ARGOX
Empower the Barcode

Portable Data Collector PT-20 & PA-20 Programming Guide for PC Application

Raw Library Version: 1.0.0.2
Standard Library Version: 1.0.0.5
Support: PT-20 & PA-20

2012/4/10

http://www.argox.com

Table of Content

1 Preface

Its final purpose is to gather the materials well to upload to PC end to do the materials to deal with. Or the materials that must be consulted are downloaded to the materials while doing and gathering the materials of PDC end than to the procedure. Like come say, customer will is it use procedure making one's own systematic procedure to use, develop course they can meet how link , communicate , exchange demand of materials in it, in order to solve above-mentioned problems here , offer Windows relevant component and help the customer to develop the system.

We offer DLL for Windows OS

2 Development Environment

2.1 The Catalogue Explaining

Directory	Sub-Directory	Function Description
Raw		The raw transmission
	Include	The include header file.(*.h)
	Debug	Debug mode library and component
	Release	Release mode library and component
Standard		The standard transmission
	Include	The include file.(*.h)
	Debug	Debug mode library and component
	Release	Release mode library and component

2.2 Developing Instrument

Support Operator System: Microsoft Windows 2000/XP/Vista/7

2.3 Component List

	File Name	Function Description
Raw	Communication.dll	Raw communication master file
	Communication.lib	Raw dll links address file
	ComPort.h	Raw dll head file
	ComErr.h	Raw communication error code file
Standard	Communication.dll	Raw communication master file
	PTProtocol.dll	The communication protocol dll file
	PTProtocol.lib	PTProtocol.dll links address file
	Protocol.h	PTProtocol dll head file
	TransErr.h	Standard communication error code file

3 Material of library

3.1 Channel for connecting

- RS233 & USB:

3.2 Raw Communication

The raw communication without any protocol, user can use this communication to create private protocol.

3.3 Standard Communication

The standard communication with standard protocol, it has regular function to communication with Terminal.

4 Function Description

4.1 Raw Communication

OpenCom_Entry	
Purpose	
	Open communication port without any protocol.Raw data transmission.
Syntax	
	HANDLE OpenCom_Entry(int nPortType, int nPort, LPSTR pName);
Return Value	
	Return the communication handle if function succeeds, or **NULL** if function fail. Use GetError() to get the error code(Reference Appendix B.1), if function fail.
Parameters	
	nPortType(in) 1: Serial, 2:USB **nSelPort (in)** COM Port: 1~255, USB Port: 1~255 (Address) **pName(in)** Reverse
Remarks	

	1. Please use it before other Raw Communication function 2. You can open the second or more port to communicate with other terminal
Example	

	// PORT SERIAL for COM1:
	if(OpenCom_Entry(1, 1, NULL) == NULL)
	return;
	// PORT USB for device number 0 without setting.
	if(OpenCom_Entry(2, 1, NULL) == NULL)
	return;

See Also	
	CloseCom_Entry, CloseAllCom_Entry, SetSerial, EnumUSB, GetUSBNameMaxLen, GetUSBName, GetError
CloseCom_Entry	
Purpose	
	Close the assigned communication port.
Syntax	
	int CloseCom_Entry(HANDLE hPort);
Return Value	
	Return **Zero** if function succeeds, or return a error code. To get the error information, please refer error code table(Appendix B.1)
Parameters	
	hPort(in) Handle to the communication port. The OpenCom_Entry function returns this handle
Remarks	
Example	
	HANDLE hPort = OpenCom_Entry(1, 1, NULL); if(CloseCom_Entry(hPort) != 0) return FAIL;
See Also	
	OpenCom_Entry, CloseAllCom_Entry

CloseAllCom_Entry	
Purpose	
	Close all communication port. If you open multi-port, you can use this function to close them all.
Syntax	
	void CloseAllCom_Entry();
Return Value	
	None
Parameters	
	None
Remarks	
Example	

	HANDLE hPort1, hPort2;
	hPort1 = OpenCom_Entry(1, 1, NULL);
	hPort2 = OpenCom_Entry(2, 1, NULL);
	……
	CloseAllCom_Entry()

See Also	
	OpenCom_Entry, CloseCom_Entry
SetSerial	
Purpose	
	To configure serial port
Syntax	
	void SetSerial(HANDLE hPort, int nBaudrate, int nByteSize, int nParity, int nStopBits, int nFlowCtrl);
Return Value	
	Return **Zero** if function succeeds, or return a error code. To get the error information, please refer error code table(Appendix B.1)
Parameters	
	hPort(in) Handle to the communication port. The OpenCom_Entry function returns this handle **nBaudrate(in)** Baud rate at serial port device.

	1 -> 4800 2 -> 9600 3 -> 19200 4 -> 38400 5 -> 57600 0, 6 -> 115200 **nByteSize(in)** 7 -> 7 bits data 0, 8 -> 8 bits data **nParity(in)** 0 -> None Parity 1 -> ODD Parity 2 -> Even Parity **nStopBits(in)** Reverse **nFlowCtrl(in)** 0 -> None 1 -> Hardware flow control
Remarks	
Example	

	// PORT_SERIAL for COM1: 115200 bps, 8 data bits,
	none parity, Enable Flow Control
	
	HANDLE hPort = OpenCom_Entry(1, 1, NULL);
	
	SetSerial(hPort, 0, 0, 0, 0, 1)
	if(CloseCom_Entry(hPort) != 0)
	return FAIL;

See Also	
	OpenCom_Entry
WriteCom_Entry	
Purpose	
	To write data to assigned channel
Syntax	
	int WriteCom_Entry(HANDLE hPort, int nDataLen, LPTSTR pData);

Return Value	
	Return the data length written to assigned channel if function succeeds, or return Zero if function fails. Use GetError() to get the error code(Reference Appendix B.1), if function fail..
Parameters	
	hPort(in) Handle to the communication port. The OpenCom_Entry function returns this handle **nDataLen(in)** The length of bytes to write **pData(in)** data buffer
Remarks	
Example	

	HANDLE hPort = OpenCom_Entry(1, 1, NULL);
	int nBytesToWrite;
	char acBuf[10] = “Test Write”;
	
	nBytesToWrite = sizeof(acBuf);
	if(WriteCom_Entry(hPort, nBytesToWrite, acBuf) != nBytesToWrite)
	FAIL;

See Also	
	OpenCom_Entry, ReadCom_Entry, SetSerial, PurgeCom, GetError
ReadCom_Entry	
Purpose	
	To read data from assigned channel
Syntax	
	int ReadCom_Entry(HANDLE hPort, LPTSTR pData, int nMaxLength);
Return Value	
	Return the read data length from assigned channel if function success, or **Zero** if function fails. Use GetError() to get the error code(Reference Appendix

	B.1), if function fail.
Parameters	
	hPort(in) Handle to the communication port. The OpenCom_Entry function returns this handle **pData(out)** The receive data buffer **nMaxLength(in)** The size of data buffer
Remarks	
Example	

	HANDLE hPort = OpenCom_Entry(1, 1, NULL);
	int nBufferSize;
	char acBuf[10];
	
	nBufferSize = sizeof(acBuf);
	if(ReadCom_Entry(hPort,acBuf, nBufferSize) == 0)
	FAIL;

See Also	
	OpenCom_Entry, WriteCom_Entry, PurgeCom, GetError, SetReadBuffer, SetReadEvent
PurgeCom	
Purpose	
	To discards all character in assigned port buffer
Syntax	
	int PurgeCom (HANDLE hPort);
Return Value	
	Return **Zero** if function success, or return a error code. To get the error information, please refer error code table(Appendix B.1).
Parameters	
	hPort(in) Handle to the communication port. The OpenCom_Entry function returns this handle
Remarks	

Example	
	HANDLE hPort = OpenCom_Entry(1, 1, NULL); if(PurgeCom(hPort) != 0) FAIL;
See Also	
	OpenCom_Entry, WriteCom_Entry, ReadCom_Entry
SetReadBuffer	
Purpose	
	To change the read buffer size.
Syntax	
	int SetReadBuffer (HANDLE hPort, int nSize);
Return Value	
	Return **Zero** if function success, or return a error code. To get the error information, please refer error code table(Appendix B.1).
Parameters	

	hPort(in)
	  Handle to the communication port. The OpenCom_Entry function returns this handle
	nSize(in)
	  The new setting size.

Remarks	
	When open connect port, there will be generate buffer to receive data. If user do not get data and receive data over the size, the data will be lost. The default buffer size is 4096 bytes. If the setting size is smaller than 4096, it will not be changed, and return error.
Example	
	HANDLE hPort = OpenCom_Entry(1, 1, NULL); if(SetReadBuffer(hPort, 5120) != 0) FAIL;
See Also	
	OpenCom_Entry, ReadCom_Entry
EnumUSB	

Purpose	
	To refresh the USB connect status. When the USB device is plug in or pull out, you should call this function to refresh the connect status.
Syntax	
	int EnumUSB();
Return Value	
	Return the USB port amount.
Parameters	
	None
Remarks	
Example	

	// Open the first USB port
	HANDLE hPort;
	
	If(EnumUSB() > 0)
	{
	  hPort = OpenCom_Entry(2, 1, NULL);
	  ……
	}

See Also	
	OpenCom_Entry, GetUSBName, GetUSBNameMaxLen
GetUSBNameMaxLen	
Purpose	
	To query the USB port name maximum length.
Syntax	
	int GetUSBNameMaxLen();
Return Value	
	Return the maximum length of USB port name.
Parameters	
	None
Remarks	
Example	
	// Get the first USB port name

	HANDLE hPort;
	int nMaxLen;
	char *pbuf;
	
	If(EnumUSB() > 0)
	{
	nMaxLen = GetUSBNameMaxLen();
	pbuf = (char *) new char[nMaxLen];
	GetUSBName(1, pbuf, &nMaxLen);
	……
	}

See Also	
	EnumUSB, GetUSBName
GetUSBName	
Purpose	
	To query the USB port name.
Syntax	
	int GetUSBName(int nPort, LPTSTR pData, int *nBufferMax);
Return Value	
	Return **Zero** if function success, or return error code. To get the error information, please refer error code table(Appendix B.1).
Parameters	
	nPort(in) assign a port to get which USB port name. **pData(out)** The data buffer to receive USB name. ***nBufferMax(in/out)** Pointer to a variable that specifies the buffer size.
Remarks	
	If the nBufferMax is smaller than usb name length, it will be false and *nBufferMax will be the name length.
Example	
	// Get the first USB port name HANDLE hPort; int nMaxLen;

	char *pbuf;
	
	If(EnumUSB() > 0)
	{
	nMaxLen = GetUSBNameMaxLen();
	pbuf = (char *) new char[nMaxLen];
	GetUSBName(1, pbuf, &nMaxLen);
	……
	}

See Also	
	EnumUSB, GetUSBNameMaxLen
SetReadEvent	
Purpose	
	After setting the event, if there are datas in communication port, it will signal the event..
Syntax	
	int SetReadEvent(HANDLE hPort, HANDLE hWatchRead);
Return Value	
	Return **Zero** if function success, or return a error code. To get the error information, please refer error code table(Appendix B.1).
Parameters	
	hPort(in) Handle to the communication port. The OpenCom_Entry function returns this handle. **hWatchRead(in)** Handle to the event object.
Remarks	
Example	

	// Monitor the communication port
	HANDLE hPort = OpenCom_Entry(1, 1, NULL);
	HANDLE hPortEvent;
	
	If(hPort){
	hPortEvent = CreateEvent(NULL, TRUE, FALSE,

	NULL);
	SetReadEvent(hPOrt, hPortEvent);
	……
	}
	
	// Create other thread to monitor the hPortEvent
	void MonitorThread()
	{
	char pBuf[1200];
	
	while(1){
	WaitForSingleObject(hPortEvent, INFINITE);
	ReadCom_Entry(hPort, pBuf,1200);
	……
	}
	……
	}

See Also	
	OpenCom_Entry
GetError	
Purpose	
	To get the error code.
Syntax	
	int GetError();
Return Value	
	Return error code. To get the error code information, please refer error code table(Appendix B.1).
Parameters	
	None
Remarks	
Example	

	// Open the Com1
	HANDLE hPort = OpenCom_Entry(1, 1, NULL);;
	Int nError;
	
	If(hPort == NULL)

	{
	nError = GetError();
	return;
	}

See Also	
	None
GetCommVer	
Purpose	
	To get this DLL edition.
Syntax	
	int GetCommVer(char *pVer);
Return Value	
	Zero.
Parameters	
	pVer(out) Make storing the space of information of the edition.
Remarks	
	The format is x.x.x.x where x is digits. (Like as 1.0.0,1)
Example	
	// Get the DLL version char cVersion[20]; GetCommVer(cVersion);
See Also	
	None

4.2 Standard Communication

PT_OpenPort	
Purpose	
	Open the demand port with standard protocol. Packaged data transmission.
Syntax	
	HANDLE PT_OpenPort(int nPortType, int nPort,

	LPSTR pName, int nBaudrate, int nByteSize, int nParity, int nStopBits, int nFlowCtrl);
Return Value	
	Return the communication handle if function succeeds, or **NULL** if function fail. Use PT_GetErrorCode() to get the error code (Reference Appendix B.2), if function fail.
Parameters	
	nPortType(in) 1: Serial, 2:USB **nPort(in)** COM Port: 1~255, USB Port: 1~255 (Address) **pName(in)** Reverse **nBaudrate(in)** Baud rate at serial port device. 1 -> 4800 2 -> 9600 3 -> 19200 4 -> 38400 5 -> 57600 0, 6 -> 115200 **nByteSize(in)** Reverse **nParity (in)** Reverse **nStopBits (in)** Reverse **nFlowCtrl(in)** Reverse
Remarks	
	1. Please use this function before other standard communication function 2. You can open the second or more port to communicate with other terminal
Example	

	// PORT SERIAL for COM1: 115200
	if(PT_OpenPort(1, 1, NULL, 0, 0, 0, 0, 0) == NULL)

	return;
	// PORT USB for device number 0 without setting.
	if(PT_OpenPort(2, 1, NULL, 0, 0, 0, 0, 0) == NULL)
	return;

See Also	
	PT_ClosePort, PT_CloseAllPort, PT_EnumUSB, PT_GetUSBNameMaxLen, PT_GetUSBName, PT_GetErrorCode
PT_EnumUSB	
Purpose	
	To refresh the USB connect status. When the USB device is plug in or pull out, you should call this function to refresh the connect status.
Syntax	
	int PT_EnumUSB();
Return Value	
	Return the USB port amount.
Parameters	
	None
Remarks	
Example	

	// Open the first USB port
	HANDLE hPort;
	
	If(PT_EnumUSB() > 0)
	{
	hPort = PT_OpenPort(2, 1, NULL, 0, 0, 0, 0, 0);
	……
	}

See Also	
	PT_OpenPort, PT_GetUSBName, PT_GetUSBNameMaxLen
PT_GetUSBNameMaxLen	
Purpose	
	To query the USB port name maximum length.

Syntax	
	int PT_GetUSBNameMaxLen();
Return Value	
	Return the maximum length of USB port name.
Parameters	
	None
Remarks	
Example	

	// Get the first USB port name
	HANDLE hPort;
	int nMaxLen;
	char *pbuf;
	
	If(PT_EnumUSB() > 0)
	{
	  nMaxLen = PT_GetUSBNameMaxLen();
	  pbuf = (char *) new char[nMaxLen];
	  PT_GetUSBName(1, pbuf, &nMaxLen);
	  ……
	}

See Also	
	PT_EnumUSB, PT_GetUSBName
PT_GetUSBName	
Purpose	
	To query the USB port name.
Syntax	
	int PT_GetUSBName(int nPort, LPTSTR pData, int *nBufferMax);
Return Value	
	Return **Zero** if function success, or return error code. To get the error information, please refer error code table(Appendix B.2).
Parameters	
	nPort(in) assign a port to get which USB port name. **pData(out)**

	The data buffer to receive USB name. ***nBufferMax(in/out)** Pointer to a variable that specifies the buffer size.
Remarks	
	If the nBufferMax is smaller than USB name length, it will be false and *nBufferMax will be the name length.
Example	

	// Get the first USB port name
	HANDLE hPort;
	int nMaxLen;
	char *pbuf;
	
	If(PT_EnumUSB() > 0)
	{
	nMaxLen = PT_GetUSBNameMaxLen();
	pbuf = (char *) new char[nMaxLen];
	PT_GetUSBName(1, pbuf, &nMaxLen);
	……
	}

See Also	
	PT_EnumUSB, PT_GetUSBNameMaxLen
PT_ClosePort	
Purpose	
	Close the assigned communication port.
Syntax	
	int PT_ClosePort(HANDLE hPort);
Return Value	
	Return **Zero** if function success, or return error code. To get the error information, please refer error code table(Appendix B.2).
Parameters	
	hPort(in) Handle to the communication port. The PT_OpenPort function returns this handle
Remarks	
Example	

	HANDLE hPort = PT_OpenPort(2, 1, NULL, 0, 0, 0, 0, 0); … PT_ClosePort(hPort);
See Also	
	PT_OpenPort, PT_CloseAllPort
PT_CloseAllPort	
Purpose	
	Close all communication port. If you open multi-port, you can use this function to close them all.
Syntax	
	void PT_CloseAllPort();
Return Value	
	None
Parameters	
	None
Remarks	
Example	

	HANDLE hPort1, hPort2;
	hPort1 = PT_OpenPort(1, 1, NULL, 0, 0, 0, 0, 0);
	hPort2 = PT_OpenPort(2, 1, NULL, 0, 0, 0, 0, 0);
	……
	PT_CloseAllPort()

See Also	
	PT_OpenPort, PT_ClosePort
PT_DownloadFile	
Purpose	
	To download a file of PC to Terminal
Syntax	
	int PT_DownloadFile(HANDLE hPort, LPTSTR pPCFile, LPTSTR pPTFile, int nFlag, int nShowProcess);
Return Value	
	Return **Zero** if function success, or return error code.

	To get the error information, please refer error code table(Appendix B.2).
Parameters	
	hPort(in) Handle to the communication port. The PT_OpenPort function returns this handle **pPCFile(in)** Specifies the source file with full path in PC **pPTFile(in)** Specifies the destination file with full path in Terminal **nFlag(in)** Specifies the flags that how to deal with if the file is exist in terminal 0 -> return error code if file is exist 1 -> overwrite the file anyway if file is exist **nShowProcess(in)** Enable/Disable the transmission process dialog
Remarks	
Example	
	// Download the file-font16.cft with process dialog, overwrite. int nResult = PT_DownloadFile(hPort, "C:\\font16.cft", "D:\\fonts\\font16.cft", 1, 1);
See Also	
	PT_GetPDTFile, PT_DirFile, PT_DeleteFile, PT_DownloadMultiFile
PT_GetPDTFile	
Purpose	
	To copy a file in Terminal to PC
Syntax	
	int PT_GetPDTFile(HANDLE hPort, LPTSTR pPTFile, LPTSTR pPCFile, int nShowProcess);
Return Value	
	Return **Zero** if function success, or return error code. To get the error information, please refer error code

	table(Appendix B.2).
Parameters	
	hPort(in) Handle to the communication port. The PT_OpenPort function returns this handle **pPTFile(in)** Specifies the source file with full path in Terminal **pPCFile(in)** Specifies the destination file with full path in PC **nShowProcess(in)** Enable/Disable the transmission process dialog
Remarks	
Example	
	// Upload the file-font16.cft with process dialog, overwrite. int nResult = PT_DownloadFile(hPort, "D:\\fonts\\font16.cft", "C:\\font16.cft", 1);
See Also	
	PT_DownloadFile, PT_DirFile, PT_DeleteFile, PT_DownloadMultiFile
PT_DirFile	
Purpose	
	Ask the file catalogue in terminal directory
Syntax	
	int PT_DirFile(HANDLE hPort, LPTSTR pFolder, int *pnItems, LPTSTR pData, int *pnMaxBuf);
Return Value	
	Return **Zero** if function success, or return error code. To get the error information, please refer error code table(Appendix B.2).
Parameters	
	hPort(in) Handle to the communication port. The PT_OpenPort function returns this handle **pFolder(in)** Specifies folder that be queried with,

	pnItems(out) Pointer to buffer received the number of queried file. **pData(out)** Pointer to the buffer to receive the data. The buffer data is for DIRINFO structure. **pnMaxBuf(in/out)** The maximum pData buffer size.
Remarks	
	1. If the pnMaxBuf is smaller than indeed size, it will be false and *pnMaxBuf will be the indeed size
Example	

	// Query D:\fonts file
	int nResult, nItem, nBuffSize
	char pBuf[1024];
	
	nBufferSize = 1024;
	nResult = PT_DirFile(hPort, “D:\\fonts”, &nItem, pBuf, &nBufferSize);

See Also	
	PT_DownloadFile, PT_GetPDTFile, PT_DeleteFile, PT_DownloadMultiFile
PT_DeleteFile	
Purpose	
	Delete a assigned terminal file.
Syntax	
	int PT_DeleteFile(HANDLE hPort, LPTSTR strFile);
Return Value	
	Return **Zero** if function success, or return error code. To get the error information, please refer error code table(Appendix B.2).
Parameters	
	hPort(in) Handle to the communication port. The PT_OpenPort function returns this handle **strFile(in)** Specifies file that will be deleted
Remarks	

Example	
	// Delete D:\fonts\font16.cft int nResult = PT_DeleteFile(hPort, “D:\\fonts\\font16.cft”);
See Also	
	PT_DownloadFile, PT_GetPDTFile, PT_DirFile, PT_DownloadMultiFile
PT_DownloadApplication	
Purpose	
	Download application to terminal
Syntax	
	int STDCALL PT_DownloadApplication(HANDLE hPort, LPTSTR APfile, int nShowProcess);
Return Value	
	Return **Zero** if function success, or return error code. To get the error information, please refer error code table(Appendix B.2).
Parameters	
	hPort(in) Handle to the communication port. The PT_OpenPort function returns this handle **APfile(in)** Specifies the application file with full path in PC **nShowProcess(in)** Enable/Disable the transmission process dialog
Remarks	
Example	
	// Download ap.bin with process dialog int nResult = PT_DownloadApplication(hPort, “C:\\Ap.bin”, 1);
See Also	
PT_DownloadMultiFile	
Purpose	

	Ask to download multi files to terminal
Syntax	
	int PT_DownloadMultiFile(HANDLE hPort, LPTSTR pPCFile, LPTSTR pPTFile, int nFlag, int nMaxFile, int nFileIndex, int nShowProcess);
Return Value	
	Return **Zero** if function success, or return error code. To get the error information, please refer error code table(Appendix B.2).
Parameters	
	hPort(in) Handle to the communication port. The PT_OpenPort function returns this handle **pPCFile(in)** Specifies the source file with full path in PC **pPTFile(in)** Specifies the destination file with full path in Terminal **nFlag(in)** Specifies the flags that how to deal with if the file is exist in terminal 0 -> return error code if file is exist 1 -> overwrite the file anyway if file is exist **nMaxFile(in)** Specifies the number of total files that will download to terminal. **nFileIndex(in)** Specifies the number that current file index in this procedure. **nShowProcess(in)** Enable/Disable the transmission process dialog
Remarks	
Example	

	// Download multi file
	int i, nResult, nMaxFile;
	
	for(i = 0 ; i < nMaxFile ; i++)

	{
	nResult = PT_DownloadMultiFile(hPort, strPCFile, strPTFile, 1, nMaxFile, i, 1);
	}

See Also	
	PT_DownloadFile, PT_GetPDTFile, PT_DirFile, PT_DeleteFile
PT_FormatDisk	
Purpose	
	To format disk catalogues of terminal
Syntax	
	int PT_FormatDisk(HANDLE hPort, LPTSTR strDisk);
Return Value	
	Return **Zero** if function success, or return error code. To get the error information, please refer error code table(Appendix B.2).
Parameters	
	hPort(in) Handle to the communication port. The PT_OpenPort function returns this handle **strDisk (in)** Specifies the disk that will be formatted.
Remarks	
Example	

	If(!PT_FormatDisk(hPort, “C:\\”))
	AfxMessageBox(“Format disk C succeed!”);
	Else
	AfxMessageBox(“Format disk C fail!”);

See Also	
	PT_GetDiskInfo, PT_SetTime, PT_GetPDTInformation
PT_GetDiskInfo	
Purpose	
	Inquire terminal relevant information
Syntax	

	int PT_GetDiskInfo(HANDLE hPort, LPTSTR strDisk, long &lTotal, long &lUsed, long &lFree);
Return Value	
	Return **Zero** if function success, or return error code. To get the error information, please refer error code table(Appendix B.2).
Parameters	
	hPort(in) Handle to the communication port. The PT_OpenPort function returns this handle **strDisk (in)** Specifies the inquired disk. **lTotal(out)** Pointer to the variable that receives the number of bytes for disk capacity **lUsed(out)** Pointer to the variable that receives the number of bytes for used space. **lFree(out)** Pointer to the variable that receives the number of bytes for free space.
Remarks	
Example	
	// Get Disk C relevant information long lTotal, lUsed, lFree PT_GetDiskInfo(hPort, "C:\\", &lTotal, &lUsed, &lFree)
See Also	
	PT_FormatDisk, PT_SetTime, PT_GetPDTInformation
PT_SetTime	
Purpose	
	To setup the terminal date and time
Syntax	
	int PT_SetTime(HANDLE hPort, char Time[14]);
Return Value	
	Return **Zero** if function success, or return error code.

	To get the error information, please refer error code table(Appendix B.2).
Parameters	
	hPort(in) Handle to the communication port. The PT_OpenPort function returns this handle **Time(in)** The date and time Format:. YYYYMMDDhhmmss Ex: “20081201123020”
Remarks	
Example	

	// Set terminal time 2008/12/31 11:30:59
	nResult = PT_SetTime(hPort, “20081231113059”);

See Also	
	PT_FormatDisk, PT_GetDiskInfo, PT_GetPDTInformation
PT_GetPDTInformation	
Purpose	
	Inquire terminal relevant information
Syntax	
	int PT_GetPDTInformation(HANDLE hPort, char PDTName[10], char FWVer[10], char SN[20], char EquipID[20]);
Return Value	
	Return **Zero** if function success, or return error code. To get the error information, please refer error code table(Appendix B.2).
Parameters	
	hPort(in) Handle to the communication port. The PT_OpenPort function returns this handle **PDTName(out)** The buffer received terminal name **FWVer(out)**

	The buffer received firmware version **SN(out)** The buffer received serial number **EquipID(out)** The buffer received equip ID
Remarks	
Example	
	char Name[10], Ver[10], SN[20], EquipID[20]; nResult = PT_GetPDTInformation(hPort, Name, Ver, SN, EquipID);
See Also	
	PT_FormatDisk, PT_GetDiskInfo, PT_SetTime
PT_EncryptFile	
Purpose	
	To encrypt a file of PC
Syntax	
	int PT_EncryptFile(LPTSTR strSource, LPTSTR strNewFile);
Return Value	
	Return **Zero** if function success, or return error code. To get the error information, please refer error code table(Appendix B.2).
Parameters	
	strSource(in) Specifies the source file with full path in PC. **strNewFile(in)** Specifies the new encrypted file name in PC. This parameter can be omitted.
Remarks	
Example	

	// Encrypt the file-test.dat to a new file-123.dat.
	int nResult = PT_EncryptFile("C:\\test.dat","C:\\123.dat");
	// Encrypt the file-test.dat to itself. The file-test.dat will be overwritten.

	int nResult = PT_EncryptFile("C:\\test.dat");
See Also	
	PT_DownloadEncryptFile
PT_DownloadEncryptFile	
Purpose	
	To download encrypted file of PC to Terminal.
Syntax	
	int PT_DownloadEncryptFile(HANDLE hPort, LPTSTR pPCSource, int nPTDest, int nFlag, int nShowProcess);
Return Value	
	Return **Zero** if function success, or return error code. To get the error information, please refer error code table(Appendix B.2).
Parameters	
	hPort(in) Handle to the communication port. The PT_OpenPort function returns this handle **pPCSource(in)** Specifies the source file with full path in PC **nPTDest(in)** Specifies the destination path in Terminal 1 -> C:\Data 2 -> D:\Fonts 3 -> D:\Lookup 4 -> D:\Program Other value -> D:\Program **nFlag(in)** Specifies the flags that how to deal with if the file is exist in terminal 0 -> return error code if file is exist 1 -> overwrite the file anyway if file is exist **nShowProcess(in)** Enable/Disable the transmission process dialog
Remarks	
	The file with an extended name ".bas" will be absolutely downloaded to "D:\Program" in terminal.

Example	
	// Decrypt and download the file-default.bas with process dialog, overwrite. int nResult = PT_DownloadEncryptFile(hPort, “C:\\default.bas”, 4, 1, 1);
See Also	
	PT_EncryptFile
PT_SetAgentID	
Purpose	
	Setup the agency ID for security.
Syntax	
	int PT_SetAgentID(HANDLE hPort, LPTSTR UserID, LPTSTR Password);
Return Value	
	Return **Zero** if function success, or return error code. To get the error information, please refer error code table(Appendix B.2).
Parameters	
	hPort(in) Handle to the communication port. The PT_OpenPort function returns this handle **UserID(in)** Set the user identification, 4 ~ 8 characters **Password(in)** Set the user password 4 ~ 8 characters
Remarks	
Example	
	nResult = PT_SetAgentID(hPort, “12345678”, “87654321”);
See Also	
GetPTProtocolVersion	
Purpose	
	Obtain the library edition
Syntax	

	int GetPTProtocolVersion(char *pVer);
Return Value	
	Zero.
Parameters	
	pVer(out) The buffer received version
Remarks	
Example	
	char Ver[20]; nResult = GetPTProtocolversionVer);
See Also	
PT_GetErrorCode	
Purpose	
	Obtain the error code
Syntax	
	int PT_GetErrorCode();
Return Value	
	Return error code. Refer error code table(Appendix B.2).
Parameters	
	None
Remarks	
Example	

	// PORT SERIAL for COM1: 115200
	if(PT_OpenPort(1, 1, NULL, 0, 0, 0, 0, 0) == NULL)
	{
	nError = PT_GetErrorCode();
	return;
	}

See Also	
PT_ConvertFile	

Purpose	
	Convert lookup file format
Syntax	
	int PT_ConvertFile(LPTSTR strSource, LPTSTR strNewFile, char cOriDeli, LPCSTR strOriFieldData, unsigned int uConvFieldCount, char cConvDeli, LPCSTR strConvFieldData);
Return Value	
	Return **Zero** if function success, or return error code. To get the error information, please refer error code table(Appendix B.2).
Parameters	
	strSource(in) Assigned the convert source file to convert. **strNewFile(in)** Assigned a new converted file name. **cOriDeli(in)** Assigned the original separator in source file. If this value is set to **Zero**, the original file is fixed length format and set the field length in parameter **strOriFieldData.** **strOriFieldData(in)** Set each field length. Example: "3,2,2,12,5" **uConvFieldCount(in)** Assigned the convert field count. **cConvDeli(in)** Assigned the converted separator. If this value is set to **Zero**, the converted file is fixed length format and set the field length in parameter **strConvFieldData.** **strConvFieldData(in)** Set each converted field length. Example: "3,2,2,12,5"
Remarks	
Example	
	// Convert order.txt with separator ‘,’(0x2c) to order1.txt with fixed length(Field length 3,2,2,12,5) .

	PT_ConvertFile("C:\\Order.txt", "C:\\Order1.txt", ',', NULL, 5, 0, "3,2,2,12,5");
See Also	

5 Appendix

A. Function List

A.1 Raw Communication Function List

Function	Desecription
OpenCom_Entry	Open communication port without any protocol.
CloseCom_Entry	Close assigned communication port.
CloseAllCom_Entry	Close all opened communication port
SetSerial	Configure the serial port
WriteCom_Entry	Write into the materials to the channel
ReadCom_Entry	Read the materials from channel
PurgeCom	Discards characters in output or input buffer of a specified communication port.
SetReadBuffer	Change the input buffer size
EnumUSB	Refresh connected USB device status
GetUSBNameMaxLen	Query the maximum USB device name length
GetUSBName	Obtain a USB device name.
SetReadEvent	Set the read event to monitor the input buffer
GetError	Get error code
GetCommVer	Obtain raw communication library version.

A.2 Standard Communication Function List

Function	Desecription
PT_OpenPort	Open communication port with standard protocol.
PT_EnumUSB	Refresh connect USB device status
PT_GetUSBNameMaxLen	Obtain maximum USB device name length
PT_GetUSBName	Obtain a USB device name
PT_ClosePort	Close assigned communication port

PT_CloseAllPort	Close all opened communication port
PT_DownloadFile	Download a file to terminal in single file mode
PT_DownloadApplication	Download a application to terminal
PT_DownloadMultiFile	Download a file to terminal in multi-file mode
PT_GetPDTFile	Upload terminal file.
PT_DirFile	Ask the file catalogue in terminal directory
PT_DeleteFile	Delete terminal file
PT_FormatDisk	Format terminal disk
PT_GetDiskInfo	Obtain the disk in terminal space information
PT_SetTime	Setup terminal date and time
PT_GetPDTInformation	Obtain terminal name, firmware version, serial number, equip ID
PT_EncryptFile	Encrypt a file of PC
PT_DownloadEncryptFile	Download encrypted file to terminal
PT_SetAgentID	Setup the agency ID for security.
GetPTProtocolVersion	Obtain standard communication library version
PT_GetErrorCode	Obtain the error code.
PT_ConvertFile	Convert lookup file format

B. Error Code

B.1 Raw Communication Error Code Table

Define	Code	Description
	0	Function Succeed
ERR_NO_SUPPORT_PORT	101	Invalid Port Type
ERROR_OPEN_FAIL	102	The port open fail
ERR_INVALID_HANDLE	103	Invalid handle
ERR_OUT_OF_RANGE	104	The parameter is out of the range
ERR_OUT_OF_MEMORY	105	The buffer is too small
ERR_FUNCTION_FAIL	106	Function fail

ERR_TIMEOUT	107	Work timeout

B.2 Standard Communication Error Code Table

Define	Code	Description
	0	Function Succeed
ERR_NO_SUPPORT_PORT	1101	Invalid Port Type
ERROR_OPEN_FAIL	1102	The port open fail
ERR_OUT_OF_RANGE	1104	The parameter is out of the range
ERR_OUT_OF_MEMORY	1105	The buffer is too small
ERR_FUNCTION_FAIL	1106	Function fail
ERR_TIMEOUT	1107	Work timeout
ERR_INVALID_HANDLE	3001	Invalid handle
ERR_INVALID_PROTOCOL_VERSION	3002	Invalid protocol version
ERR_INVALID_PACKET	3003	Invalid packet
ERR_OTHRER_TRANSMISSION_WORKING	3004	Another transmission procedure is working
ERR_NOT_SUPPORT_FUNCTION	3005	This version protocol not support this function
ERR_FILE_OPEN_FAIL	3006	File open fail
ERROR_ERAD_TIMEOUT	3007	Communication timeout
ERR_TRANSMISSION_FAIL	3008	Transmission fail
ERROR_INSUFFICIENT_SPACE	3009	The buffer is too small for received data
ERR_TRANS_BREAK	3010	Transmission break
ERR_INVALID_FILE	3011	The file is not exist
ERR_INVALID_FUNCTION	3012	Terminal not support this function
ERR_T_FUNCTION_FALSE	4001	Function false
ERR_T_DISK_UNFORMAT	4002	Disk unformat
ERR_T_INSUFFICIENT_SPACE	4003	Disk is insufficient space
ERR_T_FILE_EXIST	4004	File have already exist
ERR_INVALID_PATH	4005	File path error
ERR_T_NOT_IN_AP_DL_MO	4006	The terminal not in AP

DE		download mode
ERR_T_FILE_IN_USING	4007	The file is in using
ERR_T_FILE_NOT_EXIST	4008	File is not exist
ERR_PARAMETER_ERROR	4009	Parameter error
ERR_T_WRITE	4010	Terminal system error. Report the error code for manufacturer.
ERR_T_IPL_INSUFFICIENT	40011	
ERR_T_BOOT_INSUFFICIENT	40012	
ERR_T_KERNEL_INSUFFICIENT	40013	
ERR_T_HEAP_RUN_OUT	40014	

C. Structure Description

C.1 DIRINFO

```
typedef struct DIRINFO
{
    unsigned char    assFName[8];    // Filename
    unsigned char    assExtend[3];   // Extension
    unsigned char    usType;         // File type
    unsigned char    usDate[8];      // Access Date
    unsigned char    usTime[6];      // Access Time
    unsigned int     ulSize;         // file size
    unsigned char    endChar[2];     // partition char.(\x0d\x0a)
}_DIRINFO;
```

assFName[8];	XXXXXXXX	Name for directory or file
assExtend[3];	XXX	Extended name for file
usType;	0x00 or 0x01	The point is to appoint the assFName for directory.
	0x02	The point is to appoint the assFName and assExtend for file.
usDate[8];	YYYYMMDD	Date:20081231(2008/12/31)
usTime[6]	hhmmss	Time:123140(12:31:40)
ulSize	xxxx	file size
endChar[2[		partition char.(\x0d\x0a)

6