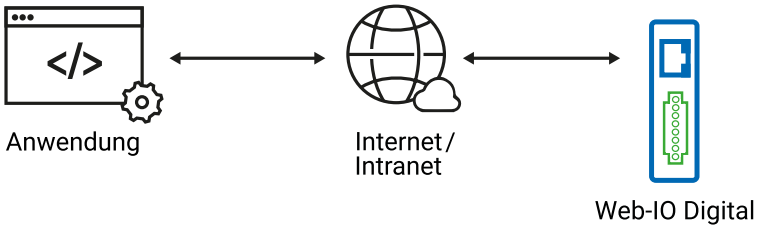
Applikation zum Web-IO Digital:

Weitere Applikationen
Produktübersicht

Web-IO Digital mit Visual Basic steuern und überwachen

Visualisieren, Überprüfung und Steuern

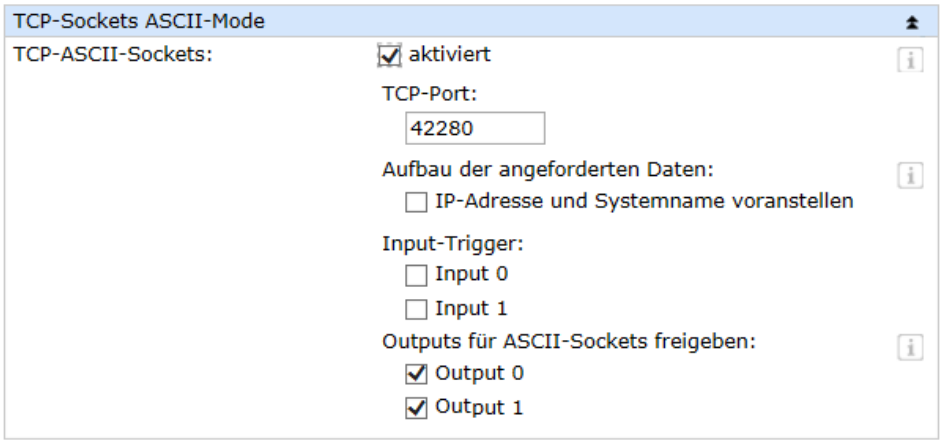
Als einfach zu erlernende Hochsprache bietet Visual Basic alles, was zum Programmieren von TCP/IP-Anwendungen nötig ist. Damit ist Visual Basic auch ein beliebtes Hilfsmittel um Anwendungen zu erstellen, die mit dem Web-IO Digital kommunizieren. Zusätzliche Treiber oder DLLs werden nicht benötigt.



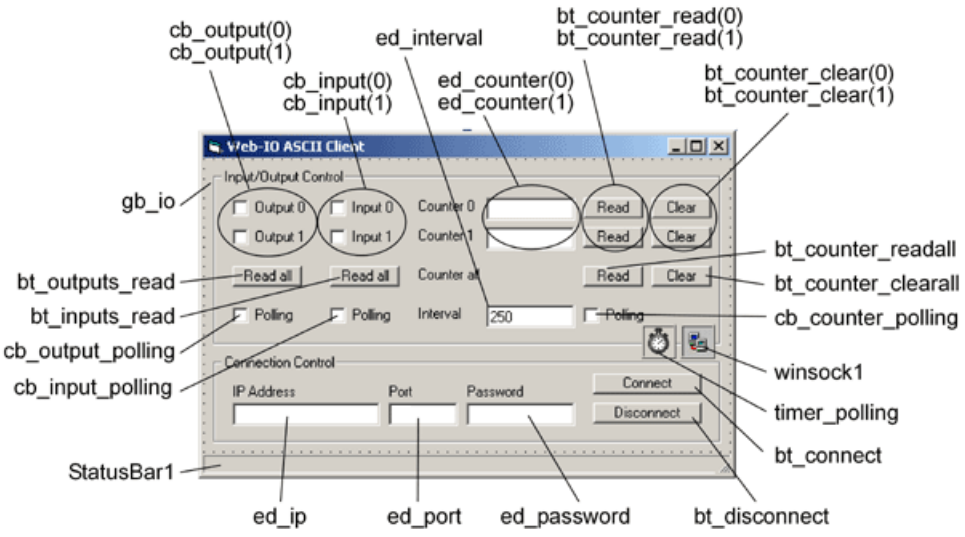
Mit dem folgenden Programmbeispiel können Sie Ihr Web-IO Digital mit seinen Inputs und Outputs in einer Windows-Anwendung abbilden. Darüber hinaus können Sie die Outputs des Web-IO schalten.

Vorbereitungen

- Web-IO mit Spannung versorgen und IOs verdrahten
- Web-IO mit dem Netzwerk verbinden
- IP-Adressen vergeben
- Beim Web-IO im Bereich *Kommunikationswege >> Socket-API* die *TCP-ASCII-Sockets* aktivieren und die *Outputs zum Schalten freigeben*



Zusammenstellen der verschiedenen Bedienelemente und Anzeigeobjekte im VB-Form



Bei der Benennung der einzelnen Objekte ist es hilfreich, sinngebende Namen zu verwenden. In diesem Beipiel beschreibt der erste Teil des Namens die Art des Objektes und der zweite Teil die Funktion.

Programmstart

Einrichten der Bedienelemente
Die Gruppe mit den Bedienelementen für das Web-IO wird zunächst für die Bedienung gesperrt. In der Statuszeile wird angezeigt, dass noch keine Verbindung besteht.

```
  Private Sub Form_Load()
    fr_io.Enabled = False
    bt_disconnect.Enabled = False
    StatusBar1.SimpleText = "No Connection"
  End Sub
```

Die Verbindungskontrolle

Einleiten der Verbindung
Durch Eingabe der IP-Adresse des Web-IO in das Textfeld*ed_ip* und Klick auf den Button *bt_connect* wird der Verbindungsaufbau gestartet.

```
  Private Sub bt_connect_Click()
    If ed_ip.Text <> "" Then
      winsock1.RemoteHost = ed_ip.Text
      winsock1.RemotePort = Val(ed_port.Text)
      winsock1.Connect
    End If
  End Sub
```

Verbindungsaufbau
Für die Abwicklung des TCP/IP-Handlings wird das Winsock-Steuerelement von Visual Basic benutzt. Dieses Steuerelement erlaubt es, bei Eintreten verschiedener Verbindungszustände Prozeduren vorzugeben und zu starten.

Verbindung kommt zustande
Sobald das Web-IO die Verbindung annimmt, führt das Winsock-Steuerelement die entsprechende Prozedur aus. In der Statuszeile wird das Zustandekommen der Verbindung angezeigt, die Bedienelemente werden zur Benutzung freigegeben und der Disconnect-Button wird bedienbar. Der Connect-Button wird für die Bedienung gesperrt.

```
  Private Sub winsock1_Connect()
    fr_io.Enabled = True
    bt_disconnect.Enabled = True
    bt_connect.Enabled = False
    StatusBar1.SimpleText = "Connected to " + ed_ip.Text
  End Sub
```

Trennen der Verbindung
Die Verbindung bleibt solange bestehen, bis sie vom Benutzer durch Klick auf den Disconnect-Button beendet wird oder das Web-IO die Verbindung beendet.

```
  Private Sub bt_disconnect_Click()
    winsock1.Close
    fr_io.Enabled = False
    bt_disconnect.Enabled = False
    bt_connect.Enabled = True
    StatusBar1.SimpleText = "No Connection"
  End Sub
```

Auch in diesem Fall ruft das Winsock-Steuerelement eine entsprechende Prozedur auf

```
Private Sub winsock1_Close()
  fr_io.Enabled = False
  bt_disconnect.Enabled = False
  bt_connect.Enabled = True
  StatusBar1.SimpleText = "No Connection"
  winsock1.Close
End Sub
```

Verbindungsfehler
Auch im Fall eines Verbindungsfehlers führt das Winsock-Steuerelement eine entsprechende Prozedur aus, die im Groben der Disconnect Prozedur entspricht. In der Statuszeile wird zusätzlich die Winsock-Fehlernummer mit ausgegeben.

```
Private Sub winsock1_Error(ByVal Number As Integer, Description As String, _
ByVal Scode As Long, ByVal Source As String, ByVal HelpFile As String, _
ByVal HelpContext As Long, CancelDisplay As Boolean)
  fr_io.Enabled = False
  bt_disconnect.Enabled = False
  bt_connect.Enabled = True
  StatusBar1.SimpleText = "ERROR " + Str(Number) + " : " + Description
  winsock1.Close
End Sub
```

Bedienung und Kommunikation von Client-Seite

Sobald eine Verbindung mit dem Web-IO zustandegekommen ist, kann der Anwender durch Bedienung der entsprechenden Programmelemente Kommandos an das Web-IO senden.

Setzen der Outputs

Das Setzen der Outputs wird dem Anwender über zwei Checkboxen cb_outputx ermöglicht. Das Programm nutzt dazu das MouseUP-Ereignis dieses Objektes. Wird ein Mouse Up, also ein Loslassen der Output-Checkbox registriert, führt das Programm die entsprechende Prozedur aus und gibt - je nach dem ob die Checkbox gesetzt ist oder nicht - das passende Kommando an das Web-IO weiter.

```
Private Sub cb_output_MouseUp(Index As Integer, Button As Integer, _
Shift As Integer, X As Single, Y As Single)
  If cb_output(Index).Value Then
    winsock1.SendData ("GET /outputaccess" + Trim(Str(Index)) + _
    "?PW=" + ed_password.Text + "&State=ON&")
  Else
    winsock1.SendData ("GET /outputaccess" + Trim(Str(Index)) + _
    "?PW=" + ed_password.Text + "&State=OFF&")
  End If
End Sub
```

Output/Input-Status Abfragen
Den Status der Outputs und Inputs kann der Anwender durch Anklicken des zugehörigen Buttons anfordern.

```
Private Sub bt_outputs_read_Click()
  winsock1.SendData ("GET /output?PW=" + ed_password.Text + "&")
End Sub

Private Sub bt_inputs_read_Click()
  winsock1.SendData ("GET /input?PW=" + ed_password.Text + "&")
End Sub
```

Counter abfragen Löschen
Auch die Zählerstände der input-Counter lassen sich abfragen bzw. löschen.

```
Private Sub bt_counter_read_Click(Index As Integer)
  winsock1.SendData ("GET /counter" + Trim(Str(Index)) + _
  "?PW=" + ed_password.Text + "&")
End Sub

Private Sub ed_counter_clear_Click(Index As Integer)
  winsock1.SendData ("GET /counterclear" + Trim(Str(Index)) + _
  "?PW=" + ed_password.Text + "&")
End Sub
```

Natürlich lassen sich auch alle Counter zeitgleich lesen bzw. löschen.

```
Private Sub cb_counter_readall_Click()
  winsock1.SendData ("GET /counter?PW=" + ed_password.Text + "&")
End Sub

Private Sub cb_counter_clearall_Click()
  winsock1.SendData ("GET /counterclear?PW=" + ed_password.Text + "&")
End Sub
```

Datenempfang vom Web-IO

Auswerten und Anzeigen der empfangenen Daten
Alle Kommandos und Anfragen an das Web-IO werden mit einem Antwort-String quittiert. Dabei haben die Antworten je nach Type einen spezifischen Aufbau.

Für die Outputs: output;<Binärwert des Outputstatus im hexadezimalen Format>
Für die Inputs: input;<Binärwert des Outputstatus im hexadezimalen Format>
Für die Counter: counterx;<dezimaler Zählerstand>
oder counter;<dezimaler Zählerstand 0 >; <dezimaler Zählerstand 0 >; ..... wenn alle Counter auf einmal gelesen werden sollen.
Alle Antwort-Strings sind mit einem 0-Byte abgeschlossen.

Werden vom Winsock-Steuerelement Daten empfangen, ruft dieses die entsprechende Prozedur auf.

```
Private Sub winsock1_DataArrival(ByVal bytesTotal As Long)
  Dim receivestring As String
  winsock1.GetData receivestring
  Select Case Left(receivestring, 1)
    Case "i"
      If (Val(Mid(receivestring, 7, 1)) And 1) = 1 Then
        cb_input(0).Value = 1
      Else
        cb_input(0).Value = 0
      End If
      If (Val(Mid(receivestring, 7, 1)) And 2) = 2 Then
        cb_input(1).Value = 1
      Else
        cb_input(1).Value = 0
      End If
    Case "o"
      If (Val(Mid(receivestring, 8, 1)) And 1) = 1 Then
        cb_output(0).Value = 1
      Else
       cb_output(0).Value = 0
      End If
      If (Val(Mid(receivestring, 8, 1)) And 2) = 2 Then
        cb_output(1).Value = 1
      Else
        cb_output(1).Value = 0
      End If
    Case "c"
      Dim tabpos
      If Mid(receivestring, 8, 1) = "0" Then
        ed_counter(0).Text = Mid(receivestring, 10, Len(receivestring) - 10)
      End If
      If Mid(receivestring, 8, 1) = "1" Then
       ed_counter(1).Text = Mid(receivestring, 10, Len(receivestring) - 10)
      End If
      If Mid(receivestring, 8, 1) = ";" Then
        tabpos = InStr(9,receivestring, ";")
        ed_counter(0).Text = Mid(receivestring,9, tabpos - 9)
        ed_counter(1).Text = Mid(receivestring,tabpos + 1, _
        Len(receivestring) - tabpos - 1)
      End If
  End Select
End Sub
```

Die Empfangsprozedur überprüft anhand des ersten Zeichens der Empfangsdaten, ob es um Input-, Output- oder Counter-Meldungen geht. Abhängig davon, wird z.B. festgestellt welcher Output welchen Status hat. Bei den Countern ist es sowohl möglich, einzelne Zählerwerte abzufragen, als auch alle Counter in einem Zug auszulesen. Die einzelnen Zählerstände werden dann dezimal, mit Semikolon getrennt in einem String ausgegeben.

Polling

Zyklisches Abfragen bestimmter Werte
Um auch eine automatische Aktualisierung der Anzeige zu ermöglichen, wird ein Timer benutzt. In Abhängigkeit der Checkboxen für Output-, Input- und Counter-Polling werden die entsprechenden Informationen im eingestellten Intervall vom Web-IO abgerufen.

```
Private Sub timer_polling_Timer()
  If winsock1.State = sckConnected Then
    If cb_output_polling.Value Then
      winsock1.SendData ("GET /output?PW="+ ed_password.Text + "&")
    End If
    If cb_input_polling.Value Then
      winsock1.SendData ("GET /input?PW="+ ed_password.Text + "&")
    End If
    If cb_counter_polling.Value Then
      winsock1.SendData ("GET /counter?PW="+ ed_password.Text + "&")
    End If
  End If
End Sub
```

Das gewünschte Intervall kann in das entsprechende Textfeld eingegeben werden. Bei Änderung wird das Timer-Intervall dann automatisch angepasst.

```
Private Sub ed_interval_Change()
  timer_polling.Interval = Val(ed_interval)
End Sub
```

Natürlich kann das Web-IO auch so konfiguriert werden, dass es von sich aus Meldungen zum Programm sendet, wenn sich an einem der Inputs etwas ändert.

Das Beispielprogramm unterstützt alle gängigen Funktionen des Web-IO im Kommando-String Modus, optimiert für das Web-IO 2x Digital Input, 2x Digital Output. Für die anderen Web-IO Modelle müssen ggf. Anpassungen am Programm vorgenommen werden. Weitere Programmbeispiele zur Socket-Programmierung finden Sie auf den Tool-Seiten zum Web-IO. Eine detaillierte Beschreibung zur Socketschnittstelle der Web-IO Digital Modelle finden Sie im Referenzhandbuch.

Programmbeispiel herunterladen

Produkte

Web-IO 4.0 Digital
2xIn, 2xOut

Bei Bedarf auch über PoE zu versorgen

Web-IO 4.0 Digital
12xIn, 12xOut

12x Eingänge,
12x Ausgänge

Weitere Web-IOs

Alle W&T Web-IO Digital 24V

W&T
www.WuT.de

Wir sind gerne persönlich für Sie da:

Wiesemann & Theis
GmbH
Porschestr. 12
42279 Wuppertal
Tel.: 0202/2680-110 (Mo-Fr. 8-17
Uhr)
Fax: 0202/2680-265
info@wut.de

Datenschutz