[ESC Windows SDK]

[Printer ESC Command Development Manual v1.3]

CONTENTS

Information of the Manual

This SDK manual provides the dll file information for Windows application development.

We continuously promote and update the function and quality of all our products. Any change to the product specification and the manual will be without any further notice.

1.1 Operation System

Windows 2003/XP/7/8/10

1.2 Remark

- When error code Return Value is greater than 0, it is the internal error of Windows system, please refer to related help file.
- This SDK contains two versions of dll files --- Ansi character and Unicode character, please select the dll file accordingly per the development environment.

PrinterCreator

Set up the target printer of specified model (should create target printer before using any function).

int PrinterCreater(
    void* *handle*,
    const TCHAR* *model*
);

Parameter：

void handle*
    [in,out] The created target printer object.

const TCHAR model*
    [in] Specify the model of target printer.

Return Value：

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_NO_ENOUGMEMORY	-2	No enough memory
E_INVALID_MODEL	-8	Invalid model name

PrinterCreatorS

Set up the target printer of specified model, the function is same to PrinterCreator (should create target printer before using any function).

void* PrinterCreaterS(
    const TCHAR* *model*
);

Parameter：

const TCHAR model*
    [in] Specify the model of target printer.

Return：

Success: return the handle of printer object.

Fail: return NULL.

PrinterDestroy

Release the resource of specified model printer that has set up (after operation completed and no more operation for printer, it should release the printer that has set up).

int PrinterDestroy(
    void* *handle*
);

Parameter：

void handle*
    [in] The handle of target printer object which needs to release.

Return Value：

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_BAD_HANDLE	-6	Invalid handle

PortOpen

Open the communication port and connect with the printer. After successfully connected, other functions can be used. If failed connecting, please check the error information. Currently it supports USB, internet, serial interface and LPT.

int PortOpen(
    void* *handle*,
    const TCHAR* *ioSettings*
);

Parameter：

void handle*
    [in,out] The created target printer object.
const TCHAR* ioSettings
    [in] Set up the parameter of communication port that connected to the target printer. Please see as below:
    Configuration List：

Type	Configuration	Description	Sample
USB	**USB***[,Position]*	USB：connect any USB printer of our company USB*[,Position]*：When connecting to multi printers of our company, can specify connecting to one particular USB printer through USB position information (Position parameter)	USB USB,Port_#0004.Hub_#0003
NET	**NET**, *IP Add (IPV4)[,Port]*	Specify the IP add and port of internet printer. If not specifying port, the default port is 9100.	NET,192.168.0.36 NET,192.168.0.36,9100
COM	**COM*n*,**BAUDRATE_*rate*	Specify the number and baud rate of connected serial port .	COM5,BAUDRATE_19200
LPT	**LPT*n***	Specify the number of connected parallel port.	LPT1

Note：[ ] indicates selective parameter

How to check the information of USB printer position (Position parameter):

In Windows device manager, unfold “USB controller”, select “USB print support” device, select “Property” on right click menu, click “Detail Information”.

The value of Property “Position Information” is the position information of USB printer. (Position parameter)

Return Value：

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_OPEN_FAILED	-311	Port open failed

PortClose

This function is to close the communication port and disconnect with the printer.

int PortClose(
    void* *handle*
);

Parameter：

void handle*
    [in,out] The created target printer object.

Return Value：

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_BAD_HANDLE	-6	Invalid handle

PrinterInitialize

Clear the data in the print buffer and reset the printer modes to the modes that were in effect when power was turned on.
Any macro definitions are not cleared.
Offline response selection is not cleared.
Contents of user NV memory are not cleared.
NV graphics (NV bit image) and NV user memory are not cleared.
The maintenance counter value is not effected by this command.
The specifying offline response isn't cleared.

int PrinterInitialize(
    void* *handle*
);

Parameter：

void handle*
    [in,out] The created target printer object.

Return Value：

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

SetTextLineSpace

Set the line spacing to lineSpace × (vertical or horizontal motion unit).
When standard mode is selected, the vertical motion unit is used.
When page mode is selected, the vertical horizontal motion unit is used for the print direction.

int SetTextLineSpace(
    Void* *handle*,
    int *lineSpace*
);

Parameter：

void handle*
    [in,out] The created target printer object.

int lineSpace
    [in] Set the line spacing of characters. 0≤linespace≤255

Return Value：

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

CancelPrintDataInPageMode

In page mode, delete all the print data in the current print area.

int CancelPrintDataInPageMode(
    void* *handle*
);

Parameter：

void handle*
    [in,out] The created target printer object.

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

GetPrinterState

This function is for getting the printer real-time state.

int GetPrinterState(
    void* *handle,*
    unsigned int* *printerStatus*
);

Parameter：

void handle*
    [in,out] The created target printer object.

unsigned int printerStatus*
    [in,out] Pinter real-time state, when returning to multi-state, value is showed as accumulation, return state please refer to below:

Error Code	Value	Description
STS_Normal	0	Normal
STS_PAPEREMPTY	1	Paper out
STS_COVEROPEN	2	Upper cover opened
STS_PAPERNEAREND	4	Paper near end
STS_ERROR	32	Error occured when getting state
STS_NOT_OPEN	64	Port not open
STS_OFFLINE	128	Printer off line

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout
E_IO_READ_FAILED	-331	Read failed
E_IO_READ_TIMEOUT	-332	Read timeout

SetCodePage

Select character code table.

int SetCodePage(
    void* *handle,*
    int *characterSet,*
    int *type*
);

Parameter：

void handle*
    [in,out] The created target printer object.

int *characterSet*
    [in] Select code page setting.
        Default: Chinese character set.

Character Set	**Value**	**Description**
CHARACTERSET_DEFAULT	0	Default
CHARACTERSET_USA	437	English character set
CHARACTERSET_MULTILINGUAL	850	Multilingual character set
CHARACTERSET_LATIN2	852	Latin character set
CHARACTERSET_EURO	858	European character set
CHARACTERSET_PORTUGUESE	860	Portuguese character set
CHARACTERSET_CANADIAN_FRE	863	Canadian-French character set
CHARACTERSET_NORDIC	865	Nordic character set
CHARACTERSET_CYRILLIC2	866	Cyrillic character set
CHARACTERSET_WPC1252	1252	WPC character set

int *type*
[in] Select whether to save settings in the printer flash memory.
    0: Not write to the flash memory, settings will not be saved when power is off.
    1: Write to the flash memory, settings will be saved when power is off.

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

SetInternationalCharacter

Select an international character set.

int SetInternationalCharacter(
    void* *handle,*
    int *characterSet*
);

Parameter：

void handle*
    [in,out] The created target printer object.

int *characterSet*
    [in] Select international character setting.
        Default: U.S.A.

Value	Description
0	U.S.A
1	France
2	Germany
3	U.K.
4	Denmark I
5	Sweden
6	Italy
7	Spain
8	Japan
9	Norway
10	Denmark II
11	Spain II
12	Latin America
13	Korean
14	Slovenia / Croatia
15	Chinese

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

CutPaper

Feed paper to (cutting position + distance × vertical motion unit) and execute a full cut (cuts the paper completely) or execute a partial cut (one point left uncut), then feed paper to the print start position.

int CutPaper(
    void* *handle,*
    int *cutMode,*
    int *distance*
);

Parameter：

void handle*
    [in,out] The created target printer object.

int cutMode
    [in] Paper cut mode. Execute a full cut or a partial cut.

Paper Cut Mode	**Value**	**Description**
FULL_CUT	0	Full cut
PARTIAL_CUT	1	Partial cut

int distance
    [in] Specify a range of paper cut. 0≤distance≤255

Return Value：

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

FeedLine

Print the data in the print buffer and feed lines, when printer in page mode, only the print position moves, and the printer does not perform actual printing.

int FeedLine(
    void* *handle,*
    int *lines*
);

Parameter：

void handle*
    [in,out] The created target printer object.

int lines
    [in] Set lines of paper feed. 0≤lines≤255

Return Value：

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

OpenCashDrawer

This function is for opening the cash drawer(printer should be connected with cash drawer).

int OpenCashDrawer(
    void* *handle,*
    int *pinMode,*
    int *onTime,*
    int *offTime*
);

Parameter：

void handle*
    [in,out] The created target printer object.

int cutMode
    [in] Select the pin which cash drawer connected.

Pin	Value	Description
CASDRAWER_1	0	Pin 2
CASDRAWER_2	1	Pin 5

int onTime
    [in] Set the start time of pulse, onTime*2ms.

int offTime
    [in] Set the end time of pulse, offTime*2ms.
    Remark: When the setting value of end time is less than that of start time, end time is equal to the start time.

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

PrintText

This function is for printing printer text with attribute.

int PrintText(
    void* *handle*,
    const TCHAR* *data*,
    int *alignment,*
    int *attribute,*
    int *textSize*
);

Parameter：

void handle*
    [in,out] The created target printer object.

const TCHAR data,*
    [in] The text data which needs to print.

int alignment
    [in] The alignment method of text.

Alignment Method	Value	Description
ALIGNMENT_LEFT	0	Left alignment
ALIGNMENT_CENTER	1	Center alignment
ALIGNMENT_RIGHT	2	Right alignment

int attribute
    [in] Set text font property, property setting can be accumulated.

Text Property	Value	Description
TEXT_NORMAL_MODE	0	Default setting font A
TEXT_FONT_BOLD	2	Set font bold
TEXT_FONT_UNDERLINE_MODE	4	Set font underline mode
TEXT_FONT_REVERSE	8	Set font reverse
TEXT_FONT_DW_DMODE	48	Set font double-width and double-height

int textSize
    [in] Set the text size (It will not be printed if the text length exceeds print paper area).

Set text width:

Text Width	Value	Description
TEXT_SIZE_0WIDTH	0	Text width × 1
TEXT_SIZE_1WIDTH	16	Text width × 2
TEXT_SIZE_2WIDTH	32	Text width × 3
TEXT_SIZE_3WIDTH	48	Text width × 4
TEXT_SIZE_4WIDTH	64	Text width × 5
TEXT_SIZE_5WIDTH	80	Text width × 6
TEXT_SIZE_6WIDTH	96	Text width × 7
TEXT_SIZE_7WIDTH	112	Text width × 8

Set text height:

Text Height	Value	Description
TEXT_SIZE_0HEIGHT	0	Text height × 1
TEXT_SIZE_1HEIGHT	1	Text height × 2
TEXT_SIZE_2HEIGHT	2	Text height × 3
TEXT_SIZE_3HEIGHT	3	Text height × 4
TEXT_SIZE_4HEIGHT	4	Text height × 5
TEXT_SIZE_5HEIGHT	5	Text height × 6
TEXT_SIZE_6HEIGHT	6	Text height × 7
TEXT_SIZE_7HEIGHT	7	Text height × 8

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

PrintTextS

This function is for printing printer text.

int PrintTextS(
  void* *handle*,
  const TCHAR* *data*
);

Parameter:

void handle*
  [in,out] The created target printer object.

const TCHAR data,*
  [in] The text data which needs to print.

Return Value：

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

PrintBarCode

This function is for printing bar code. In standard mode, only when bar code print position is in new line or without data in buffer can print normally. In page mode, when the command of printing bar code is not received, the bar code data will be saved in buffer and print will not be executed.

int PrintBarCode(
void* *handle*,
int *bcType,*
const TCHAR* *bcData,*
int *width,*
int *height,*
int *alignment,*
int *hriPosition*
);

Parameter：

void handle*
[in,out] The created target printer object.

int bcType
[in] Set bar code type.

const TCHAR bcData,*
[in] Bar code data.

Bar Code Type	Value	Bar Code Data Length	Effective Data Range
BARCODE_UPC_A	65	11≤n≤12	48≤data≤57
BARCODE_UPC_E	66	11≤n≤12	48≤data≤57
BARCODE_EAN13 BARCODE_JAN13	67	12≤n≤13	48≤data≤57
BARCODE_EAN8 BARCODE_JAN8	68	7≤n≤8	48≤data≤57
BARCODE_CODE39	69	1≤n≤255	48≤data≤57,65≤data≤90, data=32,36,37,43,45,46,47
BARCODE_ITF	70	1≤n≤255 (even number)	48≤data≤57
BARCODE_CODABAR	71	1≤n≤255	48≤data≤57,65≤data≤68, data =36,43,45,46,47,58
BARCODE_CODE93	72	1≤n≤255	0≤data≤127
BARCODE_CODE128	73	2≤n≤255	0≤data≤127

BARCODE_STANDARD_PDF417	101	2≤n≤928	0≤data≤255
BARCODE_TRUNCATED_PDF417	102	2≤n≤928	0≤data≤255
BARCODE_QRCODE1	103	2≤n≤928	0≤data≤255
BARCODE_QRCODE2	104	2≤n≤928	0≤data≤255

int width

[in] Bar code width effective value range: 2-7, when bar code print width exceeds printable area, bar code print will not be executed. The parameter is ineffective for 2D code.

int height,

[in] Set bar code print height. Effective range:1-255, this parameter is ineffective for 2D code.

int alignment,

[in] Set bar code alignment method.

Alignment Method	**Value**	**Description**
ALIGNMENT_LEFT	0	Left alignment
ALIGNMENT_CENTER	1	Center alignment
ALIGNMENT_RIGHT	2	Right alignment

int hriPosition

[in] Set bar code visible character position.

Position	**Value**	**Description**
BRACODE_HRI_NONE	0	Not print visible character
BRACODE_HRI_ABOVE	1	Print visible character above bar code
BRACODE_HRI_BELOW	2	Print visible character below bar code
BRACODE_HRI_BOTH	3	Print visible character above/below bar code

Return Value：

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

PrintSymbol

This function is for printing 2D code.

int PrintSymbol(
void* *handle*,
int *type,*
const TCHAR* *data,*
int *errLevel,*
int *width,*
int *height,*
int *alignment*
);

Parameter：

void handle*
[in,out] The created target printer object.

int type
[in] 2D code type.

Type	Value	Description
BARCODE_STANDARD_PDF417	101	Standard sample PDF417 code
BARCODE_TRUNCATED_PDF417	102	Simple sample PDF417 code
BARCODE_QRCODE1	103	QR Code sample 1
BARCODE_QRCODE2	104	QR Code sample 2

const *TCHAR* data,*
[in] 2D code data.

Data Length	Data Range
1≤n≤7089	0≤data≤255

int errLevel
[in] 2D code setting error correction level.

Error Correction	Value	Code or Error-tolerant Rate
PDF417_ERROR_CORRECTION_LEVEL_0	48	2
PDF417_ERROR_CORRECTION_LEVEL_1	49	4
PDF417_ERROR_CORRECTION_LEVEL_2	50	8
PDF417_ERROR_CORRECTION_LEVEL_3	51	16
PDF417_ERROR_CORRECTION_LEVEL_4	52	32
PDF417_ERROR_CORRECTION_LEVEL_5	53	64
PDF417_ERROR_CORRECTION_LEVEL_6	54	128
PDF417_ERROR_CORRECTION_LEVEL_7	55	256
PDF417_ERROR_CORRECTION_LEVEL_8	56	512
PDF417_ERROR_CORRECTION_LEVEL_L	48	7%
PDF417_ERROR_CORRECTION_LEVEL_M	49	15%
PDF417_ERROR_CORRECTION_LEVEL_Q	50	25%
PDF417_ERROR_CORRECTION_LEVEL_H	51	30%

int width

[in] 2D code width. 0≤n≤255

int height

[in] 2D code height. 0≤n≤255 (This parameter is ineffective for QR Code.)

int alignment,

[in] 2D code alignment method

Alignment Method	Value	Description
ALIGNMENT_LEFT	0	Left alignment
ALIGNMENT_CENTER	1	Center alignment
ALIGNMENT_RIGHT	2	Right alignment

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

PrintTwoQRCode

This function is for printing two QR codes in the same area. The printer should support page mode.

int PrintTwoQRCode(
void* *handle,*
TCHAR* *data1,*
int *data1Len,*
int *width1,*
int *hAlign1,*
int *vAlign1,*
TCHAR* *data2,*
int *data2Len,*
int *width2,*
int *hAlign2,*
int *vAlign2*
);

Parameter：

void handle*
[in,out] The created target printer object.

TCHAR data1*
[in] QR code data 1.

int data1Len
[in] QR code data 1 length.

Data Length	**Data Range**
1≤n≤7089	0≤data≤255

int width1
[in] QR code module 1 width. 0≤n≤255

int hAlign1
[in] QR code 1 horizontal alignment method.
[in] When value of hAligh1 is greater than 2, horizontal position of QR code 1 can be defined by user(cannot over current page width).

Alignment Method	Value	Description
ALIGNMENT_LEFT	0	Left alignment
ALIGNMENT_CENTER	1	Center alignment
ALIGNMENT_RIGHT	2	Right alignment

int vAlign1

[in] QR code 1 vertical alignment method.

[in] When value of vAligh1 is greater than 2, vertical position of QR code 1 can be defined by user(cannot over current height, otherwise QR code cannot be printed).

Alignment Method	Value	Description
ALIGNMENT_TOP	0	Top alignment
ALIGNMENT_CENTER	1	Center alignment
ALIGNMENT_BOTTOM	2	Bottom alignment

TCHAR data2*

[in] QR code data 2.

int data2Len

[in] QR code data 2 length.

Data Length	Data Range
1≤n≤7089	0≤data≤255

int width2

[in] QR code module 2 width. 0≤n≤255

int hAlign2

[in] QR code 2 horizontal alignment.

[in] When value of hAligh1 is greater than 2, horizontal position of QR code 1 can be defined by user(cannot over current page width).

Alignment Method	Value	Description
ALIGNMENT_LEFT	0	Left alignment
ALIGNMENT_CENTER	1	Center alignment
ALIGNMENT_RIGHT	2	Right alignment

int vAlign2

[in] QR code 2 vertical alignment method.

[in] When value of vAligh1 is greater than 2, vertical position of QR code 1 can be defined by user(cannot over current height, otherwise QR code cannot be printed).

Alignment Method	Value	Description
ALIGNMENT_TOP	0	Top alignment
ALIGNMENT_CENTER	1	Center alignment
ALIGNMENT_BOTTOM	2	Bottom alignment

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

PrintImage

Print specified image (support bmp, jpg, gif, etc.). In page mode, the bit images is only stored in the print buffer and is not printed.

int PrintImage(
  void* *handle,*
  const TCHAR* *filePath,*
  int *scaleMode*
);

Parameter：

void handle*
  [in,out] The created target printer object.
const TCHAR filePath*
  [in] The complete path of image.
int scaleMode
  [in] The scale mode for printing image.

Mode	Value	Description
PRINT_IMAGE_NORMAL	0	Normal mode
PRINT_IMAGE_DOUBLE_WIDTH	1	Double-width mode
PRINT_IMAGE_DOUBLE_HEIGHT	2	Double-height mode
PRINT_IMAGE_QUADRUPLE	3	Quadruple mode

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_NOT_ENOUGMEMORY	-9	Not enough memory
E_IMAGE_BAD_SIZE	-25	Image size error
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

PrintBitMapData

Prints a raster bit image by specified the bit image data (raster format).

int PrintBitMapData(
  void* *handle,*
  int *scaleMode,*
  int *width,*
  int *height,*
  unsigned char* *data*
);

Parameter：

void handle*
  [in,out] The created target printer object.

int scaleMode
  [in] The scale mode for printing image.

Mode	Value	Description
PRINT_IMAGE_NORMAL	0	Normal mode
PRINT_IMAGE_DOUBLE_WIDTH	1	Double-width mode
PRINT_IMAGE_DOUBLE_HEIGHT	2	Double-height mode
PRINT_IMAGE_QUADRUPLE	3	Quadruple mode

int *width*
  [in] Width specifies n bytes in horizontal direction for the bit image.
    0≤width≤72

int *height*
  [in] Height specifies n dots in vertical direction for the bit image.

unsigned char* *data*
  [in] Data specifies the bit image data (raster format).

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_NOT_ENOUGMEMORY	-9	Not enough memory
E_IMAGE_BAD_SIZE	-25	Image size error
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

DefineNVImageCompatible

Define the NV bit image in the NV graphics area. It is able to download several pictures at the same time. The downloaded pictures are numbered from 1. This function is supported only by some printer models and may not be supported by future models.It is recommended to use NV graphics function <DefineNVImage>.

int DefineNVImageCompatible(
  void* *handle,*
  const TCHAR** *filePathList,*
  int ***i****mageQty*
);

Parameter：

void handle*
  [in,out] The created target printer object.
*const TCHAR**] filePathList*
  [in] The specified image path list.
int imageQty
  [in] The specified image quantity.

Return Value：

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_NOT_ENOUGMEMORY	-9	Not enough memory
E_IMAGE_BAD_SIZE	-25	Image size error
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

PrintNVImageCompatible

Print the NV bit image downloaded by <DefineNVImageCompatible>. This function is supported only by some printer models and may not be supported by future models. It is recommended to use NV graphics function <PrintNVImage>.

int PrintNVImageCompatible(
void* *handle,*
int *imgNo,*
int *scaleMode*
);

Parameter：

void handle*
[in,out] The created target printer object.
int imgNo,
[in] Print specified nth image (the image serial number which undefined in NV buffer area will not be printed). 1 ≤ n ≤ 255
int scaleMode,
[in] The scale mode for printing image.

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

DefineDownloadedImageCompatible

Define the downloaded bit image in the downloaded graphic area. This function is supported only by some printer models and may not be supported by future models. It is recommended to use NV graphics function <DefineDownloadedImage>.

int DefineDownloadedImageCompatible(
  void* *handle,*
  const TCHAR* *filePath*
);

Parameter：

void handle*
  [in,out] The created target printer object.
const TCHAR filePath*
  [in] The complete path of image.

Return Value：

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_NOT_ENOUGMEMORY	-9	Not enough memory
E_IMAGE_BAD_SIZE	-25	Image size error
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

PrintDownloadedImageCompatible

Print downloaded bit image.This function is supported only by some printer models and may not be supported by future models. It is recommended to use NV graphics function <PrintDownloadedImage>.

int PrintDownloadedImageCompatible(
void* *handle,*
i**nt** *scaleMode*
);

Parameter：

void handle*
[in,out] The created target printer object.
int scalemode
[in] The scale mode for printing image.

Mode	Value	Description
PRINT_IMAGE_NORMAL	0	Normal mode
PRINT_IMAGE_DOUBLE_WIDTH	1	Double-width mode
PRINT_IMAGE_DOUBLE_HEIGHT	2	Double-height mode
PRINT_IMAGE_QUADRUPLE	3	Quadruple mode

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

GetFirmwareVersion

Get the current printer firmware version.

int GetFirmwareVersion(
    void* *handle,*
    int* *version,*
    int *versionLen*
);

Parameter：

void handle*
    [in,out] The created target printer object.
int version*
    [in,out] Printer firmware version number, e.g. 1.3.12.
int versionLen
    [in] Firmware version data length.

Return Value：

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout
E_IO_READ_FAILED	-331	Read failed
E_IO_READ_TIMEOUT	-332	Read timeout

SelectPageMode

Switch from standard mode to page mode (only effective when printer supports page mode and in standard mode).

int SelectPageMode(
    void* *handle*
);

Parameter：

void handle*
    [in,out] The created target printer object.

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

SelectStandardMode

Switch from page mode to standard mode (only effective in page mode).

int SelectStandardMode(
    void* *handle*
);

Parameter：

void handle*
    [in,out] The created target printer object.

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

SelectPrintDirectionInPageMode

In page mode, select printer print direction. This function is only effective in page mode.

int SelectPrintDirectionInPageMode(
    void* *handle,*
    int *direction*
);

Parameter：

void handle*
    [in,out] The created target printer object.
int direction
    [in] Select print direction.

Print Direction	Value	Description	Start Position
PRINT_DIRECTION_LEFT_TO_RIGHT	0	Left->Right	Top left corner
PRINT_DIRECTION_BOTTOM_TO_TOP	1	Bottom->Top	Bottom left corner
PRINT_DIRECTION_RIGHT_TO_LEFT	2	Right->Left	Bottom right corner
PRINT_DIRECTION_TOP_TO_BOTTOM	3	Top->Bottom	Top right corner

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

SetAbsoluteVerticalPrintPositionInPageMode

In page mode, set the vertical print position (when print start position is top left corner or bottom right corner, is vertical position setting. When print start position is bottom left corner or top right corner, is horizontal setting).

int SetAbsoluteVerticalPrintPositionInPageMode(
    void* *handle,*
    int *position*
);

Parameter：

void handle*
    [in,out] The created target printer object.
int position
    [in] Set vertical position.

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

SetPrintAndReturnStandardMode

Print and return standard mode (only effective in page mode).

int SetPrintAndReturnStandardMode(
    void* *handle,*
);

Parameter：

void handle*
    [in,out] The created target printer object.

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

SetPrintAreaInPageMode

In page mode, set the size and the logical origin of the print area. Both print area width and height cannot be set to 0.

int SetPrintAreaInPageMode(
    void* *handle,*
    int *horizontal,*
    int *vertical,*
    int *width,*
    int *height*
);

Parameter：

void handle*
    [in,out] The created target printer object.
int horizontal
    [in] Set horizontal position of print start.
int vertical
    [in] Set vertical position of print start.
int width
    [in] Set horizontal width of printable area.
int height
    [in] Set vertical height of printable area.
        When print width is 80mm: horizontal star point = 0, vertical start point = 0, width = 576, height = 840.

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

PrintDataInPageMode

Print data in page mode, and not return standard mode after printing (only effective in page mode).

int PrintDataInPageMode(
    void* *handle*
);

Parameter：

void handle*
    [in,out] The created target printer object.

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_MODEL_TYPE	-3	The model type does not support this function
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

DirectIO

User-defined send and read printer data. If function port is not provided, users can send the command data to printer by this interface.

int DirectIO(
    void* *handle,*
    unsigned char*writeData,*
    unsigned int *writeNum,*
    unsigned char* *readData,*
    unsigned int *readNum,*
    unsigned int* *preadedNum*
);

Parameter：

void handle*
    [in,out] The created target printer object.
*unsigned char*writeData*
    [in] Printer write data.
unsigned int writeNum,
    [in] The number of write data. When writNum=0, the operation of write data is not preformed.
unsigned char readData,*
    [in,out] Get the printer return data.
unsigned int readNum,
    [in] Pre-set the number of data that need to read. When readNum=0, the operation of read data is not performed.
unsigned int preadedNum*
    [in,out] The number of actual read data.

Return Value：

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout
E_IO_READ_FAILED	-331	Read failed
E_IO_READ_TIMEOUT	-332	Read timeout

SetAbsolutePrintPosition

Moves the print position to n ×(horizontal or vertical motion unit) from the left edge of the print area.
The printer ignores any setting that exceeds the print area.
When standard mode is selected, the horizontal motion unit is used.
When page mode is selected, the horizontal or vertical motion unit is used for the print direction

int SetAbsolutePrintPosition(
    void* *handle,*
    int *position*
);

Parameter：

void handle*
    [in,out] The created target printer object.
int position
    [in] Horizontal print start position.

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

PositionNextLabel

Print the label and locate the start position of next label.

int PositionNextLabel(
    void* *handle*
);

Parameter：

void handle*
    [in,out] The created target printer object.

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_BAD_HANDLE	-6	Invalid parameter
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

DefineNVImage

Define the NV graphics data (raster format) as a record specified by the key codes (kc1 and kc2) in the NV graphics area.

int DefineNVImage(
    void* *handle,*
    const char* *imagePath,*
    unsigned char *kc1,*
    unsigned char *kc2*
);

Parameter：

void handle*
    [in,out] The created target printer object.
const char imagePath*
    [in] Specify the complete path of image.
unsigned char kc1
    [in] Key code 1. 32 <= kc1 <= 126
unsigned char kc2
    [in] Key code 2. 32 <= kc2 <= 126

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

PrintNVImage

Print the NV graphics data defined by the key codes (kc1 and kc2).

int PrintNVImage(
    void* *handle*,
    unsigned char *kc1,*
    unsigned char *kc2,*
);

Parameter：

void handle*
    [in,out] The created target printer object.
unsigned char kc1
    [in] Key code 1. 32 <= kc1 <= 126
unsigned char kc2
    [in] Key code 2. 32 <= kc2 <= 126

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

DefineDownloadedImage

Defines the downloaded graphics data (raster format) as a record specified by the key codes (kc1 and kc2) in the downloaded graphics area.

int DefineDownloadedImage(
    void* *handle*,
    const char* *imagePath*,
    unsigned char *kc1,*
    unsigned char *kc2*
);

Parameter：

void handle*
    [in,out] The created target printer object.
const char imagePath*
    [in] The path of image.
unsigned char kc1
    [in] Key code 1. 32 <= kc1 <= 126
unsigned char kc2
    [in] Key code 2. 32 <= kc2 <= 126

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

PrintDownloadedImage

Prints the downloaded graphicsdata defined by the key codes (kc1 and kc2).

int PrintDownloadedImage(
    void* *handle*,
    unsigned char *kc1,*
    unsigned char *kc2*
);

Parameter：

void handle*
    [in,out] The created target printer object.
unsigned char kc1
    [in] Key code 1. 32 <= kc1 <= 126
unsigned char kc2
    [in] Key code 2. 32 <= kc2 <= 126

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

DefineBufferedImage

Stores the graphics data (raster format) in the print buffer.

int DefineBufferedImage(
    void* *handle*,
    const char* *imagePath*
);

Parameter：

void handle*
    [in,out] The created target printer object.
const char imagePath*
    [in] Specify the complete path of image.

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

PrintBufferedImage

Prints the buffered graphics data stored by the <DefineBufferedImage>,the printer cannot print when there is no graphics data stored in the print buffer.

int PrintBufferedImage(
  void* *handle*
);

Parameter：

void handle*
  [in,out] The created target printer object.

Return Value：

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

DeleteAllNVImages

Deletes all NV graphics data,deleted areas are designated “Unused Areas”, all key codes are designated as undefined.

int PrintBufferedImage(
    void* *handle*
);

Parameter：

void handle*
    [in,out] The created target printer object.

Return Value：

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

GetCashDrawerState

Get the current cash drawer state.

int GetCashDrawerState(
    void* *handle*,
    int* *drawerState*
);

Parameter：

void handle*
    [in,out] The created target printer object.
int drawerState*
    [in,out] Get the value of cash drawer state.
        0: cash drawer opened;
        1: cash drawer closed or without connection.

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

ClearBuffer

Clear the printer buffer.

int ClearBuffer(
    void* *handle*
);

Parameter：

void handle*
    [in,out] The created target printer object.

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

FormatError

This function is for returning the information of failing to call the function.

int FormatError(
    int *errorNo,*
    int *langid,*
    unsigned char* *buf,*
    int *pos,*
    int *bufSize*
);

Parameter：

int errorNo
    [in] The return error number of function.
int langid
    [in] Language ID currently only supports simplified Chinese and English. Default is 0 (English).
unsigned char buf*
    [in,out] Save the error information.
int pos
    [in] Save the buffer start position.
int bufSize
    [in] Size of buffer.

Return Value：

Error Code	Value	Description
E_SUCCESS	0	Normal
	last-error code	

SetAlign

Set print justification.The justification has no effect in page mode.

int SetAlign(
void* *handle,*
int *align*
);

Parameter:

void handle*
[in,out] The created target printer object.
int align
[in] Set the justification.

align	Justification
0,48	Left
1,49	Center
2,50	Right

Return Value:

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

SetTextBold

Turn emphasized mode on or off.

int SetTextBold(
    void* *handle,*
    int   *bold*
);

Parameter:

void handle*
    [in,out] The created target printer object.
int bold
    [in] Set the emphasized mode for the text.
        0: emphasized mode is turned off.
        1: emphasized mode is turned on.

Return Value:

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

SetTextFont

Set the text font.

int SetTextFont(
    void* *handle,*
    int *font*
);

Parameter:

void handle*
    [in,out] The created target printer object.
int font
    [in] Set the font type.

font	Font
0,48	Font A
1,49	Font B

Return Value:

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

SetBuzzer

Turn buzzer on or off.

int SetBuzzer(
    void* *handle,*
    int *enable*
);

Parameter:

void handle*
    [in,out] The created target printer object.
int enable
    [in] Set the buzzer on or off.
        0: buzzer is turned off.
        1: buzzer is turned on.

Return Value:

Error Code	**Value**	**Description**
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout