Windows Label CE SDK

PV3/PV4

BARCODE PRINTER

Ver. 1.01

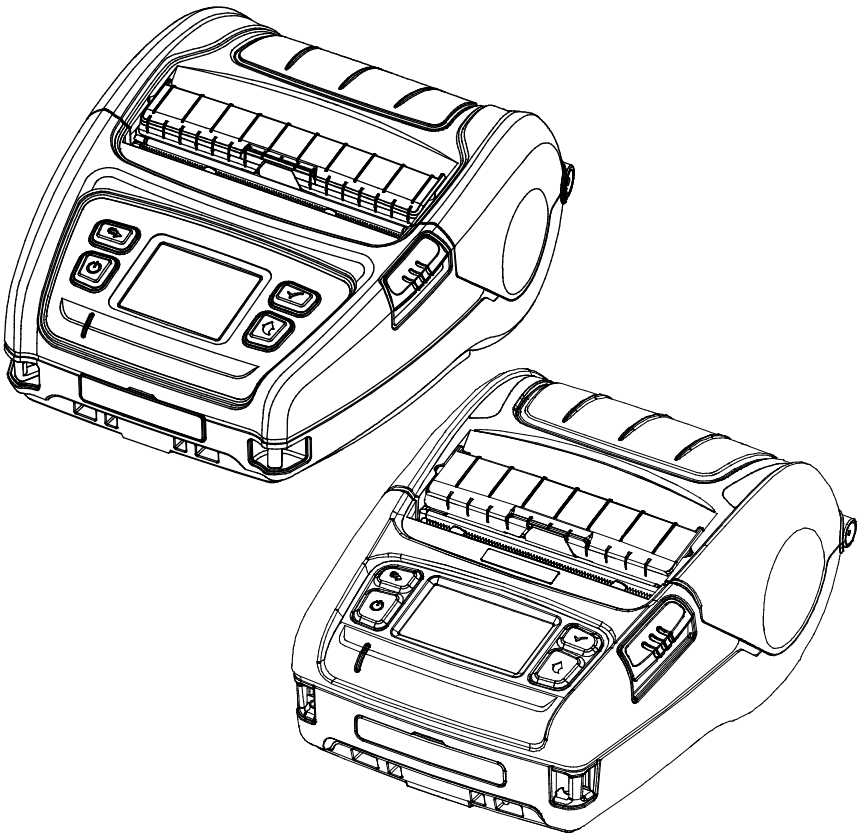

Table of Contents

1. Manual Guide

• This SDK manual describes the contents of the dll file required to develop Windows CE application programs.

1-1 Supported OS & Platform

- OS
  - • Microsoft Windows CE 5.0
  - • Microsoft Windows CE 6.0 (supported after building with custom SDK)

- Platform
  - • Windows Mobile 5
  - • Windows Mobile 6

1-2 Supported Interface

- Bluetooth, WLAN

1-3 Available range of supported models and X/Y coordinates

Model	X Coordinates		Y Coordinates
	Minimum	Maximum	Minimum
PV3	0	576	0
PV4	0	832	0

※ The maximum Y Coordinate is dependent on paper length

2. Property

• The constants used in the dll file are declared in SatoPrints.cs file.

2-1 CharacterSet (long R/W)

• This is the property for defining the printer's code page and International Character Set and the default values are set to BXL_CS_437 and BXL_ICS_USA.
The values can be set using SetCharacterSet() function.

⚠ *Note*	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

Code	Value	Description
BXL_CS_CP437	0	U.S.A
BXL_CS_CP850	1	Latin 1
BXL_CS_CP852	2	Latin 2
BXL_CS_CP860	3	Portuguese
BXL_CS_CP863	4	Canadian French
BXL_CS_CP865	5	Nordic
BXL_CS_WPC1252	6	Latin I
BXL_CS_CP865_WCP1252	7	European Combined
BXL_CS_CP857	8	Turkish
BXL_CS_CP737	9	Greek
BXL_CS_WCP1250	10	Latin 2
BXL_CS_WCP1253	11	Greek
BXL_CS_WCP1254	12	Turkish
BXL_CS_CP855	13	Cyrillic
BXL_CS_CP862	14	Hebrew
BXL_CS_CP866	15	Cyrillic
BXL_CS_WCP1251	16	Cyrillic
BXL_CS_WCP1255	17	Hebrew
BXL_CS_CP928	18	Greek
BXL_CS_CP864	19	Arabic
BXL_CS_CP775	20	Baltic
BXL_CS_WCP1257	21	Baltic
BXL_CS_CP858	22	Latin 1 + Euro

• The following International Character Set can be used.

Code	Value	Description
BXL_ICS_USA	0	Set USA code
BXL_ICS_FRANCE	1	Set FRANCE code
BXL_ICS_GERMANY	2	Set GERMANY code
BXL_ICS_UK	3	Set UK code
BXL_ICS_DENMARK_I	4	Set DENMARK1 code
BXL_ICS_SWEDEN	5	Set SWEDEN code
BXL_ICS_ITALY	6	Set ITALY code
BXL_ICS_SPAIN_I	7	Set SPAIN code
BXL_ICS_NORWAY	8	Set NORWAY code
BXL_ICS_DENMARK_II	9	Set DENMARK 2 code
BXL_ICS_JAPAN	10	Set JAPAN code
BXL_ICS_SPAIN_II	11	Set SPAIN 2 code
BXL_ICS_LATIN	12	Set LATIN code
BXL_ICS_KOREA	13	Set KOREA code
BXL_ICS_SLOVENIA	14	Set SLOVENIA code
BXL_ICS_CHINA	15	Set CHINA code

* Example

```
long lResult;

lResult = ConnectPrinter (“portinfo…”, 1000);

……

SetCharacterSet(BXL_CS_PC850, BXL_ICS_UK);

……
```

2-2 State (long R)

• This is the property that sets the printer status. It calls the CheckPrinterStatus function to check the printer status and receive the status information as a return value.
The status value can be set in duplicate and each value can be checked by bit operation.

• These are the printer status values

Code	Value	Description
BXL_STS_NORMAL	0	Normal printer status
BXL_STS_WAITTAKEN	2	Wait for paper to be taken
BXL_STS_BOARDOVERHEAT	4	Board overheat
BXL_STS_GAPERROR	8	Gap recognition error (auto sensing failure)
BXL_STS_TPHOVERHEAT	16	TPH overheat
BXL_STS_MOTOROVERHEAT	32	Motor overheat
BXL_STS_COVEROPEN	64	Cover open
BXL_STS_PAPEREMPTY	128	No paper

* Example

```
long lResult;

lResult = ConnectPrinter (“portinfo…”, 1000);
……

CheckPrinter();
……..

int iState;

iState = CheckPrinterStatus();

if ((iState & BXL_STS_BOARDOVERHEAT) == BXL_STS_BOARDOVERHEAT)
     ……
if ((iState & BXL_STS_GAPERROR) == BXL_STS_GAPERROR)
     ……
…….
```

3. Method

• The functions disclosed in the provided module are declared in the SatoPrints.cs file.

3-1 ConnectPrinter

• This function is enabled to use the module.
For PDA types with separate In/Out ports, port settings for each are required.
In general, two-way communication is possible only by strOutPortName setting.

```
long ConnectPrinter(
   LPCTSTR strOutPortName,
   long lTimeout
);
```

[Parameters]

* LPCTSTR strOutPortName
[in] Unicode Data with a null terminator. It receives the virtual serial port number and communication speed of the currently connected device as a factor.
If strInPortName is null, strOutPortName sets the In/Out data handling.

⚠ *Note*	Example of how to input Bluetooth & serial address - Bluetooth & Serial: COM4: 19200 Example of how to input WLAN address - NETxxx.xxx.xxx.xxx:9100 or ETHxxx.xxx.xxx.xxx:9100

* long lTimeout
[in] Timeout value for open attempt, processed in ms. (default : 1000ms.)

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	Unable to open communication port
BXL_CREATE_ERROR	-102	Failed to create communication target
BXL_CONNECT_ERROR	-105	Failed to connect
BXL_BAD_ARGUMENT	-108	The specified argument is not correct
BXL_WRITE_ERROR	-300	Failed to transmit data
BXL_READ_ERROR	-301	The received value is invalid, or reception failed

* Example

```
long lResult;

// serial or bluetooth
lResult = ConnectPrinter(“COM4:19200”, 1000);
……
// WLAN
lResult = ConnectPrinter(“ETHxxx.xxx.xxx.xxx:9100”, 1000);
……
```

3-2 DisconnectPrinter

• This function disables the module.
  Functions related to printing and setting cannot be used.

long DisconnectPrinter();

[Parameters]

None

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful

* Example

```
long lResult;

lResult = DisconnectPrinter();
……
```

3-3 FeedOneLabel

• This function feeds one label.

long FeedOneLabel ();

[Parameters]

None

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……

lResult = FeedOneLabel();

……
```

3-4 SetConfigOfPrinter

• This function sets the print options.

```
long SetConfigOfPrinter(
    int speed,
    int density,
    int orientation,
    bool autoCut,
    int cuttingPeriod,
    bool backFeed
);
```

[Parameters]

* int speed
  [in] print speed

Code	Value	Description
BXL_SPEED_25	0	2.5 ips
BXL_SPEED_30	1	3.0 ips
BXL_SPEED_40	2	4.0 ips
BXL_SPEED_50	3	5.0 ips
BXL_SPEED_60	4	6.0 ips
BXL_SPEED_70	5	7.0 ips
BXL_SPEED_80	6	8.0 ips

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

* int orientation
  [in] print orientation

Code	Value	Description
BXL_TOP_TO_BOTTOM	84	Print from top to bottom
BXL_BOTTOM_TO_TOP	66	Print from bottom to top

* bool autoCut
  [in] Cutting. false: no cutting, true : cutting

* int cuttingPeriod
  [in] Cutting interval

* bool backFeed
  [in] Back-feed before print start. false: no back-feed, true : back-feed

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……
SetConfigOfPrinter(BXL_SPEED_60, 20, BXL_TOP_TO_BOTTOM, false, 0, false);

……
```

3-5 SetPaper

• This function sets the options for print paper.

```
long SetPaper(
    int paperWidth,
    int paperLength,
    int mediaType,
    int offset,
    int gapLength
);
```

[Parameters]

* int paperWidth
  [in] Paper width. [dot] (Refer to: Available range of X/Y coordinates for each model)

* int paperLength
  [in] Paper length. [dot] (Refer to: Available range of X/Y coordinates for each model)

* int mediaType
  [in] Paper type

Code	Value	Description
BXL_GAP	0	Gap paper
BXL_CONTINUOUS	1	Continuous paper
BXL_BLACKMARK	2	BlackMaark paper

* int offset
  [in] Offset of gap or blackmark

* int gapLength
  [in] Gap length or blackmark thickness [dot]

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……
SetPaper(832, 1261, BXL_GAP, 0, 20);

……
```

3-6 SetMargin

• This function sets the margins of print paper.

```
long SetMargin(
    int horizontalMargin,
    int verticalMargin
);
```

[Parameters]

* int horizontalMargin
  [in] Horizontal margin [dot] (-100 ~ 100)

* int verticalMargin
  [in] Vertical margin [dot] (-100 ~ 100)

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……
SetMargin(10, 10);

……
```

3-7 ClearBuffer

• This function deletes the data in the printer's image buffer.

long ClearBuffer();

[Parameters]

None

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……
ClearBuffer();

……
```

3-8 PrintDirect

• This function sends SLCS instructions directly.

```
long PrintDirect(
   PCHAR directData
);
```

[Parameters]

* PCHAR directData
  [in] Instruction to be sent

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……
PrintDirect("T20,20,3,1,1,0,0,N,N,\SATO Label Printer\"");

……
```

3-9 PrintDeviceFont

• This function supports the printing of strings using device fonts.

```
long PrintDeviceFont(
   int xPos,
   int yPos,
   int fontName,
   int xMulti,
   int yMulti,
   int rotation,
   BOOL bold,
   LPCTSTR pData
);
```

[Parameters]

* int xPos
  [in] Horizontal position (X) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int yPos
  [in] Vertical position (Y) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int fontName
  [in] Font name

Code	Value	Description
BXL_ENG_9X15	0	Size 6 (9 X 15)
BXL_ENG_12X20	1	Size 8 (12 X 20)
BXL_ENG_16X25	2	Size 10 (16 X 25)
BXL_ENG_19X30	3	Size 12 (19 X 30)
BXL_ENG_24X38	4	Size 15 (24 X 38)
BXL_ENG_32X50	5	Size 20 (32 X 50)
BXL_ENG_48X76	6	Size 30 (48 X 76)
BXL_ENG_22X34	7	Size 14 (22 X 34)
BXL_ENG_28X44	8	Size 18 (28 X 44)
BXL_ENG_37X58	9	Size 24 (37 X 58)
BXL_KOR_16X16	0x61	Size 1 (16 X 16)
BXL_KOR_24X24	0x62	Size 2 (24 X 24)
BXL_KOR_20X20	0x63	Size 3 (20 X 20)
BXL_KOR_26X26	0x64	Size 4 (26 X 26)
BXL_KOR_20X26	0x65	Size 5 (20 X 26)
BXL_KOR_38X38	0x66	Size 6 (38 X 38)
BXL_CHN_GB2312	0x6D	GB2312 (24 X 24)
BXL_CHN_BIG5	0x6E	BIG5 (24 X 24)

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

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

* int rotation
  [in] Set the rotation value.

Code	Value	Description
BXL_ROTATE_0	0	Rotate 0 degrees
BXL_ROTATE_90	1	Rotate 90 degrees
BXL_ROTATE_180	2	Rotate 180 degrees
BXL_ROTATE_270	3	Rotate 270 degrees

* bool bold
  [in] Thickness (false: normal / true: bold)

* LPCTSTR pData
  [in] String to be printed

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……

PrintDeviceFont(20, 10, 3, 1, 1, 0, false, "SATO LABEL SDK TEST.");
Prints(1, 1);

……
```

3-10 PrintVectorFont

• This function supports the printing of strings using vector fonts.

```
long PrintVectorFont(
    int xPos,
    int yPos,
    int font,
    int fontWidth,
    int fontHeight,
    int rightSpace,
    bool bold,
    bool reverse,
    int rotation,
    int alignment,
    int printDirection,
    LPCTSTR pData
);
```

[Parameters]

* int xPos
  [in] Horizontal position (X) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int yPos
  [in] Vertical position (Y) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int font
  [in] Font selection

Code	Value	Description
BXL_VECTOR_ASCII	0	ASCII (1Byte Code)
BXL_VECTOR_KS5601	1	KS5601(2Byte Code)
BXL_VECTOR_BIG5	2	BIG5(2Byte Code)
BXL_VECTOR_GB2312	3	GB2312(2Byte Code)
BXL_VECTOR_JIS	4	Shift-JIS(2Byte Code)
BXL_VECTOR_OCRA	5	OCR-A(1Byte Code)
BXL_VECTOR_OCRB	6	OCR-B(1Byte Code)

* int fontWidth
  [in] Font width [dot]

* int fontHeight
  [in] Font height [dot]

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

* bool bold
  [in] Thickness (false: normal / true: bold)

* bool reverse
  [in] Reversed character (false: normal / true: reversed)

* int rotation
  [in] Set the rotation value.

Code	Value	Description
BXL_ROTATE_0	0	Rotate 0 degrees
BXL_ROTATE_90	1	Rotate 90 degrees
BXL_ROTATE_180	2	Rotate 180 degrees
BXL_ROTATE_270	3	Rotate 270 degrees

* int alignment
  [in] Text alignment

Code	Value	Description
BXL_ALIGNMENT_LEFT	0	Rotate 0 degrees
BXL_ALIGNMENT_CENTER	1	Rotate 90 degrees
BXL_ALIGNMENT_RIGHT	2	Rotate 180 degrees

* int printDirection
  [in] Printing direction of character strings.
  0: Print from left to right (ex. SATO)
  1: Print from right to left (ex. OTAS)

* LPCTSTR pData
  [in] String to be printed

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……

PrintVectorFont(20, 150, BXL_VECTOR_ASCII, 10, 10, 3, false, false, false, 0, 0, 0,
"VECTOR FONT.");
Prints(1, 1);

……
```

3-11 Prints

• This method prints the contents stored in the output buffer of the printer.

```
long Prints(
   int nLabelSet,
   int nCopies
);
```

[Parameters]

* int nLabelSet
  [in] Number of Label Set (1 ~ 65535)

* int nCopies
  [in] Number of Label Copies (1 ~ 65535)

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……
Prints(1, 1);

……
```

3-12 Print1DBarcode

• This function supports the printing of one-dimensional barcodes.

```
long Print1DBarcode(
    int xPos,
    int yPos,
    int barcodeType,
    int narrowBarWidth,
    int wideBarWidth,
    int barcodeHeight,
    int rotation,
    int HRI,
    LPCTSTR barcodeData
);
```

[Parameters]

* int xPos
  [in] Horizontal position (X) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int yPos
  [in] Vertical position (Y) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int barcodeType
  [in] Define the barcode type. It is defined in SatoPrints.cs.

Code	Value	Description
BXL_1DBAR_CODE39	0	Code39
BXL_1DBAR_CODE128	1	Code128
BXL_1DBAR_I2OF5	2	Interleaved 2of5
BXL_1DBAR_CODABAR	3	Codabar
BXL_1DBAR_CODE93	4	Code93
BXL_1DBAR_UPCA	5	UPC-A
BXL_1DBAR_UPCE	6	UPC-E
BXL_1DBAR_EAN13	7	EAN13
BXL_1DBAR_EAN8	8	EAN8
BXL_1DBAR_EAN128	9	UCC/EAN128

* int narrowBarWidth
  [in] The value in dot units that sets the width of the narrow bar

* int wideBarWidth
  [in] The value in dot units that sets the width of the wide bar

* int barcodeHeight
  [in] The value in dot units that sets the height of the barcode

* int rotation
  [in] Set the rotation value.

Code	Value	Description
BXL_ROTATE_0	0	Rotate 0 degrees
BXL_ROTATE_90	1	Rotate 90 degrees
BXL_ROTATE_180	2	Rotate 180 degrees
BXL_ROTATE_270	3	Rotate 270 degrees

* int HRI
  [in] This is the value to set HRI output position and size and has a value from 0 to 8.

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

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

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

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

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

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

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

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

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

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

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

……
```

3-13 PrintMaxiCode

• This function supports the printing of two-dimensional barcodes (Maxicode).

```
long PrintMaxiCode(
   int xPos,
   int yPos,
   int mode,
   LPCTSTR barcodeData
);
```

[Parameters]

* int xPos
  [in] Horizontal position (X) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int yPos
  [in] Vertical position (Y) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int mode
  [in] Maxicode mode

Code	Value	Description
BXL_MAXICODE_MODE0	0	mode 0
BXL_MAXICODE_MODE90	2	mode 2
BXL_MAXICODE_MODE180	3	mode 3
BXL_MAXICODE_MODE270	4	mode 4

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……
PrintMaxiCode(30, 100, BXL_MAXICODE_MODE0, "990,840,06840,THIS IS A TEST
                OF MODE 0 SATO LABEL PRINTER");
Prints(1, 1);

……
```

3-14 PrintPDF417

• This function supports the printing of two-dimensional barcodes (PDF417).

```
long PrintPDF417(
   int xPos,
   int yPos,
   int verticalCount,
   int horizontalCount,
   int errLevel,
   int dataComp,
   BOOL HRI,
   int startPosition,
   int moduleWidth,
   int barHeight,
   int rotation,
   LPCTSTR barcodeData
);
```

[Parameters]

* int xPos
  [in] Horizontal position (X) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int yPos
  [in] Vertical position (Y) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

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

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

* int errorLevel
  [in] Error correction level. It is defined in SatoPrints.cs.

Code	Value	Description
BXL_PDF417_ECL0	0	EC Level : 0. EC Codeword : 2
BXL_PDF417_ECL1	1	EC Level : 1. EC Codeword : 4
BXL_PDF417_ECL2	2	EC Level : 2. EC Codeword : 8
BXL_PDF417_ECL3	3	EC Level : 3. EC Codeword : 16
BXL_PDF417_ECL4	4	EC Level : 4. EC Codeword : 32
BXL_PDF417_ECL5	5	EC Level : 5. EC Codeword : 64
BXL_PDF417_ECL6	6	EC Level : 6. EC Codeword : 128
BXL_PDF417_ECL7	7	EC Level : 7. EC Codeword : 256
BXL_PDF417_ECL8	8	EC Level : 8. EC Codeword : 512

* int dataComp
  [in] Data compression method. It is defined in SatoPrints.cs.

Code	Value	Description
BXL_PDF417_COMP_TEXT	0	2 Characters per codeword.
BXL_PDF417_COMP_NUM	1	2.93 Characters per codeword.
BXL_PDF417_COMP_BINARY	2	1.2 Bytes per codeword.

* bool HRI
  [in] Set whether to output HRI.

* int startPosition
  [in] When it is 0, the middle of the barcode is the starting point.
  When it is 1, the upper left of the barcode is the starting point.

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

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

* int rotation
  [in] Set the rotation value.

Code	Value	Description
BXL_ROTATE_0	0	Rotate 0 degrees
BXL_ROTATE_90	1	Rotate 90 degrees
BXL_ROTATE_180	2	Rotate 180 degrees
BXL_ROTATE_270	3	Rotate 270 degrees

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……
PrintPDF417(30, 100, 30, 5, BXL_PDF417_ECL0, BXL_PDF417_COMP_TEXT, true, 1, 3, 10,
            0, "SATO Label Printer, This is Test Printing.");
Prints(1, 1);

……
```

3-15 PrintQRCode

• This function supports the printing of two-dimensional barcodes (QR Code).

```
long PrintQRCode(
   int xPos,
   int yPos,
   int model,
   int eccLevel,
   int barSize,
   int rotation,
   LPCTSTR barcodeData
);
```

[Parameters]

* int xPos
  [in] Horizontal position (X) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int yPos
  [in] Vertical position (Y) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int model
  [in] Model selection. 1: model1, 2: model2

* int eccLevel
  [in] ECC Level

Code	Value	Description
BXL_QRCODE_ECC7	0	Recovery rate 7%
BXL_QRCODE_ECC15	1	Recovery rate 15%
BXL_QRCODE_ECC25	2	Recovery rate 25%
BXL_QRCODE_ECC30	3	Recovery rate 30%

* int barSize
  [in] Set the size of the barcode (1 ~ 4)

* int rotation
  [in] Set the rotation value.

Code	Value	Description
BXL_ROTATE_0	0	Rotate 0 degrees
BXL_ROTATE_90	1	Rotate 90 degrees
BXL_ROTATE_180	2	Rotate 180 degrees
BXL_ROTATE_270	3	Rotate 270 degrees

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……
PrintQRCode(30, 100, 1, BXL_QRCODE_ECC7, 4, 0, "ABCDEFGHIJKLMN1234567890");
Prints(1, 1);

……
```

3-16 PrintDataMatrix

• This function supports the printing of two-dimensional barcodes (Data Matrix).

```
long PrintDataMatrix(
   int xPos,
   int yPos,
   int barSize,
   bool reverse,
   int rotation,
   LPCTSTR barcodeData
);
```

[Parameters]

* int xPos
  [in] Horizontal position (X) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int yPos
  [in] Vertical position (Y) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int barSize
  [in] Set the size of the barcode (1 ~ 4)

* bool reverse
  [in] Reversed barcode image. (false: normal image / true: reversed image)

* int rotation
  [in] Set the rotation value.

Code	Value	Description
BXL_ROTATE_0	0	Rotate 0 degrees
BXL_ROTATE_90	1	Rotate 90 degrees
BXL_ROTATE_180	2	Rotate 180 degrees
BXL_ROTATE_270	3	Rotate 270 degrees

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……
PrintDataMatrix(30, 100, 4, false, 0, "SATO Label Printer");
Prints(1, 1);

……
```

3-17 PrintAztec

• This function supports the printing of two-dimensional barcodes (Aztec).

```
long PrintAztec(
   int xPos,
   int yPos,
   int barSize,
   int interpretation,
   int errCodeNSymbolSize,
   bool menuSymbol,
   int numOfSymbol,
   int optID,
   int rotation,
   LPCTSTR barcodeData
);
```

[Parameters]

* int xPos
  [in] Horizontal position (X) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int yPos
  [in] Vertical position (Y) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int barSize
  [in] Set barcode size (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 letter)

* int rotation
  [in] Set the rotation value.

Code	Value	Description
BXL_ROTATE_0	0	Rotate 0 degrees
BXL_ROTATE_90	1	Rotate 90 degrees
BXL_ROTATE_180	2	Rotate 180 degrees
BXL_ROTATE_270	3	Rotate 270 degrees

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

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

……
```

3-18 PrintCode49

• This function supports the printing of two-dimensional barcodes (Code49).

```
long PrintCode49(
    int xPos,
    int yPos,
    int narrowWidth,
    int wideWidth,
    int barHeight,
    int HRI,
    int startingMode,
    int rotation,
    LPCTSTR barcodeData
);
```

[Parameters]

* int xPos
  [in] Horizontal position (X) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int yPos
  [in] Vertical position (Y) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int narrowWidth
  [in] The width of narrow bar [dot]

* int wideWidth
  [in] The width of wide bar [dot]

* int barHeight
  [in] The height of barcode [dot]

* int HRI
  [in] HRI output. 0: No print, 1: above the barcode, 2: below 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] Set the rotation value.

Code	Value	Description
BXL_ROTATE_0	0	Rotate 0 degrees
BXL_ROTATE_90	1	Rotate 90 degrees
BXL_ROTATE_180	2	Rotate 180 degrees
BXL_ROTATE_270	3	Rotate 270 degrees

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……
PrintCode49(30, 100, 2, 7, 22, 2, 7, 0, "12345ABC");
Prints(1, 1);

……
```

3-19 PrintCODELOCK

• This function supports the printing of two-dimensional barcodes (CODELOCK).

```
long PrintCODELOCK(
    int xPos,
    int yPos,
    int narrowWidth,
    int wideWidth,
    int barHeight,
    bool security,
    int dataColumns,
    char mode,
    int encodeRow,
    LPCTSTR barcodeData
);
```

[Parameters]

* int xPos
  [in] Horizontal position (X) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int yPos
  [in] Vertical position (Y) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int narrowWidth
  [in] The width of narrow bar [dot]

* int wideWidth
  [in] The width of wide bar [dot]

* int barHeight
  [in] The height of barcode [dot]

* bool security
  [in] Security function

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

* char mode
  [in] Mode

Code	Value	Description
BXL_CODABLOCK_A	0	Use the Code 39 character set
BXL_CODABLOCK_E	1	Use the Code 128 character set
BXL_CODABLOCK_F	2	Code 128 character set and Function 1 (FNC1) automatically added

* int encodeRow
  [in] Number of lines to encode

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

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……
PrintCODELOCK(30, 100, 2, 5, 30, false, 30, 'F', 4, "SATO BARCODE TEST 123SATO
                BARCODE TEST 123SATO BARCODE TEST 123");
Prints(1, 1);

……
```

3-20 PrintMicroPDF

• This function supports the printing of two-dimensional barcodes (Micro-PDF417).

```
long PrintMicroPDF(
    int xPos,
    int yPos,
    int moduleWidth,
    int barHeight,
    int mode,
    int rotation,
    LPCTSTR barcodeData
);
```

[Parameters]

* int xPos
  [in] Horizontal position (X) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int yPos
  [in] Vertical position (Y) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int moduleWidth
  [in] The width of module (2 ~ 8)

* int barHeight
  [in] The height of barcode (1 ~ 99) [dot]

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

* int rotation
  [in] Set the rotation value.

Code	Value	Description
BXL_ROTATE_0	0	Rotate 0 degrees
BXL_ROTATE_90	1	Rotate 90 degrees
BXL_ROTATE_180	2	Rotate 180 degrees
BXL_ROTATE_270	3	Rotate 270 degrees

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……
PrintMicroPDF(30, 100, 2, 6, 8, 0, "ABCDEFGHIJKLMN1234567890");
Prints(1, 1);

……
```

3-21 PrintGS1DataBar

• This function supports the printing of GS1 DataBar barcodes.

```
long PrintGS1DataBar(
    int xPos,
    int yPos,
    int barcodeType,
    int expand,
    int separatorHeight,
    int barHeight,
    int segmentWidth,
    int rotation
    LPCTSTR barcodeData
);
```

[Parameters]

* int xPos
  [in] Horizontal position (X) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int yPos
  [in] Vertical position (Y) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int barcodeType
  [in] barcode Type

Code	Value	Description
BXL_GS1DATABAR	0	GS1 DataBar
BXL_GS1DATABAR_TRUNCATED	1	GS1 DataBar Truncated
BXL_GS1DATABAR_STACKED	2	GS1 DataBar Stacked
BXL_GS1DATABAR_STACKED_OMNIDIRECTIONAL	3	GS1 DataBar Stacked Omnidirectional
BXL_GS1DATABAR_LIMITED	4	GS1 Limited
BXL_GS1DATABAR_EXPANDED	5	GS1 Expanded

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

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

* int barHeight
  [in] barcode height

* int segmentWidth
  [in] segment width (0 ~ 22. Only an even number)

* int rotation
  [in] Set the rotation value.

Code	Value	Description
BXL_ROTATE_0	0	Rotate 0 degrees
BXL_ROTATE_90	1	Rotate 90 degrees
BXL_ROTATE_180	2	Rotate 180 degrees
BXL_ROTATE_270	3	Rotate 270 degrees

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……
PrintGS1DataBar(30, 100, BXL_GS1DATABAR_EXPANDED, 2, 2, 17, 10, 0,
                "01020051900000057003151229103631030005009132000005000300004");
Prints(1, 1);

……
```

3-22 PrintBlock

• This function supports printing lines, blocks, boxes, and oblique lines.

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

[Parameters]

* int xStart
  [in] Horizontal Start Location [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int yStart
  [in] Vertical Start Location [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int xEnd
  [in] Horizontal End Location [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int yEnd
  [in] Vertical End Location [dot]
  (Refer to: Available range of X/Y coordinates for each model)

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

Code	Value	Description
BXL_BLOCK_OVERWRITE	0	Line overwriting
BXL_BLOCK_EXCLUSIVEOR	1	Line exclusive OR
BXL_BLOCK_DELETE	2	Line deleting
BXL_BLOCK_SLOPE	3	Oblique line
BXL_BLOCK_BOX	4	Box

* int thickness
  [in] Line thickness. It is applied only for oblique lines or boxes

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter(“portinfo…”, 1000);

……
PrintBlock(20, 20, 300, 300, BXL_BLOCK_BOX, 10);
PrintBlock(400, 20, 20, 500, BXL_BLOCK_SLOPE, 10);
Prints(1, 1);

……
```

3-23 PrintCircle

• This function supports the printing of circles.

```
long PrintCircle(
    int xPos,
    int yPos,
    int size,
    int multi,
);
```

[Parameters]

* int xPos
  [in] Horizontal position (X) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int yPos
  [in] Vertical position (Y) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int size
  [in] The size of the circle (1 ~ 6)

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

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……
PrintCircle(150, 150, 4, 1);
Prints(1, 1);

……
```

3-24 PrintImage

• This function supports selecting and printing image files (bmp, jpg, gif).

```
long PrintImage(
   int xPos,
   int yPos,
   LPCTSTR filename
   bool dither,
   bool enableRLE
);
```

[Parameters]

* int xPos
  [in] Horizontal position (X) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* int yPos
  [in] Vertical position (Y) [dot]
  (Refer to: Available range of X/Y coordinates for each model)

* LPCTSTR filename
  [in] Name containing the path to the image file(ex: \\logo.jpg)

* bool dither
  [in] Whether to dither (true: Dithering / false: No Dithering)

* bool enableRLE
  [in] Whether to compress (true: Data Compression / false: No Data Compression)

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data
BXL_BITMAPLOAD_ERROR	-400	Failed to load bitmap data
BXL_BITMAPDATA_ERROR	-401	Abnormal bitmap data

* Example

```
long lResult;

lResult = ConnectPrinter("portinfo…", 1000);

……

string imgPath = "….";

PrintImage(150, 150, imgPath, false, true);
Prints(1, 1);

……
```

3-25 InitializePrinter

• This function supports initializing of the printer.

long InitializePrinter();

[Parameters]

None

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;
long status;

lResult = ConnectPrinter("portinfo…", 1000);

……
InitializePrinter();

……
```

3-26 CalibrateMedia

• This function supports auto calibration function.

long CalibrateMedia();

[Parameters]

None

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;
long status;

lResult = ConnectPrinter("portinfo…", 1000);

……

CalibrateMedia();

……
```

3-27 SetRewinder

• This function sets the whether to use of Rewinder.

long SetRewinder(bool enable);

[Parameters]

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

[Return Values]

Code	Value	Description
BXL_SUCCESS	0	The function is successful
BXL_NOT_OPENED	-101	ConnectPrinter function should be called.
BXL_WRITE_ERROR	-300	Failed to transmit data

* Example

```
long lResult;
long status;

lResult = ConnectPrinter("portinfo…", 1000);

……

SetRewinder(TRUE);

……
```

Copyright

SATO 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 should turn the printer “OFF”.

Revision History

Rev.	Date	Page	Description
1.00	19.09.19	-	New
1.01	16.04.20	-	Added PV4