Woosim Windows CE SDK Reference

Version 4.1

November 2020

Woosim systems

Contents

1. Overview

1.1. Introduction

This Woosim Windows CE Software Development Kit(SDK) document provides information about Windows Mobile application development using Woosim printers.

The Library included in the SDK is a DLL file developed based on MFC. The Library API can be referenced in different ways depending on the program language, and detailed instructions can be found in the example project included in the SDK.
This document was written based on the MFC project code.
The Library and example project were developed in MS Visual Studio 2008 environment.

Woosim printers are equipped with several types of MCU. The printer's MCU can be checked through the Self-Test function. If you turn on the printer while pressing the <FEED> button, the Self-Test function will work.
The SDK supports M16C, ARM, and RX MCU. All APIs are works on RX MCU. There are some APIs that do not work properly on M16C and ARM MCU.

Most Library APIs create a printer command or a collection of printer commands. To use commands that are not provided in the Library API, refer to the document "Woosim Command Manual".

1.2. Printing Data Buffer

The Library API operations can be classified into three categories.

1) Printer connection control
2) Printing Data Buffer control and printer command processing using it
3) Printer command processing without using Printing Data Buffer

The Printing Data Buffer is a 400KB space in the Library to store data to be transmitted to the printer. When a Library API creating printer command using the Printing Data Buffer is called, the printer command is saved in the Printing Data Buffer. The contents stored in the Printing Data Buffer are all sent to the printer at once when a specific API is called.

Some APIs do not use the Printing Data Buffer and send the printer command to the printer as soon as it is called. When developing an application program, it is necessary to distinguish between APIs that use the Printing Data Buffer and APIs that do not use the Printing Data Buffer.

1.3. Printer Mode

Woosim printers provide two modes, Standard mode and Page mode.
In the Standard mode, data sent to the printer is printed immediately.
In the Page mode, data sent to the printer is drawn in the designated area to be printed. The drawn content is printed at once when a specific printer command is received. The Library APIs that are only valid in the Page mode start with "Page_".

1.4. Deprecated APIs

Some old APIs are no longer officially supported in new versions. In this case, it is recommended to update application programs by applying new APIs as much as possible. However, if you need to keep the existing program code, you can do it by adding old APIs declaration to the *Woosim_Printer_LIB.h* file.

1.5. Definitions and Abbreviations

API	Application Interface Unit
BPP	Bits Per Pixel
DBCS	Double Byte Character System
DLL	Dynamic Link Library
ECC	Error Correction Code
HRI	Human Readable Interpretation
MCU	Main Control Unit
MFC	Microsoft Foundation Class
MSR	Magnetic Stripe Reader
SCR	Smart Card Reader
SDK	Software Development Kit
TTF	True Type Font

2. API List

2.1. Device Connection Control

```
BOOL    ClosePrinterConnection()
int     ConnectSerialPrinter(TCHAR *portName, int baudRate, int timeoutMsec,
        BOOL bProtocol)
int     ConnectWirelessPrinter(TCHAR *ipAddress, int port, int timeoutMsec,
        BOOL bProtocol)
```

2.2. Direct Transfer to Device

```
void    CancelMSRMode()
void    CancelSCRMode()
void    EnterMSRMode(int n)
void    EnterSCRMode()
void    GetFirmwareVersion()
void    GetPrinterModelName()
int     GetPrinterStatus(int timeoutMsec)
int     GetPrinterStatusEx(int timeoutMsec)
```

2.3. Printing Data Buffer Handling

```
void    ClearSpool()
int     PrintSpool(BOOL bDelete)
```

2.4. Append Command to Printing Data Buffer

2.4.1. General

```
int     ControlCommand(BYTE *data, int length)
void    CutPaper(int mode)
void    FeedToMark()
void    InitLineSpace()
void    InitPrinterStatus()
void    PrintData()
void    PrintDotFeed(int dots)
void    PrintLineFeed(int lines)
int     PrintSpoolForTTF(TCHAR* data, BYTE fontWidth, BYTE fontHeight)
void    SetAbsPosition(int distance)
void    SetCharCodeTable(int n, int mcu)
void    SetCharSpace(int n)
void    SetFontForTTF(TCHAR *ttfFile)
```

```
void      SetFontSize(int n)
void      SetLineSpace(int n)
void      SetPositionFromMark(int distance)
void      SetTextAlignment(int n)
void      SetTextStyle(int underline, BOOL bold, int width, int height, BOOL reverse)
void      SetUpsideDown(BOOL set)
void      TextSaveSpool(TCHAR *text)
```

2.4.2. Page Mode

```
void      InitPageMode(int x, int y, int width, int height)
void      Page_ClearCurrentData()
void      Page_DrawBox(int x, int y, int width, int height, int thickness)
void      Page_DrawEllipse(int x, int y, int radiusW, int radiusH, int thickness)
void      Page_DrawLine(int x1, int y1, int x2, int y2, int thickness)
void      Page_DotFeed(int dots)
void      Page_LineFeed(int lines)
void      Page_Newline()
void      Page_Print()
void      Page_Print_StandardMode()
void      Page_SetArea(int x, int y, int width, int height)
void      Page_SetDirection(int n)
void      Page_SetPosition(int x, int y)
void      Page_SetStandardMode()
void      SetPageMode()
```

2.4.3. Image

```
void      CompressedBmpSaveSpool(TCHAR* bmpFilePath)
void      LoadLogoSaveSpool(int n)
void      NormalBmpSaveSpool(TCHAR* bmpFilePath)
```

2.4.4. Barcode

```
void      DataMatrixSaveSpool(int width, int height, int module, TCHAR *barcodeData)
void      GS1DatabarSaveSpool(int type, int n, TCHAR *barcodeData)
void      MaxicodeSaveSpool(int mode, TCHAR *barcodeData)
void      MicroPDF417SaveSpool(int width, int column, int row, int ratio,
          TCHAR *barcodeData, BOOL bHri)
void      OneDimensionBarcodeSaveSpool(BYTE barcodeType, int width, int height,
          BOOL bHri, TCHAR *barcodeData)
void      PDF417SaveSpool(int width, int column, int level, int ratio,
          TCHAR *barcodeData, BOOL bHri)
void      QRCodeSaveSpool(int version, TCHAR level, int module, TCHAR *barcodeData)
```

```
void        TruncatedPDF417SaveSpool(int width, int column, int level, int ratio,
            TCHAR *barcodeData, BOOL bHri)
```

2.5. Deprecated APIs

```
int         BarcodeSaveSpool(BYTE barcodeType, BYTE width, BYTE height, BOOL bHri,
            TCHAR* barcodeData, int prtWidth, int prtHeight, int xPos)

int         BmpSaveSpool(TCHAR* bmpFilePath, int x, int y)

void        CardCancel()

void        CardRead(BOOL bTrack2 = TRUE)

void        CardRead_M(BOOL bTrack2 = TRUE)

BOOL        ClosePrinter()

int         InitPrinter(TCHAR *portName, int baudRate = 57600, BOOL bProtocol = TRUE)

int         InitWlanPrinter(TCHAR *ipAddress, int port, int timeoutMsec = 5000,
            BOOL bProtocol = TRUE)

int         SaveSpool(TCHAR *text, int extension = 0, BOOL bold = FALSE)

void        SetCharCodeTableEX(int n)
```

3. API Reference

3.1. Device Connection Control

Connect to Woosim printers or terminate the connection. Serial and Wi-Fi connections are supported.

```
int ConnectSerialPrinter(TCHAR *portName, int baudRate = 57600, int timeoutMsec = 1000,
                         BOOL bProtocol = FALSE)
```

Connect to Woosim printers using serial connection

Parameters

pPortName	Serial port connected to the printer (COM1, COM2, etc.)
baudRate	Serial communication speed of the connected printer. You can check it through the printer Self-Test
timeoutMsec	Maximum time to try to connect (msec)
bProtocol	Keeps FALSE in general case

Returns

SUCCESS (1)	Connection success
ALREADY_OPENED (-1)	The printer was connected already
UNABLE_TO_OPEN_THE_PORT (-2)	The port is disabled
UNABLE_TO_CONFIGURE_THE_SERIAL_PORT (-3)	The port initializing failed
UNABLE_TO_SET_THE_TIMEOUT_PARAMETERS (-4)	Time-out setting failed
TIMEOUT (-7)	The connection failed within valid time

```
int ConnectWirelessPrinter(TCHAR *ipAddress, int port, int timeoutMsec = 5000,
                           BOOL bProtocol = FALSE)
```

Connect to Woosim printers using Wi-Fi connection

Parameters

pIPAddress	IP address of the target printer
port	Printer port number
timeoutMsec	Maximum time to try to connect (msec)
bProtocol	Keeps FALSE in general case

Returns

SUCCESS (1)	Connection success
ALREADY_OPENED (-1)	The printer was connected already
SOCKET_ERROR (-2)	Socket initialization failed

CONNECT_FAIL (-3)	Connection failure
TIMEOUT (-7)	The connection failed within valid time

```
BOOL ClosePrinterConnection()
```

Terminate the connection with the printer.

Returns

Always TRUE

3.2. Direct Transfer to Device

The APIs introduced here send commands to the printer as soon as they are called without using the Printing Data Buffer.

```
void EnterMSRMode(int n)
```

Enter the MSR mode

The magnetic card consists of three tracks, and the tracks can be read differently depending on the MSR: 12 Track, 23 Track, and 123 Track. The track read depends on the selected mode.

n	12 Track MSR	23 Track MSR	123 Track MSR
0	1 Track	2 Track	1 Track
1	2 Track	3 Track	2 Track
2	1,2 Track	2,3 Track	1,2 Track
3	N.A	N.A	1,2,3 Track
4	N.A	N.A	3 Track

If the MSR read successfully, MSR mode is automatically terminated.

For more information on the data transmitted through reading, refer to the *Magnetic Card Data Output Format* area of *Woosim Command manual*.

Parameters

n Select card track (0 ~ 4)

```
void CancelMSRMode()
```

Exit the MSR mode.

```
void EnterSCRMode()
```

Enter the SCR mode

```
void CancelSCRMode()
```

Exit the SCR mode.

```
void GetPrinterModelName()
```

Inquire device model name of the connected printer.

The response data from device is firmware dependent. For example, **R240(RX)_**.

```
void GetFirmwareVersion()
```

Inquire device version of the connected printer.

The response data from device is firmware dependent. For example, **[Ver 2.0 2018/10/31]**.

```
int  GetPrinterStatus(int timeoutMsec)
```

Inquire status of the connected printer.

The one byte response data from device includes condition information by paper sensor, cover sensor, mark sensor and so on. See the <ESC v> command explanation in the *Woosim command manual* for details because the value varies with each printer model.

Parameters

timeoutMsec	Maximum time to wait response (msec)

Returns

`SUCCESS (1)`	Success to get printer status
`TIMEOUT (-7)`	The inquiry failed within valid time
`NOT_OPEN_THE_PORT (-11)`	Printer connection error

```
int  GetPrinterStatusEx(int timeoutMsec)
```

Inquire status and battery status of the connected printer.

The one byte response data from device includes battery capacity level in high 4 bits and printer status in low 4 bits. The most significant bit of response byte is always 1.

Battery Voltage	High 4 bits
7.4 V	1 0 0 0
7.5 V	1 0 0 1
7.8 V	1 0 1 0
8.2 V	1 0 1 1

The printer status includes condition information by paper sensor, cover sensor, mark sensor and so on. See the <ESC v> command explanation in the *Woosim command manual* for details because the value varies with each printer model.

Parameters

timeoutMsec	Maximum time to wait response (msec)

Returns

`SUCCESS (1)`	Success to get printer status
`TIMEOUT (-7)`	The inquiry failed within valid time

NOT_OPEN_THE_PORT (-11)	Printer connection error

3.3. Printing Data Buffer Handling

```
int  PrintSpool(BOOL bDelete = TRUE)
```

Send the Printing Data Buffer content to the printer.

Parameters

bDelete	TRUE : Delete the Printing Data Buffer content FALSE : Remain the Printing Data Buffer content

Returns

SUCCESS (1)	Data transfer success
NOT_OPEN_THE_PORT (-11)	Printer connection error

```
void ClearSpool()
```

Delete the Printing Data Buffer content

3.4. Append Command to Printing Data Buffer

Save commands and data sequentially in the Printing Data Buffer. The stored content can be sent to the printer by the `PrintSpool()` or deleted by the `ClearSpool()`.

3.4.1. General

Generally used in standard mode, but some also work in page mode. However, there are cases in which it operates differently in standard mode and page mode.

```
int  ControlCommand(BYTE *data, int length)
```

Add byte stream content to the Printing Data Buffer.

Parameters

data	Byte stream to be added to the Printing Data Buffer
length	Byte stream length

Returns

The content size stored in the Printing Data Buffer after adding the byte stream.

```
void TextSaveSpool(TCHAR *text)
```

Add text content to the Printing Data Buffer.

Parameters

text	Text to be added to the Printing Data Buffer

```
void InitPrinterStatus()
```

A command to initialize the printer settings and delete data in the printer buffer is added to the Printing Data Buffer.

```
void PrintData()
```

Add a command to print data in the Standard mode to the Printing Data Buffer. In the Page mode, it works as a line break.

```
void PrintDotFeed(int dots)
```

Add a command to print data and feed paper in the Standard mode to the Printing Data Buffer.

Parameters

dots	The paper feed length in dot unit (0 ~ 255)

```
void PrintLineFeed(int lines)
```

Add a command to print data and feed paper in the Standard mode to the Printing Data Buffer.

Parameters

lines	The paper feed length in line unit (0 ~ 255)

```
void SetCharCodeTable(int n, int mcu)
```

Add a command to set a character code table to the Printing Data Buffer.

The code table that can be selected differs depending on the MCU.

Parameters

n	Code table (refer to the below table)
mcu	Printer MCU ID (M16C: 0, ARM: 1, RX: 2)

M16C, ARM MCU

n	Character Code Table
0	CP437 (USA, Standard Europe)
1	Katakana
2	Multilingual CP850
3	Portuguese CP860
4	ISO8859-15 (Latin9)
5	Polish
255	DBCS (Double Byte Character System) One of KSC5601, Shift-JIS, BIG5, GB2312

RX MCU

n	Character Code Table
0	USA, Standard Europe [CP437]
1	Katakana

2	Multilingual(Latin-1) [CP850]
3	Portuguese [CP860]
4	Canadian-French [CP863]
5	Nordic [CP865]
6	Slavic(Latin-2) [CP852]
7	Turkish [CP857]
8	Greek [CP737]
9	Russian(Cyrillic) [CP866]
10	Hebrew [CP862]
11	Baltic [CP775]
12	Polish
13	Latin-9 [ISO8859-15]
14	Latin1[Win1252]
15	Multilingual Latin I + Euro[CP858]
16	Russian(Cyrillic)[CP855]
17	Russian(Cyrillic)[Win1251]
18	Central Europe[Win1250]
19	Greek[Win1253]
20	Turkish[Win1254]
21	Hebrew[Win1255]
22	Vietnam[Win1258]
23	Baltic[Win1257]
24	Azerbaijani
30	Thai[CP874]
40	Arabic [CP720]
41	Arabic [Win 1256]
42	Arabic (Farsi)
43	Arabic presentation forms B
50	Hindi Devanagari
255	DBCS (Double Byte Character System) One of KSC5601, Shift-JIS, BIG5, GB18030

```
void SetFontSize(int n)
```

Add a command to set font size to the Printing Data Buffer.

The font size that can be selected differs depending on the MCU and code table. Refer to the below table.

n	RX	M16C, ARM
0	12x24	12x24

1	9x24	9x24
2	8x16	N.A

The Thai font only supports 12x24.

The Arabic and Hindi font only support 16x24.

DBCS fonts only support 24x24.

Parameters

n Font size (0 ~ 2)

```
void SetTextAlignment(int n)
```

Add a command to set alignment to the Printing Data Buffer.

Parameters

n The alignment type (Left: 0, Center: 1, Right: 2)

```
void SetTextStyle(int underline, BOOL bold, int width, int height, BOOL reverse)
```

Add a command to set character attributes to the Printing Data Buffer.

Parameters

underline Underline thickness (0 ~ 2)

bold If TRUE, following text will print bold style

width The number of times extension on width (0 ~ 7)

height The number of times extension on height (0 ~ 7)

reverse If TRUE, following text will print reverse style

```
void InitLineSpace()
```

Add a command to initialize line spacing to the Printing Data Buffer.

The initial value is 30 dots.

```
void SetLineSpace(int n)
```

Add a command to set line spacing to the Printing Data Buffer.

Parameters

n Line spacing in dot unit (0 ~ 255)

```
void SetAbsPosition(int distance)
```

Add a command to move printing position from the beginning of the line to the Printing Data Buffer.

Parameters

distance The distance from the beginning of the line in dot unit

```
void SetCharSpace(int n)
```

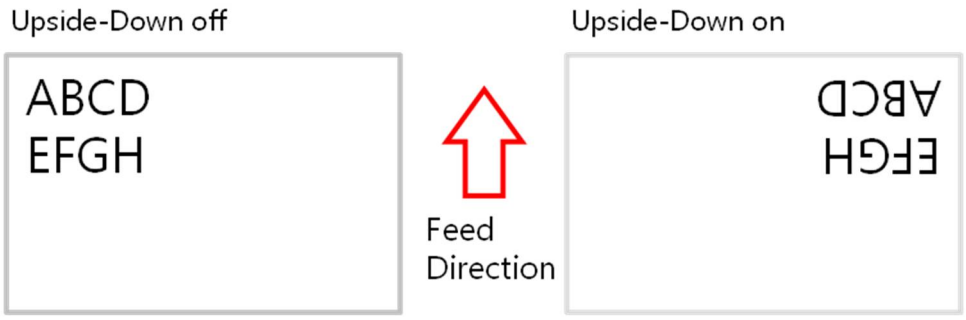
Add a command to set right-side character spacing to the Printing Data Buffer.

Parameters

n	Character spacing in dot unit (0 ~ 255)

```
void SetUpsideDown(BOOL set)
```

Add a command to turn on or off upside down printing to the Printing Data Buffer.



Parameters

set	If TRUE, following text will print upside-down style

```
void SetFontForTTF(TCHAR *ttfFile)
```

Add a command to select a TTF file to the Printing Data Buffer.

The assigned TTF file should be saved in the printer in advance. The length of the TTF file name should be less than 30 letters in English.

To save additional TTF files to the printer, *Woosim Downlaoder* program is required. Please contact Woosim Sales Department.

Parameters

ttfFile	The TTF file name

```
int  PrintSpoolForTTF(TCHAR* data, BYTE fontWidth, BYTE fontHeight)
```

Add a command to print TTF text to the Printing Data Buffer.

The TTF file should be selected by `SetFontForTTF()` in advance.

Parameters

data	The string to be printed
fontWidth	The TTF width size in point unit (4 ~ 255)
fontHeight	The TTF height size in point unit (4 ~ 255)

Returns

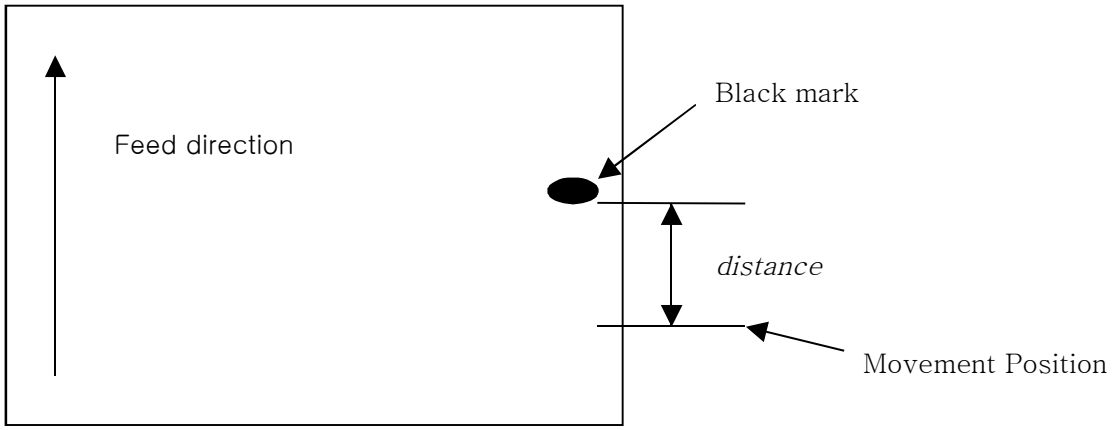
1 on success, 0 on failure

```
void SetPositionFromMark(int distance)
```

A command to save the movement position from the mark sensing to the printer flash memory is added to the Printing Data Buffer.

After the movement position is set to the printer, paper will feed the amount of assigned distance from the mark sensing position whenever call `FeedToMark()` API. This command only needs to be used once when the paper feeding length should be changed.
It is strongly recommended that this will be used in separated setting utility program because it does flash memory writing action.
Parameters

distance The distance from the mark sensing position in dot unit




```
void FeedToMark()
```

Add a command to feed paper to the mark sensing position to the Printing Data Buffer.
In practice, feed paper additionally specified by `SetPositionFromMark()`. It works only when the printer is set to use the Mark sensor.

```
void CutPaper(int mode)
```

Add a command to cut paper to the Printing Data Buffer.
It works only on printers equipped with an auto-cutter. You can select full cut or partial cut.
Parameters

mode Full-cut: 0, Partial-cut: 1

3.4.2. Page Mode

```
void SetPageMode()
```

Add a command to change mode from the Standard mode to the Page mode to the Printing Data Buffer.

```
void InitPageMode(int x, int y, int width, int height)
```

A command to change mode from the Standard mode to the Page mode and set a writing area is added to the Printing Data Buffer.
Parameters

x The horizontal starting position of writing area in dot unit

y The vertical starting position of writing area in dot unit

width	The writing area width in dot unit. It should be bigger than 0
height	The writing area height in dot unit (maximum 2400)

```
void Page_DrawLine(int x1, int y1, int x2, int y2, int thickness)
```

Add a command to draw a line to the Printing Data Buffer.

Parameters

x1	The x-coordinate of start position
y1	The y-coordinate of start position
x2	The x-coordinate of end position
y2	The y-coordinate of end position
thickness	The thickness of line in dot unit

```
void Page_DrawBox(int x, int y, int width, int height, int thickness)
```

Add a command to draw a box to the Printing Data Buffer.

Parameters

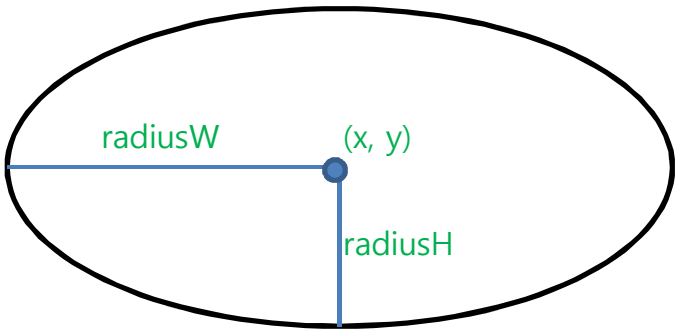
x	The x-coordinate of upper-left corner of box
y	The y-coordinate of upper-left corner of box
width	The box width in dot unit
height	The box height in dot unit
thickness	The thickness of line in dot unit


```
void Page_DrawEllipse(int x, int y, int radiusW, int radiusH, int thickness)
```

Add a command to draw an ellipse to the Printing Data Buffer.

Parameters

x	The x-coordinate of central point
y	The y-coordinate of central point
radiusW	The radius of horizontal axis
radiusH	The radius of vertical axis
thickness	The thickness of line in dot unit




```
void Page_DotFeed(int dots)
```

Add a command to move position vertically in dot unit to the Printing Data Buffer.
The new writing position is the left starting point of the moved position.
Parameters

dots	The movement length in dot unit (0 ~ 255)

```
void Page_LineFeed(int lines)
```

Add a command to move position vertically in line unit to the Printing Data Buffer.
The new writing position is the left starting point of the moved line.
Parameters

lines	The movement length in line unit (0 ~ 255)

```
void Page_Newline()
```

Add a command to move position to the next line to the Printing Data Buffer.
The new writing position is the left starting point of the next line.

```
void Page_ClearCurrentData()
```

Add a command to delete content in the writing area to the Printing Data Buffer.

```
void Page_SetArea(int x, int y, int width, int height)
```

Add a command to set a writing area to the Printing Data Buffer.
Parameters

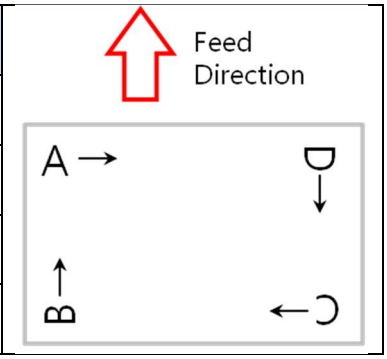
x	The horizontal starting position of writing area in dot unit
y	The vertical starting position of writing area in dot unit
width	The writing area width in dot unit. It should be bigger than 0
height	The writing area height in dot unit (maximum 2400)

```
void Page_SetDirection(int n)
```

Add a command to set the printing direction and start position to the Printing Data Buffer.
Parameters

n	The symbolic value to specify printing direction and start position (0 ~ 3)

n	starting position	printing direction
0	upper left (A)	left → right
1	lower left (B)	bottom → up
2	lower right (C)	right → left
3	upper right (D)	top → bottom




```
void Page_SetPosition(int x, int y)
```

Add a command to set writing position to the Printing Data Buffer.

Parameters

x The horizontal starting position to write in dot unit

y The vertical starting position to write in dot unit

```
void Page_Print()
```

Add a command to print content in writing areas to the Printing Data Buffer.

The contents written in the Page mode are not deleted after printing. It can be deleted by `Page_ClearCurrentData()`.

```
void Page_SetStandardMode()
```

Add a command to change mode from the Page mode to the Standard mode to the Printing Data Buffer.

```
void Page_Print_StandardMode()
```

Add a command to print content in writing areas and change mode from the Page mode to the Standard mode to the Printing Data Buffer.

3.4.3. Image

```
void NormalBmpSaveSpool(TCHAR* bmpFilePath)
```

Add a command to print image file to the Printing Data Buffer.

1 bit per pixel or 24 bits per pixel BMP format is available.

Parameters

bmpFilePath The file path for printing

```
void CompressedBmpSaveSpool(TCHAR* bmpFilePath)
```

Add a command to print image file to the Printing Data Buffer.

1 bit per pixel or 24 bits per pixel BMP format is available. Compress the image data to reduce the transmitted size.

Parameters

bmpFilePath The file path for printing

```
void LoadLogoSaveSpool(int n)
```

Add a command to print image that is downloaded in printer to the Printing Data Buffer.

The image should be saved in the printer in advance. Refer to the <ESC f> command explanation in the *Woosim Command Mmanual* for details.

To save image files to the printer, *Woosim Downlaoder* program is required. Please contact Woosim Sales Department.

Parameters

n	The index of image stored in printer device. The maximum count of images that can be stored in device is dependent on MCU type

3.4.4. Barcode

```
void OneDimensionBarcodeSaveSpool(BYTE barcodeType, int width, int height, BOOL bHri,
                                  TCHAR *barcodeData)
```

Add a command to print a barcode to the Printing Data Buffer.

Parameters

barcodeType	The barcode type (65 ~ 73) Refer to the *WOOSIM_PRINTER_LIB.h* file #define statement
width	The barcode data width in dot unit (2 ~ 8)
height	The barcode height in dot unit (0 ~ 255)
bHri	If TRUE, data values are printed at the bottom of barcode
barcodeData	The barcode source data. Data length and value is dependent on barcode type.

Type #	Barcode System	Number of characters	Remarks
65	UPC-A	11 ≤ n ≤ 12	48 ≤ d ≤ 57
66	UPC-E	11 ≤ n ≤ 12	48 ≤ d ≤ 57
67	EAN13	11 ≤ n ≤ 13	48 ≤ d ≤ 57
68	EAN8	7 ≤ n ≤ 8	48 ≤ d ≤ 57
69	CODE39	1 ≤ n ≤ 255	48 ≤ d ≤ 57 65 ≤ d ≤ 90 d = 32, 36, 37, 43, 45, 46,47
70	ITF	1 ≤ n ≤ 255 (even number)	48 ≤ d ≤ 57
71	CODABAR	1 ≤ n ≤ 255	48 ≤ d ≤ 57 65 ≤ d ≤ 68 d = 36, 43, 45, 46,

			47, 58
72	CODE93	1 ≤ n ≤ 255	0 ≤ d ≤ 127
73	CODE128	2 ≤ n ≤ 255	0 ≤ d ≤ 127 d=C1H (FNC1) d=C2H (FNC2) d=C3H (FNC3) d=C4H (FNC4)

```
void PDF417SaveSpool(int width, int column, int level, int ratio, TCHAR *barcodeData,
                     BOOL bHri)
```

Add a command to print a PDF417 barcode to the Printing Data Buffer.

Parameters

width The barcode data width (1 ~ 8)

column The column number (1 ~ 30)

level The security level to restore when barcode image is damaged (0 ~ 8)

ratio The horizontal and vertical ratio (2 ~ 5)

barcodeData The barcode source data

bHri The HRI character printing option

```
void DataMatrixSaveSpool(int height, int width, int module, TCHAR *barcodeData)
```

Add a command to print a Data Matrix barcode to the Printing Data Buffer.

Parameters

height The height of the symbol (0 : auto size)

width The width of the symbol (0 : auto size)

module The module size (1 ~ 8)

barcodeData The barcode source data

Symbol size		**Capacity (bytes)**			***ECC(%)**	**Remark**
Row	Column	Numeric	Alpha-numeric	Byte (8bit)		
10	10	6	3	3	62.5	
12	12	10	6	5	58.3	
8	18	10	6	5	58.3	Rectangular
14	14	16	9	8	55.6	
8	32	20	12	10	52.4	Rectangular
16	16	24	15	12	50.0	
12	26	32	21	16	46.7	Rectangular

18	18	36	24	18	43.8	
20	20	44	30	22	45.0	
12	36	44	30	22	45.0	Rectangular
22	22	60	24	30	40.0	
16	36	34	45	32	42.9	Rectangular
24	24	72	51	36	40.0	
26	26	88	63	44	38.9	
16	48	98	72	49	36.4	Rectangular
32	32	124	90	62	36.7	
36	36	172	126	86	32.8	
40	40	228	168	114	29.6	
44	44	288	213	144	28.0	
48	48	348	258	174	28.1	
52	52	408	303	204	29.2	
64	64	560	417	280	28.6	
72	72	736	549	368	28.1	
80	80	912	681	456	29.6	
88	88	1152	861	576	28.0	
96	96	1392	1041	696	28.1	
104	104	1632	1221	816	29.2	
120	120	2100	1572	1050	28.0	
132	132	2608	1953	1304	27.6	
144	144	3116	2334	1558	28.5	

* ECC: Error Correction Code rate

```
void QRCodeSaveSpool(int version, char level, int module, TCHAR *barcodeData)
```

Add a command to print a QR-Code to the Printing Data Buffer.

Parameters

version The size of the symbol (1 ~ 40, 0 : auto size)

level The EC level (L: 7%, M: 15%, Q: 25%, H: 30%)

module The module size (1 ~ 8)

barcodeData The barcode source data

Version	Capacity (Code words) by EC level			
	L (7%)	M (15%)	Q (25%)	H (30%)
1	19	16	13	9

2	34	28	22	16
3	55	44	34	26
4	80	64	48	36
5	108	86	62	46
6	136	108	76	60
7	156	124	88	66
8	194	154	110	86
9	232	182	132	100
10	274	216	154	122
11	324	254	180	140
12	370	290	206	158
13	428	334	244	180
14	461	365	261	197
15	523	415	295	223
16	589	453	325	253
17	647	507	367	283
18	721	563	397	313
19	795	627	445	341
20	861	669	485	385
21	932	714	512	406
22	1006	782	568	442
23	1094	860	614	464
24	1174	914	664	514
25	1276	1000	718	538
26	1370	1062	754	596
27	1468	1128	808	628
28	1531	1193	871	661
29	1631	1267	911	701
30	1735	1373	985	745
31	1843	1455	1033	793
32	1955	1541	1115	845
33	2071	1631	1171	901
34	2191	1725	1231	961
35	2306	1812	1286	986
36	2434	1914	1354	1054

37	2566	1992	1426	1096
38	2702	2102	1502	1142
39	2812	2216	1582	1222
40	2956	2334	1666	1276

```
void MicroPDF417SaveSpool(int width, int column, int row, int ratio,
                          TCHAR *barcodeData, BOOL bHri)
```

Add a command to print a Micro PDF417 barcode to the Printing Data Buffer.

Parameters

width The barcode data width (1 ~ 8)

column The column number (1 ~ 30)

row The row number of the barcode (4 ~ 44, 0 : auto size)

ratio The horizontal and vertical ratio (2 ~ 5)

barcodeData The barcode source data

bHri The HRI character printing option

Columns	Rows	Max Data Bytes	Max Alpha Characters	Max Digits
1	11	3	6	8
1	14	7	12	17
1	17	10	18	26
1	20	13	22	32
1	24	18	30	44
1	28	22	38	55
2	8	8	14	20
2	11	14	24	35
2	14	21	36	52
2	17	27	46	67
2	40	33	56	82
2	46	38	64	93
2	52	43	72	105
3	6	6	10	14
3	8	10	18	26
3	10	15	26	38
3	12	20	34	49
3	15	27	46	67

3	20	39	66	96
3	26	54	90	132
3	32	68	114	167
3	38	82	138	202
3	44	97	162	237
4	4	8	14	20
4	6	13	22	32
4	8	20	34	49
4	10	27	46	67
4	12	34	58	85
4	15	45	76	111
4	20	63	106	155
4	26	85	142	208
4	32	106	178	261
4	38	128	214	313
4	44	150	250	366

```
void TruncatedPDF417SaveSpool(int width, int column, int level, int ratio,
                              TCHAR *barcodeData, BOOL bHri)
```

Add a command to print a Truncated PDF417 barcode to the Printing Data Buffer.

Parameters

width The barcode data width (1 ~ 8)

column The column number (1 ~ 4)

level The security level to restore when barcode image is damaged (0 ~ 8)

ratio The horizontal and vertical ratio (2 ~ 5)

barcodeData The barcode source data

bHri The HRI character printing option

```
void MaxicodeSaveSpool(int mode, TCHAR *barcodeData)
```

Add a command to print a Maxicode to the Printing Data Buffer.

Parameters

mode The mode of the barcode (2 ~ 6)

barcodeData The barcode source data

```
void GS1DatabarSaveSpool(int type, int n, TCHAR *barcodeData)
```

Add a command to print a GS1 Databar to the Printing Data Buffer.

Parameters

type	The GS1 databar type (0 ~ 6) 0: GS1 Databar Omnidirectional 1: GS1 Databar Truncated 2: GS1 Databar Stacked 3: GS1 Databar Stacked Omnidirectional 4: GS1 Databar Limited 5: GS1 Databar Expanded 6: GS1 Databar Expanded Stacked
n	The segment per row that should be even number in range of 2~20. Only for the type 6(GS1 Databar Expanded Stacked).
barcodeData	The GS1 databar source data. Type 0~4: This field should be digits less than 14 Type 5~6: this field should comply with the data standard of the GS1 General Specifications. Use '[' and ']' instead of '(' and ')'.

4. Sample Projects

The Woosim Windows SDK includes example projects that you can refer to for using the Library. All example projects used Windows Mobile 6 Professional platform SDK.
The example projects were written in Visual C++ and Visual C# respectively. The Visual C++ project was developed based on MFC, and the Visual C# project was developed based on .NET Compact Framework 3.5.

All of the example projects were created with the same UI and include the following features:

- Serial and Wi-Fi connection
- Text, image, and barcode printing
- Page mode printing
- Magnetic card reading with MSR

The sample project folder also contains executable file. To operate a Woosim printer with the executable file, you need to check the printer configuration through Self-Test.

The data sent from a printer is received by the Library, and the Library sends a Windows Message to application programs to notify it. Application programs use `RegisterWindowMessage()` to receive the data sent from the printer. At this time, the string defined in the *WOOSIM_PRINTER_LIB.h* is used as a parameter.

```
UINT UWM_RECEIVE_DATA = RegisterWindowMessage("WOOSIM_PRT_OK");
```

The `UWM_RECEIVE_DATA` message is identified by the Windows message loop in the application program, and the data sent from the printer is received. Please refer to the example project for details.

4.1. C++ Program

When developing a C/C++ program using the Library, copy the following two files included in the SDK to the project source folder.

- WOOSIM_PRINTER_LIB.h
- WoosimPrinter.lib

Copy the DLL file included in the SDK to the same folder as the executable application program.

4.2. C# Program

Since the *WOOSIM_PRINTER_LIB.h* file cannot be directly referenced, the Library APIs are referenced through the `DllImport` syntax.

```
[DllImport("WoosimPrinter.DLL")]
public static extern bool ClosePrinterConnection();
```

Please refer to the "Form1.cs" file in the example project.