Application Note #0212

Network Applications with the WINSOCK Control

Introduction

Data capture and/or machine control can be provided over a standard Ethernet network using the data collection products (ET21x, EP21x, ED21x, etc.) manufactured by Computerwise, Inc. Often the application program used with these devices will be written using either the popular Visual Basic (VB) or Visual Fox Pro (VFP) development system. The “Professional” and “Enterprise” versions of these Visual Studio products includes the WINSOCK control to simplify network communications. This application note will explain some basic concepts for using the WINSOCK control to perform communications with the Ethernet devices in an application. It is assumed the reader is already familiar with network basics, the Ethernet device to be used, VB or VFP programming and the Windows operating system. No attempt will be made to explain VB or VFP programming or application software design.

Overview of the Connection

Network communications with the Ethernet data collection device(s) is accomplished using a TCP/IP “socket” connection. A “socket” is simply a logical communications channel specifying the two (2) end points of the connection. Each Ethernet device is configured with a unique IP address and port number (see device manual for configuration details). The application program must make a “socket” connection to the device using the IP address and port number assigned to it. Once the “socket” connection is established data can be transferred to and from the target device as a data stream. The contents of the actual data stream will depend on the target device (see target device manual).

WINSOCK Control with VB

The WINSOCK control can be added to a form like any other control function and allows an easy way to establish a socket connection with the target network device. It provides properties, methods and events which allow a connection to be established and perform communications with the network device. One WINSOCK control is required for each network target device socket. An array of WINSOCK controls provides a simple way for an application program to communicate with multiple target devices.

Establishing a Connection

A socket connection can be established with the network target device using the WINSOCK control. Since the control can be used for either TCP or UDP communications the “Protocol” property must be set for TCP protocol (default) first. The IP address and port number of the target device are stored in the “RemoteHost” and “RemotePort” properties. These properties can be set either at design time or at run time. Once the properties are set the “Connect” method is used to establish a connection with the target device.

```
Winsock1.Protocol = sckTCPProtocol
Winsock1.RemoteHost = “192.168.168.50”
Winsock1.RemotePort = 1070
Winsock1.Connect
```

The “Connect” event is triggered when the connection is established. This event does not have to be used but can be helpful to identify that the connection is ready for use.

```
Private Winsock1_Connect()
   Msgbox “Connection established”
End Sub
```

Sending Data

Once the connection is established data can be sent to the target device using the “SendData” method. The “SendComplete” and “SendProgress” events can be used to monitor the transmission.

```
Winsock1.SendData “DISPLAY=Hello World” & vbCr
```

Receiving Data

Normally, the WINSOCK “DataArrival” event is used to capture data sent by the target device. Within the event the “GetData” method is used to capture the received data. Typically, the captured data is moved to a temporary buffer and then processed by another routine. Since the received data is a stream the processing routine may have to parse incomplete strings or blocks as they arrive.

```
Private Sub Winsock1_DataArrival(ByVal bytesTotal As Long)
   Dim buf As String

   Winsock1.GetData buf
   ProcessingRoutine buf
End Sub
```

Closing the Connection

The established socket connection typically remains open for as long as it is needed to communicate with the target device. Either end of an established socket connection can close the connection when it is no longer needed. The application program can use the WINSOCK “Close” method to force the connection closed. If the target device initiates the close the WINSOCK “Close” event is triggered to notify the program that the connection has been closed. If the connection needs to be reestablished it should first be closed and then reconnected as described above in establishing a connection.

```
Winsock1.Close
```

Monitoring the Connection

The WINSOCK control provides a “State” property which can be read at any time to determine the current connection status. An “Error” event is triggered whenever a connection error as occurred. Often the “State” property is used in conjunction with the “Error” event to determine what steps should be taken. Constants for the “State” properties and “Error” event “Number” can be found in the VB documentation for the WINSOCK control.

```
Private Sub Winsock1_Error(ByVal Number As Integer, Description As String, ...)
   Dim ms as String

   ms = “Error=” & Number & “ “ & Description & vbCrLf
   ms = ms & Winsock1.State
   Msgbox ms,,”Connection Error”
End Sub
```

WINSOCK Control with VFP

Using the WINSOCK control with a Visual Fox Pro application is very similar to using it with a Visual Basic application. The main difference is the syntax used. The previous VB examples can be rewritten for a VFP application as follows.

Establishing a Connection

```
thisform.Winsock1.Protocol = 0
thisform.Winsock1.RemoteHost = “192.168.168.50”
thisform.Winsock1.RemotePort = 1070
thisform.Winsock1.Connect
```

Connect Event

```
Messagebox(“Connection Established”)
```

Sending Data

```
thisform.Winsock1.SendData(“DISPLAY=Hello World” + CHR(13))
```

Receiving Data – DataArrival Event

```
LPARAMETERS bytestotal
LOCAL cString
this.GetData(@cString, 8, bytestotal)
thisform.ProcessingRoutine(cString)
```

Closing the Connection

```
thisform.Winsock1.object.Close
```

Monitoring the Connection – Error Event

```
LPARAMETERS number, description, scode, source, helpfile, helpcontext, canceldisplay
LOCAL cMesg
cMesg = "Error =" + ALLTRIM(STR(number))+" "+description+CHR(13)
cMesg = cMesg + thisform.WinSock1.State
Messagebox(cMesg,0,"Connection Error")
```

Summary

The WINSOCK control makes it extremely simple for the VB or VFP application programmer to provide network communications with a target network device.