

VxcommCE.DLL

User Guide

for Visual studio 2003 C#.NET, and
VB.NET

(Version 1.1)

Dynamic Link Library (DLL) for WinCon-8000
Using I-7188E/I-8000 Series VxComm Controller

Warranty

All products manufactured by ICPDAS Inc. are warranted against defective materials for a period of one year from the date of delivery to the original purchaser.

Warning

ICPDAS Inc. assumes no liability for damages consequent to the use of this product. ICPDAS Inc. reserves the right to change this manual at any time without notice. The information furnished by ICPDAS Inc. is believed to be accurate and reliable. However, no responsibility is assumed by ICPDAS Inc. for its use, or for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Copyright 1997-2004 by ICPDAS Inc., LTD. All rights reserved worldwide.

Trademark

The names used for identification only maybe registered trademarks of their respective companies.

License

The user can use, modify and backup this software on a single machine. The user may not reproduce, transfer or distribute this software, or any copy, in whole or in part.

Contents

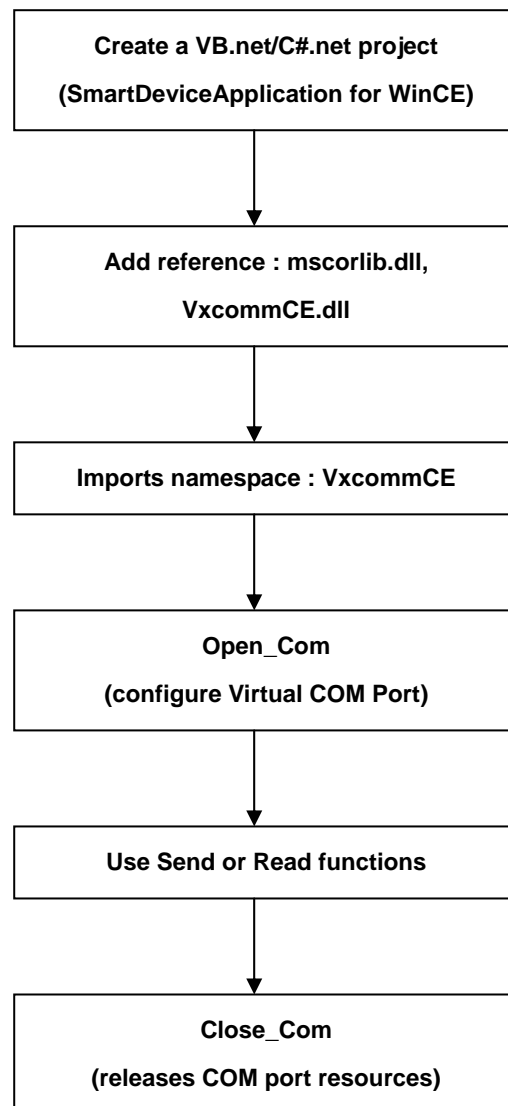
CONTENTS.....	2
1 INTRODUCTION VXCOMMCE.DLL.....	3
1.1 Introduction	3
1.2 Architecture	3
2 VXCOMMCE.DLL SDK.....	4
2.1.0 Open_Com	5
2.1.1 Close_Com	6
2.1.2 Send_Read_String.....	7
2.1.3 Send_String.....	8
2.1.4 Read_String	9
2.1.5 Send_Read_Binary.....	10
2.1.6 Send_Binary.....	12
2.1.7 Read_Binary	13
2.1.8 Get_Com_Status.....	15
2.1.9 DataInCom.....	16
2.2 Error Code Table.....	17

1 Introduction VxcommCE.DLL

1.1 Introduction

VxcommCE supports VB.net and C#.net. User can develop WindowsCE applications for WinCon. VxcommCE is be able to create Virtual COM Ports from a server(I-7188E/I-8000). The maximum Virtual COM Port number is 255.

1.2 Architecture



2 VxcommCE.DLL SDK

In this section we will focus on examples for the description and application of the control functions on the use Vxcomm for use on Wincon-8000. The functions on the VxcommCE.DLL can be classified 11 parts as below:

1. Open Com Function;
2. Close Com Function;
3. Send and Read String Function;
4. Send String Function;
5. Read String Function;
6. Send and Read Binary data Function;
7. Send Binary data Function;
8. Read Binary data Function;
9. Get Com Status Function;
10. Data in Com Fcuction .
11. Error Code Table

For the application of visual studio VB.NET and C#.NET, the namespace for VxcommCE.DLL is **VxcommCE** and should not be changed during the development of applications. However, the Vxcomm.DLL on Wincon-8000 should be used with a 7188E/8000.. which is VxComm Server firmware inside.

2.1.0 Open_Com

Description:

This function is used to configure and open a Server(7188E/8000..) COM port for service by Ethernet. The COM port must be called up once before users can begin sending/receiving commands through it. The Com_No of Virtual COM is definable by user.

Syntax:

[Visual Basic.net, C#.net]

```
byte Vxcomm.Open_Com(byte Com_No, string Server_IP, uint16  
Command_Port, byte Server_COM, int Baudrate, byte CData, byte Cparity, byte  
CStop)
```

Parameter:

Com_No : [Input] Virtual COM Port No. 0~255
Server_IP: [Input] IP Address of Server(7188E/8000..
Command_Port: [Input] Base Command Port
Server_COM: [Input] COM port of Server(7188E/8000..
Baudrate : [Input] 300/600/1200/1800/2400/4800/7200/9600/19200/38400/
57600/115200
Cdata : [Input] 5/6/7/8 data bit
CParity : [Input] 0=None Parity, 1=Odd Parity, 2=Even Parity
CStop : [Input] 1=1-stop, 2=2-stop

Return Value:

0: NO_ERROR
Others: Refer to error code table

Example:

[Visual Basic.net]

```
Dim result As Byte
```

```
result = Vxcomm.Open_Com(3, "192.168.255.1", Convert.ToInt16(10000), 1, 9600, 8, 0, 1)  
' If result = 0 the opening virtual COM3, Port1 of Server success.
```

Remark:

2.1.1 Close_Com

Description:

This function closes and releases COM port resources from the Wincon and Server. It must be called on before exiting the application program. The Open_Com function will return error message if the program exit without calling Close_Com function.

Syntax:

[Visual Basic.net, C#.net]

Vxcomm.Close_Com(byte Com_No)

Parameter:

Com_No : [Input] Virtual COM Port No. 0~255

Example:

[Visual Basic.net]

```
Vxcomm.Close_Com(3)  
'closing COM3 port.
```

Remark:

2.1.2 Send_Read_String

Description:

This function sends a string to the Virtual COM and then will receive a response from it.

Syntax:

<p style="text-align: center;">[Visual Basic.net, C#.net]</p> <pre>byte Vxcomm.Send_Read_String(byte Com_No, string Out_String, String In_String)</pre>

Parameter:

Com_No : [Input] Virtual COM Port No. 0~255
Out_String: [Input] Sending string
In_String: [Output] Receiving the response string from the modules

Return Value:

0: NO_ERROR
Others: Refer to error code table

Example:

```
[Visual Basic.net]  
  
Dim result As Byte  
result = Vxcomm.Open_Com(3, "192.168.255.1", 2, 9600, 8, 0, 1)  
  
Dim SS As String  
Dim RR As String  
  
SS = "$03M" & VbCrLf 'Use string-command(must add "vbCrLf") to control I-7000/I-87K modules  
result = Vxcomm.Send_Read_String(3, SS,RR )  
'RR string is to receive data.
```

Remark:

2.1.3 Send_String

Description:

This function sends a string to the Virtual COM.

Syntax:

[Visual Basic.net, C#.net]

byte Vxcomm.Send_String(byte Com_No, string data)

Parameter:

Com_No : [Input] Virtual COM Port No. 0~255

data: [Input] Sending string

Return Value:

0: NO_ERROR

Others: Refer to error code table

Example:

[Visual Basic.net]

```
Dim result As Byte
```

```
result = Vxcomm.Open_Com(3, "192.168.255.1", 2, 9600, 8, 0, 1)
```

```
Dim SS As String
```

```
SS = "$03M" & VbCrLf 'Send sting-command(must add "vbCrLf") from the Com Port out.
```

```
result = Vxcomm.Send_String(3, SS)
```

Remark:

2.1.4 Read_String

Description:

Users can utilize this function to obtain a response string from Virtual COM.

Syntax:

[Visual Basic.net, C#.net]	
<code>byte</code>	<code>Vxcomm.Read_String(byte Com_No, String data)</code>

Parameter:

cPort : [Input] Virtual COM Port No. 0~255
data: [Output] Receiving the response string from the modules

Return Value:

0: NO_ERROR
Others: Refer to error code table

Example:

[Visual Basic.net]

```
Dim result As Byte
Dim RR As String
result = Vxcomm.Open_Com(3, "192.168.255.1", 2, 9600, 8, 0, 1)
result = Vxcomm.Read_String (3, RR)
'The RR string is to receive data.
```

Remark:

2.1.5 Send_Read_Binary

Description:

To send out a binary data via fix length. Send_Binary terminates the sending process by the string length "iLen" instead of the character "CR"(Carry return). Therefore, this function can send out a command string with or without a null character under the consideration of the command length. However, because this function operates without any error checking mechanisms (Checksum, CRC, LRC... etc.), the user would have to add in error checking information into the raw data manually, if a communication checking system was required.

Syntax:

[Visual Basic.net, C#.net]

```
byte Vxcomm.Send_Read_Binary(byte Com_No, byte[] Send_Data, Int16  
Send_Len, byte[] Read_Data, Int16 Read_Len)
```

Parameter:

Com_No : [Input] Virtual COM Port No. 0~255
Send_Data: [Input] Sending data byte
Send_iLen: [Input] The length of data
Read_Data: [Output] result data byte
Read_Len: [Input] The length of result data.

Return Value:

0: NO_ERROR
Others: Refer to error code table

Example:

[Visual Basic.net]

```
Dim result As Byte  
result = Vxcomm.Open_Com(3, "192.168.255.1", 1, 9600, 8, 0, 1) 'Initial Com3, Open port 1 of server,  
Dim s(100) As Byte  
Dim SS as String  
Dim r(100) as Byte  
SS = "Hello World" 'Send sting from the Com Port out.  
s = System.Text.Encoding.ASCII.GetBytes(SS)  
result = Vxcomm.Send_Read_Binary(3, s, SS.Length, r, 100)
```

```
Dim len As Int16  
len = 11  
Dim RR As String  
RR = System.Text.Encoding.ASCII.GetString(r, 0, len)  
'The RR string is to receive data.
```

Remark:

2.1.6 Send_Binary

Description:

To send out a binary data via fix length. It is controlled by the parameter "iLen". The difference between this function and the Send_String function is that Send_Binary terminates the sending process by the string length "iLen" instead of the character "CR"(Carry return). Therefore, this function can send out a command string with or without a null character under the consideration of the command length. However, because this function operates without any error checking mechanisms (Checksum, CRC, LRC... etc.), the user would have to add in error checking information into the raw data manually, if a communication checking system was required.

Syntax:

[Visual Basic.net, C#.net]

```
byte Vxcomm.Send_Binary(byte Com_No, byte[] data, Int16 iLen)
```

Parameter:

Com_No : [Input] Virtual COM Port No. 0~255
data: [Input] Sending data byte
iLen: [Input] The length of data.

Return Value:

0: NO_ERROR
Others: Refer to error code table

Example:

[Visual Basic.net]

```
Dim result As UInt16  
result = Vxcomm.Open_Com(3, "192.168.255.1", 1, 9600, 8, 0, 1)  
Dim s(100) As Byte  
Dim SS as String  
SS = "Hello World" 'Send sting from the Com Port out.  
s = System.Text.Encoding.ASCII.GetBytes(SS)  
result = Vxcomm.Send_Binary(3, s, SS.Length)
```

Remark:

2.1.7 Read_Binary

Description:

This function is applied to receive a fix length response. The length of the receiving response is controlled by the parameter "iLen". The difference between this function and the Read_String function is that Read_Binary function terminates the receiving process using the string length "iLen" instead of the character "CR"(Carry return). Therefore, this function can be used to receive response string data with or without a null character under the consideration of the receiving length. Besides, because this function operates without any error checking mechanisms (checksum, CRC, LRC... etc.), the user has to remove their error checking information from the raw data themselves if a communication checking system must be applied.

Syntax:

[Visual Basic, C#.net]

```
byte Vxcomm.Read_Binary(byte Com_No, byte[] data,  
                        Int16 In_len)
```

Parameter:

Com_No : [Input] Virtual COM Port No. 0~255
data: [Output] The receiving data from the module
In_len: [Input] The length of result data.

Return Value:

0: NO_ERROR
Others: Refer to error code table

Example:

[Visual Basic.net]

```
Dim result As Byte  
result = Vxcomm.Open_Com(3, "192.168.255.1", 1, 9600, 8, 0, 1)  
Dim r(100) As Byte  
Dim len As Int16  
len = 10  
result = Vxcomm.Read_Binary(3, r, 100)  
Dim RR As String
```

```
RR = System.Text.Encoding.ASCII.GetString(r, 0, len)
```

```
'The RR string is to receive data.'
```

Remark:

2.1.8 Get_Com_Status

Description:

This function can obtain the Virtual COM Port's status. If the return value is "0" (false), it means that "The COM Port is not in use!", but if the return value is "1" (true), it means that "The COM Port is in use!"

Syntax:

[Visual Basic.net, C#.net]

`byte Vxcomm.Get_Com_Status(byte Com_No)`

Parameter:

Com_No : [Input] Virtual COM Port No. 0~255

Return Value:

0: COM port is not in used.
1: COM port is in used.

Example:

[Visual Basic]

```
Dim result As Byte
result = Vxcomm.Get_Com_Status(3)
' If result = 0 COM port is not in used.
```

Remark:

2.1.9 DataInCom

Description:

Getting know that data exists on the input buffer of Virtual COM port.

Syntax:

[Visual Basic.net, C#.net]	
<code>boolean</code>	<code>VxcommCE.DataInCom(byte Com_No)</code>

Parameter:

Com_No : [Input] Virtual COM Port No. 0~255

Return Value:

True : There is data in buffer

Flase : There is no data in buffer

Example:

[Visual Basic]

```
Dim RET As Boolean
Vxcomm.Open_Com(3, "192.168.255.1", 2, 8, 0, 1)
RET = Vxcomm.DataInCom(3)
```

Remark:

2.2 Error Code Table

Value	Description
0	OK
8	Com_NO was not open
15	Time Out
90	Com Config Setting Error
99	Com Config Socket Error