Dispatcher Schnittstellen und Protocols

Protocols

BoBo (420) MicroController auf der RS232

```
+--------------------+---------------+--------------+
|  Befehl (2 bytes)  | Wert (1 byte) |  0 (2 bytes) |
+--------------------+---------------+--------------+
```

HW/SW

UBW32

Verwendung

- Einige Eingänge / Ausgänge wurden am BoBo Führerstand vom alten MicroController schon auf eine UBW32 migriert.
- Das Stellwerk ist an einem UBW32 angeschlossen
- Zukünftige Führerstände sollten auch mit diesem System gemacht werden

Technologie

- RS232, Synchron → ein timer polled die Schnittstelle alle 100ms
  .BaudRate = 9600
  .Parity = Parity.None
  .DataBits = 8
  .StopBits = StopBits.One

http://www.schmalzhaus.com/UBW32/doc/UBW32Documentation_v1_6_3.html

1. Read all register (to capture state of digital inputs)

```
' Read digital ports (all register)
‘ Die “ReadLine()” dazwischen sind da um die “<OK>” von dem Input Buffer zu lesen
‘ (habe dann das alles ignoriert und nur die eigentlichen Daten genommen)
UBW32_1.WriteLine("I" & vbLf)
UBW32_1.ReadLine()
UBW32_1_PortString = UBW32_1.ReadLine()
UBW32_1.ReadLine()
UBW32_1.ReadLine()

' PortStatus = I,<StatusA>,<StatusB>,<StatusC>,<StatusD>,<StatusE>,<StatusF>,<StatusG>
PortArray = Split(UBW32_1_PortString, ",")
Form4.Label1A.Text = "PortA:" & PortArray(1)
Form4.Label1B.Text = "PortB:" & PortArray(2)
Form4.Label1C.Text = "PortC:" & PortArray(3)
Form4.Label1D.Text = "PortD:" & PortArray(4)
Form4.Label1E.Text = "PortE:" & PortArray(5)
Form4.Label1F.Text = "PortF:" & PortArray(6)
Form4.Label1G.Text = "PortG:" & PortArray(7)
```

2. Creates all digital commands for the simulation ()

```
‘ Nun müssen alle pin-Zustände  mit dem vorherigen Zustand verglichen werden.
‘ Falls gleich → ignore
‘ Falls geändert → ein Event generieren und an der Simulation senden.
```

3. Sets all output values (digital and analog) from the "UBW32_1_Queue"

```
‘ Hier müsste der Timer “loop” die “sender queue” ab arbeiten.
‘ Falls gleich → ignore
```

LocSim

Verwendung

Simulationn vom Rohrer
Braucht diverse “init” funtionen um die Simulation zu verbinden.
RS232, asynchron

```
 .BaudRate = 38400
 .Parity = Parity.None
.DataBits = 8
 .StopBits = StopBits.One
.Handshake = Handshake.XOnXOff
```

Technology

Muss zwingend auf “`DataReceived`” event vom OS reagieren:

```
  Private Sub SerialPortSimulator_DataReceived(ByVal sender As System.Object, ByVal e As
System.IO.Ports.SerialDataReceivedEventArgs) Handles SerialPortSimulator.DataReceived
        Me.Invoke(New rxSimulatorDataDelegate(AddressOf RecievedSimulatorData), New Object() {})
    End Sub
```

Command schliesst mit einem CRLF ab:

```
    Private Sub RecievedSimulatorData()
        ' Following Buffer route is build for the "LOCSIM Simulator" only !!!!

        ' insert here route to receive commands from the simulator
        'Dim ContainsCRorY As Boolean = False
        Dim Pos1, Pos2, Pos As Integer
```

```
Dim bytesToRead As Integer


Dim StringCommand As String
Dim readByte As Byte


bytesToRead = SerialPortSimulator.BytesToRead


' Read the bytes into the event buffer
For i = 0 To bytesToRead - 1
    readByte = SerialPortSimulator.ReadByte()
    SimulatorStringBuffer = SimulatorStringBuffer & Chr(readByte)
Next

Label23.Text = SimulatorStringBuffer
Pos1 = InStr(SimulatorStringBuffer, Chr(13))
Pos2 = InStr(SimulatorStringBuffer, Chr(10))

' Check if Carriage Return has been send
If Pos1 <> 0 Or Pos2 <> 0 Then
    If Pos1 <> 0 Then
        Pos = Pos1
    Else
        Pos = Pos2
    End If
    TextBox2.AppendText(SimulatorStringBuffer & vbCrLf)
```

```
            If CheckBox3.Checked Then
                MyFile.Write("From LocSim " & Now & " " & SimulatorStringBuffer & vbCrLf)
            End If

            Do Until Pos = 0 'do until no furhter complete command
                StringCommand = Mid(SimulatorStringBuffer, 1, Pos - 1)


                Do Until StringCommand = ""
                    ' check if more then one command before Carriage Return
                    If Mid(StringCommand, 1, 1) = "I" Then
                        executeLocSimCommand(Mid(StringCommand, 1, 4))
                        If StringCommand.Length <= 4 Then
                            StringCommand = ""
                        Else
                            StringCommand = Mid(StringCommand, 5)
                        End If
                    ElseIf Mid(StringCommand, 1, 1) = "X" Then
                        executeLocSimCommand(Mid(StringCommand, 1, 9))
                        If StringCommand.Length <= 9 Then
                            StringCommand = ""
                        Else
                            StringCommand = Mid(StringCommand, 10)
                        End If
                    Else
                        MsgBox("Command " & StringCommand & " receivd from LocSim; probably
wrong...")
                    End If

                Loop
```

```
            ' remove the command from the SimulatorStringBuffer
            If Pos = SimulatorStringBuffer.Length Then
                SimulatorStringBuffer = ""
            Else
                SimulatorStringBuffer = Mid(SimulatorStringBuffer, Pos + 1)
            End If

            Pos = InStr(SimulatorStringBuffer, Chr(13))
        Loop
    End If




End Sub
```

BoBo (420) MicroController

RS232

Asynchron

```
+--------------------+---------------+--------------+
|  Befehl (2 bytes)  | Wert (1 byte) |  0 (2 bytes) |
+--------------------+---------------+--------------+
```

ZuSi