

* int orientation
  [in] Printing orientation

Constant	Value	Description
TOP_TO_BOTTOM	84	Print from top to bottom.
BOTTOM_TO_TOP	66	Print from bottom to top.

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

SetOrientation(TOP_TO_BOTTOM);

.....
```

3-8 SetCutter

• Set the cutting options.

int SetCutter(bool autoCut, int cutPeriod);

[Parameters]

* bool autoCut
  [in] Enable/disable auto cutting. false: disable auto cutting, true: enable auto cutting

* int cuttingPeriod
  [in] Cut interval

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

SetCutter(false, 0);

.....
```

3-9 SetBackFeed

• Set the back-feeding options.

int SetBackFeed(bool backFeed);

[Parameters]

* bool backFeed
  [in] Enable/disable back-feed. false: disable back-feed, true: enable back-feed

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-300	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

SetBackFeed(false);

.....
```

3-10 SetPaper

• Set the paper options.

int SetPaper(int width, int height, int mediaType, int offset, int gapLength);

[Parameters]

* int width
  [in] Paper width. max. 832(4.1 inch) [dot]

* int height
  [in] Paper height. max. 2432(12 inch) [dot]

* int mediaType
  [in] Paper type

Constant	Value	Description
MEDIA_GAP	0	Gap
MEDIA_CONTINUOUS	1	Continuous
MEDIA_BLACKMARK	2	Black Mark

* int offset
  [in] Gap or Blackmark offset

* int gapLength
  [in] Gap length or Blackmark depth [dot]

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

SetPaper(832, 1261, MEDIA_GAP, 0, 20);

.....
```

3-11 SetMargin

• Set the paper margins.

int SetMargin(int horizontalMargin, int verticalMargin);

[Parameters]

* int horizontalMargin
  [in] Horizontal margin [dot]

* int verticalMargin
  [in] Vertical margin [dot]

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

SetMargin(10, 10);

.....
```

3-12 SetOffset

• Set the offset between Black Mark/Gap and cutting line.

int SetOffset(int offset);

[Parameters]

* int offset
  [in] Offset length [dot] (-100 ~ 100)

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

SetOffset(10);

.....
```

3-13 PrintDeviceFont

• Print the device font.

int PrintDeviceFont(const char *text, const int xPos, const int yPos, const int fontName,
const int xMulti, const int yMulti, const int rotation, const bool bold);

[Parameters]

* const char *text
[in] Character strings to print

* const int xPos
[in] Horizontal position (X) [dot]

* const int yPos
[in] Vertical position (Y) [dot]

* const int fontName
[in] Font name

Constant	Value	Description
DEVICE_ENG_9X15	0	Size 6 (9 X 15)
DEVICE_ENG_12X20	1	Size 8 (12 X 20)
DEVICE_ENG_16X25	2	Size 10 (16 X 25)
DEVICE_ENG_19X30	3	Size 12 (19 X 30)
DEVICE_ENG_24X38	4	Size 15 (24 X 38)
DEVICE_ENG_32X50	5	Size 20 (32 X 50)
DEVICE_ENG_48X76	6	Size 30 (48 X 76)
DEVICE_ENG_22X34	7	Size 14 (22 X 34)
DEVICE_ENG_28X44	8	Size 18 (28 X 44)
DEVICE_ENG_37X58	9	Size 24 (37 X 58)
DEVICE_KOR_16X16	0x61	Size 1 (16 X 16)
DEVICE_KOR_24X24	0x62	Size 2 (24 X 24)
DEVICE_KOR_20X20	0x63	Size 3 (20 X 20)
DEVICE_KOR_26X26	0x64	Size 4 (26 X 26)
DEVICE_KOR_20X26	0x65	Size 5 (20 X 26)
DEVICE_KOR_38X38	0x66	Size 6 (38 X 38)
DEVICE_CHN_GB2312	0x6D	GB2312 (24 X 24)
DEVICE_CHN_BIG5	0x6E	BIG5 (24 X 24)

* const int xMulti
  [in] Horizontal zoom in (1 ~ 4)

* const int yMulti
  [in] Vertical zoom in (1 ~ 4)

* const int rotation
  [in] Rotation value

Constant	Value	Description
ROTATE_0	0	Rotate by 0 degree
ROTATE_90	1	Rotate by 90 degrees
ROTATE_180	2	Rotate by 180 degrees
ROTATE_270	3	Rotate by 270 degrees

* bool bold
  [in] Bold. false: disable, true: enable

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

PrintDeviceFont("BIXOLON LABEL SDK TEST.", 20, 10, DEVICE_ENG_24X38,
                1, 1, ROTATE_0, false);
Prints(1, 1);

.....
```

3-14 PrintDeviceFontW

• Print the device font. (Use UTF-8)

int PrintDeviceFontW(const char *text, const int xPos, const int yPos, const int fontName, const int xMulti, const int yMulti, const int rotation, const bool bold, const unsigned codePage);

[Parameters]

* const char *text
  [in] Character strings to print

* const int xPos
  [in] Horizontal position (X) [dot]

* const int yPos
  [in] Vertical position (Y) [dot]

* const int fontName
  [in] Font name

Constant	Value	Description
DEVICE_ENG_9X15	0	Size 6 (9 X 15)
DEVICE_ENG_12X20	1	Size 8 (12 X 20)
DEVICE_ENG_16X25	2	Size 10 (16 X 25)
DEVICE_ENG_19X30	3	Size 12 (19 X 30)
DEVICE_ENG_24X38	4	Size 15 (24 X 38)
DEVICE_ENG_32X50	5	Size 20 (32 X 50)
DEVICE_ENG_48X76	6	Size 30 (48 X 76)
DEVICE_ENG_22X34	7	Size 14 (22 X 34)
DEVICE_ENG_28X44	8	Size 18 (28 X 44)
DEVICE_ENG_37X58	9	Size 24 (37 X 58)
DEVICE_KOR_16X16	0x61	Size 1 (16 X 16)
DEVICE_KOR_24X24	0x62	Size 2 (24 X 24)
DEVICE_KOR_20X20	0x63	Size 3 (20 X 20)
DEVICE_KOR_26X26	0x64	Size 4 (26 X 26)
DEVICE_KOR_20X26	0x65	Size 5 (20 X 26)
DEVICE_KOR_38X38	0x66	Size 6 (38 X 38)
DEVICE_CHN_GB2312	0x6D	GB2312 (24 X 24)
DEVICE_CHN_BIG5	0x6E	BIG5 (24 X 24)

* const int xMulti
  [in] Horizontal zoom in (1 ~ 4)

* const int yMulti
  [in] Vertical zoom in (1 ~ 4)

* const int rotation
  [in] Rotation value

Constant	Value	Description
ROTATE_0	0	Rotate by 0 degree
ROTATE_90	1	Rotate by 90 degrees
ROTATE_180	2	Rotate by 180 degrees
ROTATE_270	3	Rotate by 270 degrees

* bool bold
  [in] Bold. false: disable, true: enable

* const unsigned int codePage
  [in] Set the encoding type for character strings.

Constant	Value	Description
CP_EUCKR	0	Korean (EUC-KR)
CP_CP949	1	Korean (CP949)
CP_EUCCN	2	Chinese (EUC-CN)
CP_GB18030	3	Chinese (GB18030)
CP_BIG5	4	Chinese (BIG5)
CP_CP950	5	Chinese (CP950)
CP_EUCJP	6	Japanese (EUC-JP)
CP_CP932	7	Japanese (CP932)
CP_CP874	8	Thai (CP874)

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

PrintDeviceFontW("BIXOLON Label Printer SDK Test.", 20, 10, DEVICE_ENG_24X38,
                 1, 1, ROTATE_0, false, CP_CP949);
Prints(1, 1);

.....
```

3-15 PrintVectorFont

• Print the vector font.

```
int PrintVectorFont(const char *text, const int xPos, const int yPos, const int font,
                    const int fontWidth, const int fontHeight, const int rightSpace,
                    const bool bold, const bool reverse, const bool italic,
                    const int rotation, const int alignment, const int printDirection);
```

[Parameters]

* const char *text
  [in] Character strings to print

* const int xPos
  [in] Horizontal position (X) [dot]

* const int yPos
  [in] Vertical position (Y) [dot]

* const int font
  [in] Font option

Constant	Value	Description
VECTOR_ASCII	0	ASCII (1Byte Code)
VECTOR_KS5601	1	KS5601(2Byte Code)
VECTOR_BIG5	2	BIG5(2Byte Code)
VECTOR_GB2312	3	GB2312(2Byte Code)
VECTOR_JIS	4	Shift-JIS(2Byte Code)
VECTOR_OCRA	5	OCR-A(1Byte Code)
VECTOR_OCRB	6	OCR-B(1Byte Code)

* const int fontWidth
  [in] Font width [dot]

* const int fontHeight
  [in] Font height [dot]

* const int rightSpace
  [in] Right space [dot]. + / - option can be used.

* const bool bold
  [in] Bold. false: disable, true: enable

* const bool reverse
[in] Reverse. false: disable, true: enable

* const bool italic
[in] Italic. false: disable, true: enable

* const int rotation
[in] Rotation value

Constant	Value	Description
ROTATE_0	0	Rotate by 0 degree
ROTATE_90	1	Rotate by 90 degrees
ROTATE_180	2	Rotate by 180 degrees
ROTATE_270	3	Rotate by 270 degrees

* const int alignment
[in] text alignment

Constant	Value	Description
ALIGNMENT_LEFT	0	Align to the left
ALIGNMENT_CENTER	1	Align to the center
ALIGNMENT_RIGHT	2	Align to the right

* int printDirection
[in] print direction for character strings

Constant	Value	Description
LEFT_TO_RIGHT	0	Print from left to right (ex. BIXOLON)
RIGHT_TO_LEFT	1	Print from right to left (ex. NOLOXIB)

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

PrintVectorFont("VECTOR FONT.", 20, 150, VECTOR_ASCII, 10, 10, 3, false, false,
                false, ROTATE_0, ALIGNMENT_LEFT, LEFT_TO_RIGHT);
Prints(1, 1);

.....
```

3-16 PrintVectorFontW

• Print the vector font. (Use UTF-8)

```
int PrintVectorFontW(const char *text, const int xPos, const int yPos, const int font,
                     const int fontWidth, const int fontHeight, const int rightSpace,
                     const bool bold, const bool reverse, const bool italic,
                     const int rotation, const int alignment, const int printDirection,
                     const unsigned int codePage);
```

[Parameters]

* const char *text
  [in] Character strings to print

* const int xPos
  [in] Horizontal position (X) [dot]

* const int yPos
  [in] Vertical position (Y) [dot]

* const int font
  [in] Font option

Constant	Value	Description
VECTOR_ASCII	0	ASCII (1Byte Code)
VECTOR_KS5601	1	KS5601(2Byte Code)
VECTOR_BIG5	2	BIG5(2Byte Code)
VECTOR_GB2312	3	GB2312(2Byte Code)
VECTOR_JIS	4	Shift-JIS(2Byte Code)
VECTOR_OCRA	5	OCR-A(1Byte Code)
VECTOR_OCRB	6	OCR-B(1Byte Code)

* const int fontWidth
  [in] Font width [dot]

* const int fontHeight
  [in] Font height [dot]

* const int rightSpace
  [in] Right space [dot]. + / - option can be used.

* const bool bold
  [in] Bold. false: disable, true: enable

* const bool reverse
  [in] Reverse. false: disable, true: enable

* const bool italic
  [in] Italic. false: disable, true: enable

* const int rotation
  [in] Rotation value

Constant	Value	Description
ROTATE_0	0	Rotate by 0 degree
ROTATE_90	1	Rotate by 90 degrees
ROTATE_180	2	Rotate by 180 degrees
ROTATE_270	3	Rotate by 270 degrees

* const int alignment
  [in] Text alignment

Constant	Value	Description
ALIGNMENT_LEFT	0	Align to the left
ALIGNMENT_CENTER	1	Align to the center
ALIGNMENT_RIGHT	2	Align to the right

* int printDirection
  [in] print direction for character strings

Constant	Value	Description
LEFT_TO_RIGHT	0	Print from left to right (ex. BIXOLON)
RIGHT_TO_LEFT	1	Print from right to left (ex. NOLOXIB)

* const unsigned int codePage
  [in] Set the encoding type for character strings

Constant	Value	Description
CP_EUCKR	0	Korean (EUC-KR)
CP_CP949	1	Korean (CP949)
CP_EUCCN	2	Chinese (EUC-CN)
CP_GB18030	3	Chinese (GB18030)
CP_BIG5	4	Chinese (BIG5)
CP_CP950	5	Chinese (CP950)
CP_EUCJP	6	Japanese (EUC-JP)
CP_CP932	7	Japanese (CP932)
CP_CP874	8	Thai (CP874)

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

PrintVectorFontW("BIXOLON Label Printer SDK Test.", 20, 150, VECTOR_ASCII, 10,
                 10, 3, false, false, false, ROTATE_0, ALIGNMENT_LEFT,
                 LEFT_TO_RIGHT, CP_CP949);
Prints(1, 1);

.....
```

3-17 Prints

• Print the buffer.

int Prints(const int nLabelSet, const int nCopies);

[Parameters]

* const int nLabelSet
  [in] Number of label sets (1 ~ 65535)

* const int nCopies
  [in] Number of label copies (1 ~ 65535)

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

Prints(1, 1);

.....
```

3-18 Print1DBarcode

• Print 1-dimensional barcode.

int Print1DBarcode(const char *barcodeData, const int xPos, const int yPos,
const int barcodeType, const int narrowBarWidth,
const int wideBarWidth, const int barcodeHeight, const int rotation,
const int HRI);

[Parameters]

* const char *barcodeData
[in] Barcode printing data

* const int xPos
[in] Horizontal position (X) [dot]

* const int yPos
[in] Vertical position (Y) [dot]

* const int barcodeType
[in] Define barcode type. Defined in bxlConst.h

Constant	Value	Description
BAR_CODE39	0	Code39
BAR_CODE128	1	Code128
BAR_I2OF5	2	Interleaved 2of5
BAR_CODABAR	3	Codabar
BAR_CODE93	4	Code93
BAR_UPCA	5	UPC-A
BAR_UPCE	6	UPC-E
BAR_EAN13	7	EAN13
BAR_EAN8	8	EAN8
BAR_EAN128	9	UCC/EAN128
BAR_CODE11	10	Code11
BAR_PLANET	11	Planet
BAR_INDUSTRIAL2OF5	12	Industrial 2of5
BAR_STANDARD2OF5	13	Standard 2of5
BAR_LOGMARS	14	logmars
BAR_EXTENSION	15	UPC/EAN Extensions
BAR_POSTNET	16	Postnet

* int narrowBarWidth
[in] Set the narrow bar width in the unit of Dot.

* int wideBarWidth
  [in] Set the wide bar width in the unit of Dot.

* int barcodeHeight
  [in] Set the barcode height in the unit of Dot

* int rotation
  [in] Set the rotation value

Constant	Value	Description
ROTATE_0	0	Rotate by 0 degree
ROTATE_90	1	Rotate by 90 degrees
ROTATE_180	2	Rotate by 180 degrees
ROTATE_270	3	Rotate by 270 degrees

* int HRI
  [in] Set the HRI printing position and size in the range between 0 and 8.

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....
Print1DBarcode("123456789012", 220, 60, BXL_1DBAR_CODE39, 2, 5, 100, BXL_ROTATE_90, 1);
Prints(1, 1);

Print1DBarcode("123456789012", 220, 60, BXL_1DBAR_CODE128, 2, 5, 100, BXL_ROTATE_90, 1);
Prints(1, 1);

Print1DBarcode("123456789012", 220, 60, BXL_1DBAR_I2OF5, 2, 5, 100, BXL_ROTATE_90, 1);
Prints(1, 1);

Print1DBarcode("123456789012", 220, 60, BXL_1DBAR_CODABAR, 2, 5, 100, BXL_ROTATE_90, 1);
Prints(1, 1);

Print1DBarcode("123456789012", 220, 60, BXL_1DBAR_CODE93, 2, 5, 100, BXL_ROTATE_90, 1);
Prints(1, 1);

Print1DBarcode("123456789012", 220, 60, BXL_1DBAR_UPCA, 2, 5, 100, BXL_ROTATE_90, 1);
Prints(1, 1);

Print1DBarcode("123456789012", 220, 60, BXL_1DBAR_UPCE, 2, 5, 100, BXL_ROTATE_90, 1);
Prints(1, 1);

Print1DBarcode("123456789012", 220, 60, BXL_1DBAR_EAN13, 2, 5, 100, BXL_ROTATE_90, 1);
Prints(1, 1);

Print1DBarcode("123456789012", 220, 60, BXL_1DBAR_EAN8, 2, 5, 100, BXL_ROTATE_90, 1);
Prints(1, 1);

Print1DBarcode("123456789012", 220, 60, BXL_1DBAR_EAN128, 2, 5, 100, BXL_ROTATE_90, 1);
Prints(1, 1);

.....
```

3-19 PrintMaxiCode

• Print 2-dimensional barcode (Maxicode).

int PrintMaxiCode(const char *barcodeData, const int xPos, const int yPos,
const int mode);

[Parameters]

* const char *barcodeData
[in] Barcode printing data

* const int xPos
[in] Horizontal position (X) [dot]

* const int yPos
[in] Vertical position (Y) [dot]

* const int mode
[in] Maxicode mode

Constant	Value	Description
MAXICODE_MODE0	0	Mode 0
MAXICODE_MODE90	2	Mode 2
MAXICODE_MODE180	3	Mode 3
MAXICODE_MODE270	4	Mode 4

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....
PrintMaxiCode("990,840,06840,THIS IS A TEST OF MODE 0 BIXOLON LABEL
                PRINTER", 30, 100, MAXICODE_MODE0);
Prints(1, 1);

.....
```

3-20 PrintPDF417

• Print 2-dimensional barcode (PDF417).

```
int PrintBarcode(const char *barcodeData, const int xPos, const int yPos,
                 const int verticalCount, const int horizontalCount, int errorLevel,
                 int dataComp, bool HRI, int startPosition, int moduleWidth,
                 int barHeight, int rotation);
```

[Parameters]

* const char *barcodeData
  [in] Barcode printing data

* const int xPos
  [in] Horizontal position (X) [dot]

* const int yPos
  [in] Vertical position (Y) [dot]

* int verticalCount
  [in] Max. vertical count: 3 ~ 90

* int horizontalCount
  [in] Max. vertical count: 1 ~ 30

* int errorLevel
  [in] Error correction level. Defined in bxlConst.h.

Constant	Value	Description
PDF417_ECL0	0	EC Level: 0. EC Codeword: 2
PDF417_ECL1	1	EC Level: 1. EC Codeword: 4
PDF417_ECL2	2	EC Level: 2. EC Codeword: 8
PDF417_ECL3	3	EC Level: 3. EC Codeword: 16
PDF417_ECL4	4	EC Level: 4. EC Codeword: 32
PDF417_ECL5	5	EC Level: 5. EC Codeword: 64
PDF417_ECL6	6	EC Level: 6. EC Codeword: 128
PDF417_ECL7	7	EC Level: 7. EC Codeword: 256
PDF417_ECL8	8	EC Level: 8. EC Codeword: 512

* int dataComp
[in] Data compression method. Defined in bxlConst.h.

Constant	Value	Description
PDF417_COMP_TEXT	0	2 Characters per codeword.
PDF417_COMP_NUM	1	2.93 Characters per codeword.
PDF417_COMP_BINARY	2	1.2 Bytes per codeword.

* bool HRI
[in] Set the HRI printing option

* int startPosition
[in] 0: starts from the center of the barcode, 1: starts from the top left corner of the barcode

* int moduleWidth
[in] Set the module width (2 ~ 9).

* int barHeight
[in] Set the bar height (4 ~ 99).

* int rotation
[in] Set the rotation value.

Constant	Value	Description
ROTATE_0	0	Rotate by 0 degree
ROTATE_90	1	Rotate by 90 degrees
ROTATE_180	2	Rotate by 180 degrees
ROTATE_270	3	Rotate by 270 degrees

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....
PrintPDF417("BIXOLON Label Printer, This is Test Printing.", 30, 100, 30, 5,
                    PDF417_ECL0, PDF417_COMP_TEXT, true, 1, 3, 10, ROTATE_0);
Prints(1, 1);

.....
```

3-21 PrintQRCode

• Print 2-dimensional barcode (QR Code).

int PrintQRCode(const char *barcodeData, const int xPos, const int yPos, int model,
const int eccLevel, int barSize, int rotation);

[Parameters]

* const char *barcodeData
[in] Barcode printing data

* const int xPos
[in] Horizontal position (X) [dot]

* const int yPos
[in] Vertical position (Y) [dot]

* int model
[in] Model option. 1: Model 1, 2: Model 2

* int eccLevel
[in] ECC level

Constant	Value	Description
QRCODE_ECC7	0	Recovery rate 7%
QRCODE_ECC15	1	Recovery rate 15%
QRCODE_ECC25	2	Recovery rate 25%
QRCODE_ECC30	3	Recovery rate 30%

* int barSize
[in] Barcode size setting (1 ~ 4).

* int rotation
[in] Rotation value

Constant	Value	Description
ROTATE_0	0	Rotate by 0 degree
ROTATE_90	1	Rotate by 90 degrees
ROTATE_180	2	Rotate by 180 degrees
ROTATE_270	3	Rotate by 270 degrees

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

......
PrintQRCode("ABCDEFGHIJKLMN1234567890", 30, 100, 1, QRCODE_ECC7,
                4, ROTATE_0);
Prints(1, 1);

......
```

3-22 PrintDataMatrix

• Print 2-dimensional barcode (Data Matrix).

int PrintDataMatrix(const char *barcodeData, const int xPos, const int yPos,
int barSize, bool reverse, int rotation);

[Parameters]

* const char *barcodeData
[in] Barcode printing data

* const int xPos
[in] Horizontal position (X) [dot]

* const int yPos
[in] Vertical position (Y) [dot]

* int barSize
[in] Barcode size setting (1 ~ 4)

* bool reverse
[in] Barcode reverse. false: disable, true: enable

* int rotation
[in] Rotation value

Constant	Value	Description
ROTATE_0	0	Rotate by 0 degree
ROTATE_90	1	Rotate by 90 degrees
ROTATE_180	2	Rotate by 180 degrees
ROTATE_270	3	Rotate by 270 degrees

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....
PrintDataMatrix("BIXOLON Label Printer", 30, 100, 4, false, ROTATE_0);
Prints(1, 1);

.....
```

3-23 PrintAztec

• Print 2-dimensional barcode (Aztec).

```
int PrintAztec(const char *barcodeData, const int xPos, const int yPos, int barSize,
               int interpretation, int errCodeNSymbolSize, bool menuSymbol,
               int numOfSymbol, int optID, int rotation);
```

[Parameters]

* const char *barcodeData
  [in] Barcode printing data

* const int xPos
  [in] Horizontal position (X) [dot]

* const int yPos
  [in] Vertical position (Y) [dot]

* int barSize
  [in] Barcode size setting (1 ~ 10).

* int interpretation
  [in] ECI (Extended Channel Interpretation) code setting. 0: disable, 1: enable.

* int errCodeNSymbolSize
  [in] Error code and symbol size/type

Value	Description
0	Default error collection level
1 ~ 99	Error collection percent
101 ~ 104	1 ~ 4 layer compact symbol
201 ~ 232	1 ~ 32 layer full range symbol
300	Simple Aztec “Rune”

* bool menuSymbol
  [in] Menu symbol

* bool numOfSymbol
  [in] Number of symbols for structured append (1 ~ 26)

* int optID
  [in] Optional ID filed for structured append : ID field string (max. 24 characters)

* int rotation
  [in] Rotation value

Constant	Value	Description
ROTATE_0	0	Rotate by 0 degree
ROTATE_90	1	Rotate by 90 degrees
ROTATE_180	2	Rotate by 180 degrees
ROTATE_270	3	Rotate by 270 degrees

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....
PrintAztec("THIS IS AZTEC BARCODE TESTTHIS IS AZTEC BARCODE TEST",
                30, 100, 5, 0, 0, true, 1, 1, ROTATE_0);
Prints(1, 1);

.....
```

3-24 PrintCode49

• Print 2-dimensional barcode (Code49).

```
int PrintCode49(const char *barcodeData, const int xPos, const int yPos,
                const int narrowWidth, const int wideWidth, const int barHeight,
                int HRI, int startingMode, int rotation);
```

[Parameters]

* const char *barcodeData
  [in] Barcode printing data

* const int xPos
  [in] Horizontal position (X) [dot]

* const int yPos
  [in] Vertical position (Y) [dot]

* const int narrowWidth
  [in] Narrow bar width [dot]

* const int wideWidth
  [in] Wide bar width [dot]

* const int barHeight
  [in] Barcode height [dot]

* int HRI
  [in] HRI printing. 0: No printing, 1: Below the barcode, 2: Above the barcode

* int startingMode
  [in] starting mode

Value	Description
0	Regular Alphanumeric Mode
1	Multiple Read Alphanumeric
2	Regular Numeric Mode
3	Group Alphanumeric Mode
4	Regular Alphanumeric Shift 1
5	Regular Alphanumeric Shift 2
7	Automatic Mode

* int rotation
  [in] Rotation value

Constant	Value	Description
ROTATE_0	0	Rotate by 0 degree
ROTATE_90	1	Rotate by 90 degrees
ROTATE_180	2	Rotate by 180 degrees
ROTATE_270	3	Rotate by 270 degrees

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....
PrintCode49("12345ABC", 30, 100, 2, 7, 22, 2, 7, ROTATE_0);

Prints(1, 1);

.....
```

3-25 PrintCODABLOCK

• Print 2-dimensional barcode (CODABLOCK).

int PrintCODABLOCK(const char *barcodeData, const int xPos, const int yPos,
const int narrowWidth, const int wideWidth, const int barHeight,
const bool security, int dataColumns, int mode, int encodeRow);

[Parameters]

* const char *barcodeData
[in] Barcode printing data

* const int xPos
[in] Horizontal position (X) [dot]

* const int yPos
[in] Vertical position (Y) [dot]

* const int narrowWidth
[in] Narrow bar width [dot]

* const int wideWidth
[in] Wide bar width [dot]

* const int barHeight
[in] Barcode height [dot]

* const bool security
[in] Security function

* int dataColumns
[in] Number of characters per line (2 ~ 62)

* int mode
[in] Mode

Constant	Value	Description
CODABLOCK_A	0	Use Code 39 character set
CODABLOCK_E	1	Use Code 128 character set
CODABLOCK_F	2	Add Code 128 character set and Function 1 (FNC1) automatically

* int encodeRow
[in] Number of lines to encode

Value	Description
A	1 ~ 18
E	2 ~ 4
F	2 ~ 4

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

PrintCODABLOCK("BIXOLON BARCODE TEST 123BIXOLON BARCODE TEST 123",
                30, 100, 2, 5, 30, false, 30, CODABLOCK_E, 4);
Prints(1, 1);

.....
```

3-26 PrintMicroPDF

• Print 2-dimensional barcode (Micro-PDF417).

int PrintMicroPDF(const char *barcodeData, const int xPos, const int yPos,
int moduleWidth, int barHeight, int mode, int rotation);

[Parameters]

* const char *barcodeData
[in] Barcode printing data

* const int xPos
[in] Horizontal position (X) [dot]

* const int yPos
[in] Vertical position (Y) [dot]

* int moduleWidth
[in] Module width (2 ~ 8)

* int barHeight
[in] Barcode height (1 ~ 99) [dot]

* int mode
[in] Mode (0 ~ 33), refer to the command manual for details.

* int rotation
[in] Rotation value

Constant	Value	Description
ROTATE_0	0	Rotate by 0 degree
ROTATE_90	1	Rotate by 90 degrees
ROTATE_180	2	Rotate by 180 degrees
ROTATE_270	3	Rotate by 270 degrees

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

PrintMicroPDF("ABCDEFGHIJKLMN1234567890", 30, 100, 2, 6, 8, ROTATE_0);
Prints(1, 1);

.....
```

3-27 PrintGS1DataBar

• Print GS1 DataBar barcode.

```
int PrintGS1DataBar(const char *barcodeData, const int xPos, const int yPos,
                    int barcodeType, int expand, int separatorHeight,
                    int barHeight, int segmentWidth, int rotation);
```

[Parameters]

* const char *barcodeData
  [in] Print GS1 DataBar barcode.

* const int xPos
  [in] Horizontal position (X) [dot]

* const int yPos
  [in] Vertical position (Y) [dot]

* int barcodeType
  [in] Barcode type

Constant	Value	Description
GS1DATABAR	0	GS1 DataBar
GS1DATABAR_TRUNCATED	1	GS1 DataBar Truncated
GS1DATABAR_STACKED	2	GS1 DataBar Stacked
GS1DATABAR_STACKED_OMNIDIRECTIONAL	3	GS1 DataBar Stacked Omnidirectional
GS1DATABAR_LIMITED	4	GS1 Limited
GS1DATABAR_EXPANDED	5	GS1 Expanded

* int expand
  [in] Zoom in (1 ~ 10)

* int separatorHeight
  [in] Separator height (1 ~ 2)

* int barHeight
  [in] Barcode height

* int segmentWidth
  [in] Segment width (0 ~ 22. even numbers only)

* int rotation
[in] Rotation value

Constant	Value	Description
ROTATE_0	0	Rotate by 0 degree
ROTATE_90	1	Rotate by 90 degrees
ROTATE_180	2	Rotate by 180 degrees
ROTATE_270	3	Rotate by 270 degrees

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");
.....

PrintGS1DataBar("0102005190000570031512291036310300050091320000050304",
                30, 100, GS1DATABAR_EXPANDED, 2, 2, 17, 10, ROTATE_0);
Prints(1, 1);
.....
```

3-28 PrintBlock

• Print lines, blocks, boxes and slopes.

int PrintBlock(const int xStart, const int yStart, const int xEnd, const int yEnd,
const int option, const int thickness);

[Parameters]

* const int xStart
[in] X-axis starting coordinate [dot]

* const int yStart
[in] Y-axis starting coordinate [dot]

* const int xEnd
[in] X-axis end coordinate [dot]

* const int yEnd
[in] Y-axis end coordinate [dot]

* const int option
[in] Mode (0 ~ 33), refer to the command manual for details.

Constant	Value	Description
BLOCK_OVERWRITE	0	Line Overwriting
BLOCK_EXCLUSIVEOR	1	Line Exclusive OR
BLOCK_DELETE	2	Delete line
BLOCK_SLOPE	3	Slope
BLOCK_BOX	4	Box

* int thickness
[in] Applicable to line thickness, slope and box only.

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");
......

PrintBlock(20, 20, 300, 300, BLOCK_BOX, 10);
PrintBlock(400, 20, 20, 500, BLOCK_SLOPE, 10);

Prints(1, 1);

......
```

3-29 PrintCircle

• Print circles.

int PrintCircle(const int xPos, const int yPos, int size, int multi);

[Parameters]

* const int xPos
  [in] Horizontal position (X) [dot]

* const int yPos
  [in] Vertical position (Y) [dot]

* int size
  [in] Circle size (1 ~ 6)

* int multi
  [in] Zoom in (1 ~ 4)

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");
......

PrintCircle(150, 150, 4, 1);

Prints(1, 1);

......
```

3-30 PrinteBitmap

• Select and print image files (bmp, jpg, gif).

int PrintBitmap(const char *imagePath, const int xPos, const int yPos,
const bool compress);

[Parameters]

* const char *imagePath
[in] Image file path

* const int xPos
[in] Horizontal position (X) [dot]

* const int yPos
[in] Vertical position (Y) [dot]

* const bool compress
[in] Compression

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
IMAGE_OPEN_ERROR	-118	The image file cannot be opened.
MEM_ALLOC_ERROR	-120	The allocation of internal memory failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

char *imgPath = "....";

PrintBitmap(imgPath, 150, 150, false);

Prints(1, 1);

.....
```

3-31 DirectIO

• Send SLCS commands directly and retrieve calls.

int DirectIO(const char *writeData, const wirteLen, char *readData, int *readLen);

[Parameters]

* const char *writeData
  [in] Data to send

* const int writeLen
  [in] Length of the data to send

* char *readData
  [in] Data buffer to be read

* int *readLen
  [in] Send the length of the data to be read and receive the length of the data that has
  been read.

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.
READ_ERROR	-106	Data reception failed.

* Example

```
int ret;
char cmd = “T20,20,3,1,1,0,0,N,N,\'BIXOLON Label Printer\'”;

ret = ConnectToPrinter(“portinfo...”);

.....

DirectIO(cmd, strlen(cmd), NULL, 0);

.....
```

3-32 CalibrateMedia

• Support auto calibration.

int CalibrateMedia();

[Parameters]

None

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

CalibrateMedia();

.....
```

3-33 ResetPrinter

• Reboot the printer.

int ResetPrinter();

[Parameters]

None

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

ResetPrinter();

.....
```

3-34 ClearBuffer

• Delete the image buffer data of the printer.

int ClearBuffer();

[Parameters]

None

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

ClearBuffer();

.....
```

3-35 SetRewinder

• This function sets the whether to use of Rewinder.

int SetRewinder(bool rewind);

[Parameters]

* bool rewind
  [in] Rewinder whether to use (true: Use / false: Not Use)

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnectToPrinter("portinfo...");

.....

SetRewinder(true);

.....
```

3-36 SetupRFID

• This function sets the RFID options of printer

int SetupRFID(int rfidType, int numberOfRetries, int numberOfLabel, int radioPower)

[Parameters]

* int rfidType
[in] RFID transponder type

Value	Description
0	None
1	ISO 18000-6 Type A
2	ISO 18000-6 Type B
3	EPC Class 0
4	EPC Class 1
5	EPC Class Generation 2

* int numberOfRetries
[in] Number of retries in case of coding failure.

* int numberOfLabel
[in] Number of retry labels in case of RFID write failure

* int radioPower
[in] strength of transmitting and receiving (0: Maximum, 30: Minimum, Default: 15)

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnecToPrinter("portinfo...");

.....

SetupRFID(5, 3, 2, 15);

.....
```

3-37 SetRFIDPosition

• This function allows you to set the coding position of the printer RFID label directly.

int SetRFIDPosition(int transPosition)

[Parameters]

* int transPosition
  [in] coding postion of RFID label(Value of y axis)

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnecToPrinter("portinfo...");

.....

SetRFIDPosition(320);

.....
```

3-38 SetEPCDataStructure

• This function define that EPC Data structure for writing RFID EPC Data.

int SetEPCDataStructure(int totalSize, const char* fieldSizes)

[Parameters]

* int totalSize
  [in] Total byte of field

* const char* fieldSizes
  [in] Byte of each field

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnecToPrinter("portinfo...");

.....

SetEPCDataStructure(64, "2,3,14,20,25");

.....
```

3-39 WriteRFID

• This function support RFID data writing

int WriteRFID(int dataType, int startingBlockNumber, int writeBytes,
const char* writeDataString)

[Parameters]

* int dataType
[in] RFID data type

Constant	Value	Description
RFID_ASCII	1	ASCII
RFID_HEXADECIMAL	2	Hexadecimal
RFID_USER	3	User area Hexadecimal
RFID_EPC	4	EPC

* int startingBlockNumber
[in] Number of starting block(4 ~ 10)

* int int writeBytes
[in] Length of byte to write(2 ~ 12)

* const char* writeDataString
[in] RFID data to write

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnecToPrinter("portinfo...");

.....

WriteRFID(RFID_ASCII, 4, 12, "012345678912");

.....
```

3-40 SetRFIDPassword

• This function sets RFID password of printer

int SetRFIDPassword(const char* oldAccessPwd, const char* oldkillPwd,
const char* newAccessPwd, const char* newKillPwd)

[Parameters]

* const char* oldAccessPwd
[in] Current Access Password

* const char* oldkillPwd
[in] Current Kill Password

* const char* newAccessPwd
[in] New Access Password

* const char* newKillPwd
[in] New Kill Password

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnecToPrinter("portinfo...");

.....

SetRFIDPassword("00000000", "00000000", "33333333", "33333333");


.....
```

3-41 LockRFID

• This function support RFID data lock.

int LockRFID()

[Parameters]

None

[Return Values]

Constant	Value	Description
SUCCESS	0	The function is successful.
WRITE_ERROR	-105	Data transmission failed.

* Example

```
int ret;

ret = ConnecToPrinter("portinfo...");

.....
SetRFIDPassword("00000000", "00000000", "33333333", "33333333");
LockRFID();

.....
```

Revision history

Rev.	Date	Description
2.00	2018-05-23	New
2.01	2019-01-08	1. Updated supported OS 2. Add model SPP-L310, SPP-L410
2.02	2019-03-05	1. Add model XD3-40d
2.03	2019-06-25	1. Add model XD3-40t, XD5-40d, XD5-43d, XL5-40CT, XL5-43CT
2.04	2019-10-02	1. Add model XD5-40t, XD5-43t 2. Updated XD5-40d, XD5-43d speed support 3. Correct typos and add content.
2.05	2019-11-21	1. Add model XM7-40
2.06	2020-03-12	1. Add Model XM7-20
2.07	2020-09-15	1. Updated index 2. Add model XD5-40tR/43tR, XT5-40NR/43NR/46NR, SRP-S3000 3. Add description of RFID API (3-36 ~ 3-42)
2.08	2020-11-05	Changed RFID API name(3-36 ~ 3-42)
2.09	2021-02-26	Supported print speed of SRP-S3000 is changed. : 2,3,4,5,6 ips -> 3,4,5,6,7,8 ips Update description of PrintVectorFontW,PrintDeviceFontW API : Remove information of Thai codepage
2.10	2021-08-11	1. Add model XT3-40, XT3-43, XM7-40R
2.11	2022-02-04	1. Add model XF3-40
2.12	2022-04-22	1. Add Model XM7-30
2.13	2025-08-29	1. Add model XD5-40IIdt, XD5-43IIdt, BD5-40dt, BD5-43dt, BT5-40, BT5-43, BT5-46, BT3-40,BT3-43