SUB-20 .NET component. Revision 1.3

The SUB-20 .NET component is a special executable built in form of dynamic link library (sub20dnc.dll) which provides a programmable interface that is accessed by client .NET based applications. At the same time this module is a wrapper over the main SUB-20 Windows interface library (sub20.dll) and, therefore, almost all the methods correspond to appropriate functions in the main interface library. This section describes the SUB-20 .NET interface. In order to avoid doubling the functional description we provide a brief description and a link to the appropriate interface library function. All the interface classes are exported under "Xdimax" namespace.

.NET component reference:

Before using the SUB-20 .NET component you have to add a reference to it into your development environment. For VB.NET and C# go to Project->Add Reference->Browse select sub20dnc.dll, For VB6 go to the Project->References->Browse, select sub20dnc.tlb, For the MS Excel go to the VB Editor->Tools->References->Browse, select sub20dnc.tlb

The sub20dnc.tlb file can be produced by using the Microsoft regasm.exe utility. This utility located in the Microsoft .NET folder, usually \WINDOWS\Microsoft.NET\Framework\vXXXXX or for a 64-bit OS \WINDOWS\Microsoft.NET\Framework64\vXXXXX. Use /tlb and /codebase switches i.e open a command window and execute the following command

```
C:\>C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\RegAsm.exe /codebase
/tlb "C:\Program Files\SUB-20\bin\sub20dnc.dll"

Microsoft (R) .NET Framework Assembly Registration Utility
2.0.50727.1433
Copyright (C) Microsoft Corporation 1998-2004.  All rights reserved.

Types registered successfully
Assembly exported to 'C:\Program Files\SUB-20\bin\sub20dnc.tlb', and
the type library was registered successfully
C:\> _
```

SUB-20 .NET Classes

Some development environments, like NI LabView require using reference to an array for output parameters. In such cases use the "_r" version of SUB20.NET methods: I2C_Read_r, SPI_Transfer_r etc.

Class Sub20Enum

Used to enumerate all available SUB-20 devices

Public Methods:

Boolean SetDebugLevel(Integer Level)

Sets debug level

Parameters:

Integer Level - New debug level

Return Value:

Returns true if successful, false otherwise.

VB.NET example:

```
Private Sub Example()
      Dim DevEnum As New Sub20Enum
      DevEnum.SetDebugLevel(8)
End Sub
```

See also:
sub_set_debug_level

Long GetNext(Long Ref)

Scans USB devices currently connected to host looking for SUB-20 device.

Parameters:
Long Ref - SUB-20 device connection reference

Return Value:
Returns next found device reference

VB.NET example:

```
Private Function FindDeviceBySerNum(ByVal SerNum As String) As Long

      Dim DevEnum As New Sub20Enum
      Dim Dev As New Sub20
      Dim Ref As Long
      Dim SN As String

      Ref = 0
      Do While True
            Ref = DevEnum.GetNext(Ref)
            If Ref = 0 Then
                  Exit Do
            End If
            If Not Dev.Open(Ref) Then
                  Exit Do
            End If
            SN = Dev.GetSerialNumber()
            Dev.Close()
            If SN = SerNum Then
                  Exit Do
            End If
      Loop
      Return Ref
End Function
```

See also:
sub_find_devices

Class Sub20

Used to control sub-20 devices

Public Methods:

Boolean Open(Long Ref)

Opens SUB-20 device for further access.

Parameters:
Long Ref - SUB-20 device connection reference, returned by the Sub20Enum.GetNext method. If this parameter is 0 method will try to open first available SUB-20 device.

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:

```
Private Sub Example()
      Dim Dev As New Sub20

      ' Open first available SUB-20 device
      If Not dev.Open(0) Then
            MsgBox(Dev.GetStrError(Dev.GetLastError()))
            Exit Sub
      End If

      ...
      ...

      ' Close the device
      Dev.Close()
End Sub
```

See also:
sub_open
Sub20Enum.GetNext

Boolean Close( )

Closes access to the SUB-20 device

Parameters:
None

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
See example for the “Open” method

See also:
sub_close

String GetStrError(Integer ErrNum)

Return string describing an error

Parameters:
Integer ErrNum - Error number

Return Value:
Return error description

VB.NET example:
See example for the “Open” method

See also:
sub_strerror

String GetName()

Returns SUB-20 product ID string descriptor

Parameters:
None

Return Value:
Product ID string descriptor

VB.NET example:

```
Private Sub AcquireVersionInfo()
      Dim Dev As New Sub20
      Dim Name As String
      Dim FwVersion As String
      Dim BlVersion As String
      Dim DrvVersion As String
      Dim DllVersion As String
      ' Open first available SUB-20 device
      If Not Dev.Open(0) Then
            MsgBox(Dev.GetStrError(Dev.GetLastError()))
            Exit Sub
      End If

      Name = Dev.GetName()
      FwVersion = Dev.GetFwVersion()
      DrvVersion = Dev.GetDriverVersion()
      BlVersion = Dev.GetBlVersion()
      DllVersion = Dev.GetDllVersion()
      ...
      ...
      ' Close the device
      Dev.Close()
End Sub
```

See also:
sub_get_product_id

String GetSerialNumber()

Returns serial number string descriptor

Parameters:
None

Return Value:
Serial number string descriptor

VB.NET example:
See example for the “Sub20Enum.GetNext” method

See also:
sub_get_serial_number

String GetFwVersion()

Returns SUB-20 firmware version number

Parameters:
None

Return Value:
SUB-20 firmware version number

VB.NET example:
See example for the “GetName” method

See also:
sub_get_version

String GetBlVersion()

Returns SUB-20 boot loader version number

Parameters:
None

Return Value:
SUB-20 boot loader version number

VB.NET example:
See example for the “GetName” method

See also:
sub_get_version

String GetDllVersion()

Returns SUB-20 interface library version number

Parameters:
None

Return Value:
SUB-20 interface library version number

VB.NET example:
See example for the “GetName” method

See also:
sub_get_version

String GetDriverVersion()

Returns SUB-20 USB driver version number

Parameters:
None

Return Value:
SUB-20 USB driver version number

VB.NET example:
See example for the “GetName” method

See also:
sub_get_version

I2C Methods

Boolean I2C_SetConfig(Integer SlaveAddress, Integer Flags)

Configure SUB-20 I2C module.

Parameters:
Integer SlaveAddress – slave address for SUB-20 in I2C slave mode.
Integer Flags – flags. Can be set to I2C.I2C_GCE.

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:

See also:
sub_i2c_config

Boolean I2C_SetFrequency(ByRef Integer Frequency)

Sets SUB-20 I2C clock frequency.

Parameters:
Integer Frequency - On input - desired I2C clock frequency in Hz, on output - actual frequency

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
See example for the “I2C_Scan” method

See also:
sub_i2c_freq

Boolean I2C_GetFrequency(ByRef Integer Frequency)

Returns current SUB-20 I2C clock frequency.

Parameters:
Integer Frequency - On Output - current I2C clock frequency

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

See also:
sub_i2c_freq

Boolean I2C_Scan(Array Buffer, ByRef Integer SlaveCnt)

Scans I2C bus looking for connected slave devices

Parameters:
Array Buffer - The length of the array must be at least 128
On output - contains i2c slave addresses on which an acknowledgement has been received.
Integer SlaveCnt - On output - number of found i2c slave addresses

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:

```
Private Sub I2CEnumSlaves()
        Dim Dev As New Sub20
        Dim SlavesArry(127) As Byte
        Dim SlaveCntr As Integer
        ' Open SUB-20 device
        If Not Dev.Open(0) Then
            GoTo Done
        End If
        ' Set I2C clock frequency to 400kHz
        If Not Dev.I2C_SetFrequency(400000) Then
            GoTo Done
        End If
        ' Scan I2C bus for slaves
        If Not Dev.I2C_Scan(SlavesArry, SlaveCntr) Then
            GoTo Done
        End If
        If SlaveCntr > 0 Then
            For i = 0 To SlaveCntr - 1
                ' Adding all the slaves to a list
                AddToList(SlavesArry(i).ToString()
            Next
```

```
        End If
Done:
        If Dev.GetLastError() > 0 Then
            MsgBox(Dev.GetStrError(Dev.GetLastError()))
        End If
        ' Close SUB-20 device
        Dev.Close()
    End Sub
End Sub
```

See also:
sub_i2c_scan

Boolean I2C_IssueStart( )

Generates I2C start condition

Parameters:
None

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

See also:
sub_i2c_start

Boolean I2C_IssueStop( )

Generates I2C stop condition

Parameters:
None

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

See also:
sub_i2c_stop

Boolean I2C_Write(
Integer SlaveAddress,
Integer MemoryAddress,
Integer MemoryAddressSize,
Array Buffer)

Performs complete I2C write transaction with optional memory address write

Parameters:
Integer SlaveAddress - I2C slave address

Integer MemoryAddress - Memory address
Integer MemoryAddressSize - Memory address size
Array Buffer - Array containing data to be written

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:

```
Private Sub I2CReadWrite()
        Dim Dev As New Sub20
        Dim Data(1) As Byte
        ' Open SUB-20 device
        If Not Dev.Open(0) Then
            GoTo Done
        End If
        ' Set I2C clock frequency to 400kHz
        If Not Dev.I2C_SetFrequency(400000) Then
            GoTo Done
        End If
        Data(0) = &H55
        Data(1) = &HAA
        ' Write 2 bytes to slave device on address 2a
        If Not Dev.I2C_Write(&H2A, 0, 0, Data) Then
            GoTo Done
        End If
        ' Read 2 bytes from slave device on address 2a
        If Not Dev.I2C_Read(&H2A, 0, 0, Data) Then
            GoTo Done
        End If
Done:
        If Dev.GetLastError() > 0 Then
            MsgBox(Dev.GetStrError(Dev.GetLastError()) + _
                   " " + Dev.I2C_GetStatus().ToString())
        End If
        ' Close SUB-20 device
        Dev.Close()

End Sub
```

See also:
sub_i2c_write

Boolean I2C_Read(
Integer SlaveAddress,
Integer MemoryAddress,
Integer MemoryAddressSize,
Array Buffer)

Performs complete I2C read transaction with optional memory address write

Parameters:
Integer SlaveAddress - I2C slave address
Integer MemoryAddress - Memory address
Integer MemoryAddressSize - Memory address size
Array Buffer - Array to store read data

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
See example for the “I2C_Write” method

See also:
sub_i2c_read

Integer I2C_GetStatus( )

Returns status of last I2C operation

Parameters:
None

Return Value:
status of last I2C operation

VB.NET example:
See example for the “I2C_Write” method

See also:
sub_i2c_status

I2C Bit-bang Methods

Boolean I2C_BB_SetConfig(
Integer Mode,
Integer Stretch_ms)

Configure SUB-20 I2C Bit-bang module.

Parameters:
Mode – Bit-bang I2C Master mode. One of the following:

BB_I2C.FastPlus for 1000 KHz
BB_I2C.Fast for 400 KHz
BB_I2C.Std for 100 KHz

Stretch_ms – Clock stretching timeout up to 4194ms. Clock stretching feature is available in Fast and Standard modes. If SCL line is hold LOW by I2C slave device longer than stretch_ms timeout, Bit-bang I2C function will fail and I2C Status will be set to 0xE0.

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
See *I2C_SetConfig example*

See also:
sub_bb_i2c_config
I2C_SetConfig

Boolean I2C_BB_Scan(Integer Channel, Array Buffer, ByRef Integer SlaveCnt)

Scans I2C bus looking for connected slave devices

Parameters:
Integer *Channel - Bit-bang I2C Master channel 0..3*
Array Buffer - The length of the array must be at least 128
On output - contains i2c slave addresses on which an acknowledgement has been received.
Integer SlaveCnt - On output - number of found i2c slave addresses

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
See *I2C_Scan example*

See also:
I2C_Scan

Boolean I2C_BB_Write(Integer Channel, Integer SlaveAddress, Integer MemoryAddress, Integer MemoryAddressSize, Array Buffer)

Performs complete I2C write transaction with optional memory address write

Parameters:
Integer *Channel - Bit-bang I2C Master channel 0..3*
Integer SlaveAddress - I2C slave address
Integer MemoryAddress - Memory address
Integer MemoryAddressSize - Memory address size
Array Buffer - Array containing data to be written

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
See *I2C_Write example*

See also:
I2C_Write

Boolean I2C_BB_Read(
Integer Channel,
Integer SlaveAddress,
Integer MemoryAddress,
Integer MemoryAddressSize,
Array Buffer)

Performs complete I2C read transaction with optional memory address write

Parameters:
Integer *Channel - Bit-bang I2C Master channel 0..3*
Integer SlaveAddress - I2C slave address
Integer MemoryAddress - Memory address
Integer MemoryAddressSize - Memory address size
Array Buffer - Array to store read data

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
See example for the “I2C_Write” method

See also:
I2C_Read

LCD Methods

Boolean LCD_Write(String Str)

Writes control sequence to the on-board LCD

Parameters:
String Str - Control String

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:

```
Private Sub LCD_Test ()
      Dim Dev As New Sub20
      ' Open first available SUB-20 device
      If Not Dev.Open(0) Then
            GoTo Done
      End If
      ' Write to the LCD
      If Not Dev.LCD_Write("\fHello\nWorld") Then
            GoTo Done
      End If
Done:
      If Dev.GetLastError() > 0 Then
            MsgBox(Dev.GetStrError(Dev.GetLastError()))
      End If
      ' Close SUB-20 device
      Dev.Close()
End Sub
```

See also:
sub_lcd_write

GPIO/GPIOB Methods

Boolean GPIO_GetConfig(ByRef Unsigned Integer Config)
Boolean GPIO_GetConfig_i(ByRef Integer Config)
Boolean GPIOB_GetConfig(ByRef Unsigned Integer Config)
Boolean GPIOB_GetConfig_i(ByRef Integer Config)
Reads current GPIO configuration

Parameters:
Integer Config - On output - current GPIO configuration

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

See also:
sub_gpio_config

Boolean GPIO_SetConfig(Unsigned Integer Config, Unsigned Integer Mask)
Boolean GPIO_SetConfig_i(Integer Config, Integer Mask)
Boolean GPIOB_SetConfig(Unsigned Integer Config, Unsigned Integer Mask)
Boolean GPIOB_SetConfig_i(Integer Config, Integer Mask)
Configures GPIO

Parameters:
Integer Config - Input/Output bit mask
Integer Mask - Bit in the "Input/Output bit mask" parameter will take effect only if corresponding mask bit is "1".

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
See example for the “GPIO_Read” method

See also:
sub_gpio_config

Boolean GPIO_Write(Unsigned Integer Value, Unsigned Integer Mask)
Boolean GPIO_Write_i(Integer Value, Integer Mask)
Boolean GPIOB_Write(Unsigned Integer Value, Unsigned Integer Mask)
Boolean GPIOB_Write_i(Integer Value, Integer Mask)
Set GPIO status

Parameters:
Integer Value - Output Value/Pullup bitmask
Integer Mask - Bit in the "Input/Output bit mask" parameter will take effect only if corresponding mask bit is "1".

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
See example for the “GPIO_Read” method

See also:
sub_gpio_write

Boolean GPIO_Read(ByRef Unsigned Integer Value)
Boolean GPIO_Read_i(ByRef Integer Value)
Boolean GPIOB_Read(ByRef Unsigned Integer Value)
Boolean GPIOB_Read_i(ByRef Integer Value)
Reads GPIO input status.

Parameters:
Integer Value
On output - GPIO input status

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:

```
Private Sub GPIO_Test()
      Dim Dev As New Sub20
      Dim OutMask, InMask, Tmp As Long

      ' Open first available SUB-20 device
      If Not Dev.Open(0) Then
            GoTo Done
      End If
      ' Set GPIO IN and OUT bitmasks
      OutMask = 2 ^ 14    'GPIO14 bitmask
      InMask = 2 ^ 15     'GPIO15 bitmask

      ' Configure GPIO14 as an output and GPIO15 as an input
      If Not Dev.GPIO_SetConfig(OutMask, OutMask Or InMask) Then
            GoTo Done
      End If

      ' Enable a weak pull-up resistor on the GPIO15
      If Not Dev.GPIO_Write(InMask, InMask) Then
            GoTo Done
      End If

      ' Produce a positive pulse on the GPIO14

      ' Set GPIO14 to "0"
      If Not Dev.GPIO_Write(0, OutMask) Then
            GoTo Done
      End If

      ' Set GPIO14 to "1"
      If Not Dev.GPIO_Write(OutMask, OutMask) Then
            GoTo Done
      End If

      ' Set GPIO14 to "0"
      If Not Dev.GPIO_Write(0, OutMask) Then
            GoTo Done
      End If

      ' Read all the GPIOs
      If Not Dev.GPIO_Read(Tmp) Then
            GoTo Done
      End If

      ' Mask out the GPIO14
      If Tmp And InMask Then

            ' GPIO14 is "1"
            ...
            ...
      Else
            ' GPIO14 is "0"
            ...
            ...
      End If
```

```
Done:
      If Dev.GetLastError() > 0 Then
            MsgBox(Dev.GetStrError(Dev.GetLastError()))
      End If
      ' Close SUB-20 device
      Dev.Close()
End Sub
```

See also:
sub_gpio_read

SPI Methods

Boolean SPI_GetConfig(ByRef Integer Config)
Returns current SPI configuration

Parameters:
Integer Config - On output - combination of the XDimax.Spi constants

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

See also:
sub_spi_config
XDimax.Spi

Boolean SPI_SetConfig(Integer Config)
Sets SPI configuration

Parameters:
Integer Config - Combination of the XDimax.Spi constants

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:

```
Private Sub SPI_Test()
      Dim Dev As New Sub20
      Dim InOutData(2) As Byte

      ' Open first available SUB-20 device
      If Not Dev.Open(0) Then
            GoTo Done
      End If
      ' Enable and configure SPI
```

```
      If Not Dev.SPI_SetConfig(Spi.Enable + Spi.CpolRise _
            + Spi.SmplSetup + Spi.MsbFirst + Spi.Clk_4MHz) Then
            GoTo Done
      End If

      ' Output data bytes
      InOutData(0) = &H2A
      InOutData(1) = &H2B
      InOutData(2) = &H2C

      ' Exchange data with SPI slave(SS=2). SS signal stays high
      If Not Dev.SPI_Transfer(InOutData, 2, Spi.Ss_H) Then
            GoTo Done
      End If

      ' Process data received from slave into the InOutData
      ...
      ...

Done:
      If Dev.GetLastError() > 0 Then
            MsgBox(Dev.GetStrError(Dev.GetLastError()))
      End If
      ' Close SUB-20 device
      Dev.Close()
End Sub
```

See also:
sub_spi_config
XDimax.Spi
SPI_LowSpeedClk

Boolean SPI_LowSpeedClk(Integer ClkFreq)

Returns configuration value corresponding to the ClkFreq – low speed clock frequency

Parameters:
Integer ClkFreq – Desired low speed clock frequency in Hz. Valid values are 4000-250000.

Return Value:
Returns configuration value corresponding to the ClkFreq. This value can be combined with other configuration flags prior calling SPI_SetConfig method.

VB.NET example:

```
      ' setting SPI clock to 5kHz(low speed mode)
      If Not Dev.SPI_SetConfig(Spi.Enable + Spi.CpolRise _
            + Spi.SmplSetup + Spi.MsbFirst + Dev.SPI_LowSpeedClk(5000)
      ) Then
            GoTo Done
      End If
```

See also:
sub_spi_config

Boolean SPI_Write(Array Buffer, Integer Ss_Pin, Integer Ss_Mode)
Performs SPI master transaction. Read data discarded

Parameters:
Array Buffer - Array containing data to be written
Integer Ss_Pin - SS pin number
Integer Ss_Mode - One of the XDimax.Spi.Ss_ constants

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
See example for the “SPI_SetConfig” method

See also:
sub_spi_transfer
XDimax.Spi

Boolean SPI_Read(Array Buffer, Integer Ss_Pin, Integer Ss_Mode)
Performs SPI master transaction.

Parameters:
Array Buffer - Array to store write and read data
Integer Ss_Pin - SS pin number
Integer Ss_Mode - One of the XDimax.Spi.Ss_ constants

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
See example for the “SPI_SetConfig” method

See also:
sub_spi_transfer
XDimax.Spi

Boolean SPI_Transfer(Array Buffer, Integer Ss_Pin, Integer Ss_Mode)
Performs SPI master transaction.

Parameters:
Array Buffer - Array to store write and read data
Integer Ss_Pin - SS pin number
Integer Ss_Mode - One of the XDimax.Spi.Ss_ constants

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
See example for the “SPI_SetConfig” method

See also:
sub_spi_transfer
XDimax.Spi

RS232/485 Methods

Boolean RS_SetConfig(Integer Config, Integer Baudrate)
Configures SUB-20 UART (Universal Asynchronous Receiver Transmitter).

Parameters:
Integer Config - UART configuration. This value is a combination of the XDimax.Rs constants
Integer Baudrate - Desired baudrate

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:

```
Private Sub Rs_Test()
      Dim Dev As New Sub20
      Dim OutData(2) As Byte
      Dim InData(4) As Byte


      ' Open first available SUB-20 device
      If Not Dev.Open(0) Then
            GoTo Done
      End If

      ' Set 9660 bps, 8 data bits, no parity, 1 stop bit
      If Not Dev.RS_SetConfig(Rs.RxEnable Or Rs.TxEnable _
                              Or Rs.Char8 Or Rs.ParityNone _
                              Or Rs.Stop1, 9600) Then
            GoTo Done
      End If

      ' Request message transmit and after that receipt
      ' No space between transmitted bytes
      ' Message should be received in 1s
      If Not Dev.RS_SetTiming(Rs.RxAfterTx, 0, 1000000, 0) Then
            GoTo Done
      End If

      ' Transmit 3 bytes and try to receive up to 5
      ' bytes in 1s with 200ms byte to byte timeout
      OutData(0) = &H64
      OutData(1) = &H65
      OutData(2) = &H66

      If Not Dev.RS_Transfer(OutData, InData) Then
```

```
            GoTo Done
      End If
Done:
      If Dev.GetLastError() > 0 Then
            MsgBox(Dev.GetStrError(Dev.GetLastError()))
      End If
      ' Close SUB-20 device
      Dev.Close()
End Sub
```

See also:
sub_rs_set_config
XDimax.Rs

Boolean RS_GetConfig(ByRef Integer Config, ByRef Integer Baudrate)

Reads current SUB-20 UART configuration

Parameters:
Integer Config - On output - UART configuration, combination of the XDimax.Rs constants
Integer Baudrate - On output - Actual baudrate

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

See also:
sub_rs_get_config
XDimax.Rs

Boolean RS_SetTiming(Integer Flags, Integer Tx_space_us, Integer Rx_msg_us, Integer Rx_byte_us)

Configures UART transfer timing and order of transmit and receive operations.

Parameters:
Integer Flags - Integer Tx_space_us
Integer Rx_msg_us - Integer Rx_byte_us

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
See example for the “RS_SetConfig” method

See also:
sub_rs_timing

Boolean RS_Write(Array Buffer, ByRef int Transferred*)*

Transmits message(s) via SUB-20 UART

Parameters:
Array Buffer - Array containing data to be written
Transferred – On outout, number of transferred bytes

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
Success=Dev.RS_Write(Data)

See also:
sub_rs_xfer

Boolean RS_Read(Array Buffer, ByRef int Transferred*)*

Receives message(s) via SUB-20 UART

Parameters:
Array Buffer - Array to store read data
Transferred – On outout, number of transferred bytes

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
Success=Dev.RS_Read(Data)

See also:
sub_rs_xfer

Boolean RS_Transfer(Array OutBuffer, Array InBuffer, ByRef int Transferred*)*

Transmits and receives message(s) via SUB-20 UART

Parameters:
Array OutBuffer - Array containing data to be written
Array InBuffer - Array to store read data
Transferred – On outout, number of transferred bytes

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
See example for the “RS_SetConfig” method

See also:
sub_rs_xfer

FIFO Methods

Boolean FIFO_SetConfig(Integer Config)
Configures SUB-20 for fifo operations

Parameters:
Integer Config - FIFO configuration. Combination of the XDimax.Fifo constants

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
See example for the “FIFO_Read” method

See also:
sub_fifo_config

Boolean FIFO_Write(Array Buffer, Integer Timeout_ms, ByRef int Transferred*)*
Attempts to transfer data into OUT FIFO in no more then Timeout_ms time.

Parameters:
Array Buffer - Array containing data to be written
Integer Timeout_ms - Timeout in milliseconds
Transferred – On outout, number of transferred bytes

VB.NET example:
See example for the “FIFO_Read” method

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

See also:
sub_fifo_write

Boolean FIFO_Read(Array Buffer, Integer Timeout_ms, ByRef Transferred)

Attempts to read data from IN FIFO into buffer in no more then Timeout_ms time.

Parameters:
Array Buffer - Array to store read data
Integer Timeout_ms - Timeout in milliseconds
Transferred – On outout, number of transferred bytes

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:

```
Private Sub FIFO_Test ()
      Dim Dev As New Sub20
      Dim Data(7) As Byte
      ' Open first available SUB-20 device
      If Not Dev.Open(0) Then
            GoTo Done
      End If
      'Configure FIFO
      If Not Dev.FIFO_SetConfig(Fifo.SelectSpi) Then
            GoTo Done
      End If
        ...
      'Attempt to read 8 data bytes from IN FIFO, timeout=1ms
      If Not Dev.FIFO_Read(Data, 1000) Then
            GoTo Done
      End If
        ...
      'Attempt to transfer 8 data bytes to the OUT FIFO, timeout=1ms
      If Not Dev.FIFO_Write(Data, 1000) Then
            GoTo Done
      End If

Done:
      If Dev.GetLastError() > 0 Then
            (Dev.GetStrError(Dev.GetLastError()))
      End If
      ' Close SUB-20 device
      Dev.Close()
End Sub
```

See also:
sub_fifo_read

FPWM Methods

Boolean FPWM_Config(Float Freq, Integer Flags)
Configures fast PWM module.

Parameters:
Float Freq - Desired fast PWM frequency in Hz
Integer Flags - FPWM Configuration. This value is a combination of the XDimax.Fpwm constants

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:

```
Private Sub FPWM_Test()
      Dim Dev As New Sub20

      ' Open first available SUB-20 device
      If Not Dev.Open(0) Then
            GoTo Done
      End If

      ' Enable fast PWM module with FPWM_0 and FPWM_2 outputs.
      ' PWM frequency 10.6Hz
      If Not Dev.FPWM_Config(10.6, Fpwm.Enable _
       Or Fpwm.En0 Or Fpwm.En2) Then
            GoTo Done
      End If

      ' Set duty cycle 25% for CH0
      If Not Dev.FPWM_SetDutyCycle(0, 25) Then
            GoTo Done
      End If

      ' Set duty cycle 12.5% for CH2
      If Not Dev.FPWM_SetDutyCycle(2, 12.5) Then
            GoTo Done
      End If
Done:
      If Dev.GetLastError() > 0 Then
            MsgBox(Dev.GetStrError(Dev.GetLastError()))
      End If
      ' Close SUB-20 device
      Dev.Close()
End Sub
```

See also:
sub_fpwm_config
XDimax.Fpwm

Boolean FPWM_SetDutyCycle(Integer OutputIndex, float DutyCycle)

Configure specific fast PWM output

Parameters:
Integer OutputIndex - FPWM output to configure. Can be 0,1,2
float DutyCycle - Desired duty cycle % in range 0..100

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
See example for the “FPWM_Config” method

See also:
sub_fpwm_set

PWM Methods

Boolean PWM_SetConfig(Integer Resolution_us, Integer Limit)
Configures PWM module.

Parameters:
Resolution_us – PWM module clock resolution in μs. Resolution range is 20μs - 16384μs.
Limit - PWM module counter limit in range 0-255. If limit is 0 PWM module will be turned off.

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:

```
Private Sub PWM_Test()
      Dim Dev As New Sub20
      Dim Mask As Long

      ' Open first available SUB-20 device
      If Not Dev.Open(0) Then
            GoTo Done
      End If

      ' Set PWM resolution=10ms, limit=100, frequency=1Hz
      If Not Dev.PWM_SetConfig(10000, 100) Then
            GoTo Done
      End If

      ' Set PWM_0 pin (GPIO24) to output state
      Mask = 2 ^ 24
      If Not Dev.GPIO_SetConfig(Mask, Mask) Then
            GoTo Done
      End If

      ' Output 50% duty cycle on PWM_0 pin
      If Not Dev.PWM_SetDutyCycle(0, 50) Then
            GoTo Done
      End If

Done:
      If Dev.GetLastError() > 0 Then
            MsgBox(Dev.GetStrError(Dev.GetLastError()))
      End If
      ' Close SUB-20 device
      Dev.Close()
End Sub
```

See also:
sub_pwm_config

Boolean PWM_SetDutyCycle(Integer OutputIndex, Integer DutyCycle)

Configures a PWM duty cycle for specified output channel.

Parameters:
OutputIndex – Index of PWM output to configure. Can be 0..7.
DutyCycle - Duty cycle in range 0..255

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
See example for the “PWM_SetConfig” method

ADC Methods

Boolean ADC_SetConfig(Integer Flags)
Configures SUB-20 ADC module.

Parameters:
Integer Flags - ADC configuartion. This value is a combinations of the XDimax.Adc.Enable, XDimax.Adc.RefVcc, XDimax.Adc.Ref256 constants

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:

```
Private Const SamplesToRead As Integer = 24
Private Sub ADC_Test()
      Dim dev As New Sub20
      Dim Data(SamplesToRead) As Integer
      Dim Mux(SamplesToRead) As Integer
      Dim i As Integer

      ' Open first available SUB-20 device
      If Not Dev.Open(0) Then
            GoTo Done
      End If

      ' Enable the ADC and set Vref=Vcc
      If Not dev.ADC_SetConfig(Xdimax.Adc.Enable _
            Or Xdimax.Adc.RefVcc) Then
            GoTo Done
      End If

      ' Setup the Mux to read all the channels, i.e
      ' Ch0,Ch1,Ch2...Ch7,Ch0,Ch1.. etc
      For i = 0 To Data.GetLength(0) - 1
            Mux(i) = i Mod 8
      Next
      ' Read samples
      If Not dev.ADC_Read(Data, Mux) Then
            GoTo Done
      End If
Done:
```

```
      If dev.GetLastError() > 0 Then
            MsgBox(dev.GetStrError(dev.GetLastError()))
      End If
      ' Close SUB-20 device
      dev.Close()
End Sub
```

See also:
sub_adc_config
XDimax.Adc

Boolean ADC_Single(ByRef Integer Data, Integer Mux)
Read single ADC conversion result

Parameters:
Integer Data - On output - conversion result
Integer Mux - ADC input channel multiplexer control code. See XDimax.Adc constants

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

See also:
sub_adc_single
XDimax.Adc

Boolean ADC_Read(Array Data, Array Mux)
Read multiple ADC conversion results

Parameters:
Array Data - Array of Integers to store conversion result
Array Mux - Array of Integers containing input channel multiplexer control codes. See XDimax.Adc constants

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

VB.NET example:
See example for the “ADC_SetConfig” method

See also:
sub_adc_read
XDimax.Adc

Integer GetLastError()
Returns status code of last operation

Parameters:
None

Return Value:
Status code

See also:
sub_errno

Edge Methods

Boolean Edge_SetConfig(Unsigned Integer Config)
Configures EDGE module.

Parameters:
Config value, see SUB-20 User manual for more information

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

Boolean Edge_GetConfig(ByRef Unsigned Integer Config)
Boolean Edge_GetConfig_i(ByRef Integer Config)

Returns current EDGE module configuration.

Parameters:
Config value, see SUB-20 User manual for more information

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

Boolean Edge_Read(ByRef Unsigned Integer Gpio, ByRef Unsigned Integer Edge)
Boolean Edge_Read_i(ByRef Integer Gpio, ByRef Integer Edge)

Reads GPIO input status and EDGE status.

Parameters:
Gpio – Gpio input status
Edge – Edge status
see SUB-20 User manual for more information

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

EEPROM Methods

Boolean EEP_Read(int Address, Array^ Buffer);

Reads data from the on-board EEPROM

Parameters:
Address –Start address
Buffer – Memory buffer to place the read data

see SUB-20 User manual for more information

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

Boolean EEP_Write(int Address, Array^ Buffer);

Writes data to the on-board EEPROM

Parameters:
Address –Start address
Buffer – Memory buffer of the data to be written

see SUB-20 User manual for more information

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

MDIO Methods

Boolean Mdio22_Write(Integer PhyAddr,
Integer RegAddr, Integer Data)

Generates IEEE 802.3 Clause 22 MDIO WRITE frame.

Parameters:

PhyAddr - 5-bit PHY address
RegAddr - 5-bit Register address
Data - 16-bit data

see SUB-20 User manual for more information

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

Boolean Mdio22_Read(Integer PhyAddr,
Integer RegAddr, ByRef Integer Data)

Generates IEEE 802.3 Clause 22 MDIO READ frame.

Parameters:

PhyAddr - 5-bit PHY address
RegAddr - 5-bit Register address
Data - placeholder to store 16-bit data

see SUB-20 User manual for more information

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

bool Mdio22(Integer Op, Integer PhyAddr, Integer RegAddr, Integer Data, ByRef Integer Content)

Generates IEEE 802.3 Clause 22 MDIO frame

Parameters:

Op - Op code, see Mdio22 constants
PhyAddr - 5-bit PHY Address
RegAddr - 5-bit Register address
Data - Data value
Content - Placeholder to store 16-bit read value

see SUB-20 User manual for more information

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

Boolean Mdio45_Address(Integer PortAddr,
Integer DevAddr, Integer Address)

Generates IEEE 802.3 Clause 45 MDIO ADDRESS frame.

Parameters:

PortAddr - 5-bit port address
DevAddr - 5-bit device address
Address - 16-bit Address value

see SUB-20 User manual for more information

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

Boolean Mdio45_Write(Integer PortAddr,
Integer DevAddr, Integer Data)

Generates IEEE 802.3 Clause 45 MDIO WRITE frame.

Parameters:

PortAddr - 5-bit port address
DevAddr - 5-bit device address
Data - 16-bit Data value

see SUB-20 User manual for more information

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

Boolean Mdio45_Pria(Integer PortAddr,
Integer DevAddr, ByRef Integer Data)

Generates IEEE 802.3 Clause 45 MDIO POST-READ-INCREMENT-ADDRESS frame.

Parameters:

PortAddr - 5-bit port address
DevAddr - 5-bit device address
Data - Placeholder to store 16-bit Data value

see SUB-20 User manual for more information

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

Boolean Mdio45_Read(Integer PortAddr,
Integer DevAddr, ByRef Integer Data)

Generates IEEE 802.3 Clause 45 MDIO READ frame.

Parameters:

PortAddr - 5-bit port address
DevAddr - 5-bit device address
Data - Placeholder to store 16-bit Data value

see SUB-20 User manual for more information

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

bool Mdio45(Integer Op, Integer PortAddr,
Integer DevAddr, Integer Data, ByRef Integer Content)

Generates IEEE 802.3 Clause 45 MDIO frame

Parameters:

Op - Op code, see Mdio45 constants
PortAddr - *5-bit* Port Address
DevAddr - *5-bit Device address*
Data - *16-bit Data Value*
Content - *Placeholder to store 16-bit read value*

see SUB-20 User manual for more information

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

bool MdioTransferEx(int Channel, ByRef Array Frames)
bool MdioTransferEx_r(int Channel, ByRef Array Frames)

Generate a sequence of independent MDIO frames. Frames in sequence can be Clause 22 or Clause 45 format with different operations and addresses.

Parameters:

Channel - MDIO Channel, Optionally can be combined(bitwise OR) with MdioCfpMsa flag to generate CFP MSA compatible transaction(s) at 4MHz MDC frequency

Frames - *Array of Integers describing MDIO Frames to be generated*

The "Frames" argument is an array of Integers. Each MDIO frame can be described by four consequent Integer values. For Clause 22 these values defined as the following

Frames[4 * frame# + 0] - Op code, see Mdio22 constants
Frames[4 * frame# + 1] - PHY Address
Frames[4 * frame# + 2] - Register Address
Frames[4 * frame# + 3] - Data value

For Clause 45 these values defined as the following

Frames[4 * frame# + 0] - Op code, see Mdio45 constants
Frames[4 * frame# + 1] - Port Address
Frames[4 * frame# + 2] - Device Address
Frames[4 * frame# + 3] - Data value

Thus the number of Integers in the "Frames" Array must be multiple of 4. For example, to describe 10 MDIO frames the array length must be 10 * 4 = 40 Integers

see SUB-20 User manual for more information

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

Handle Methods

Boolean GetHandle(ByRef Unsigned Integer Handle)

Boolean GetHandle_i(ByRef Integer Handle)

Returns low level device handle

Parameters:
Handle – Low level device handle.

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

Boolean CloseHandle(Unsigned Integer Handle)
Boolean CloseHandle_i(Integer Handle)

Closes a device handle

Parameters:
Handle – Low level device handle.

Return Value:
Returns true if successful, false otherwise. Call GetLastError method to get an extended error information

Class Error

Error constants

Constants:
Ok=SE_OK,
NoDev=SE_NODEV,
Open=SE_OPEN,
SetConf=SE_SETCONF,
Claim=SE_CLAIM,
Seta=SE_SETA,
Submit=SE_SUBMIT,
BulkOut=SE_BULKOUT,
BulkIn=SE_BULKIN_RC,
BulkInSz=SE_BULKIN_SZ,
OutOvf=SE_OUT_OVF,
I2C=SE_I2C,
TagCode=SE_TAG_CODE,
TagSize=SE_TAG_SIZE,
Param=SE_PARAM,
SpiDis=SE_SPI_DIS,
NotSupppported=SE_NSUPPORTED,
Timeout=SE_TIMEOUT,
Init=SE_INIT,

Class I2C

I2C related constants

Constants:
Gce=I2C_GCE

Class Spi

SPI related constants

Constants:
CpolRise=SPI_CPOL_RISE,
CpolFall=SPI_CPOL_FALL,
SmplSetup=SPI_SMPL_SETUP,
SetupSmpl=SPI_SETUP_SMPL,
Enable=SPI_ENABLE,
Slave=SPI_SLAVE,
LsbFirst=SPI_LSB_FIRST,
MsbFirst=SPI_MSB_FIRST,
Clk_8MHz=SPI_CLK_8MHZ,
Clk_4MHz=SPI_CLK_4MHZ,
Clk_2MHz=SPI_CLK_2MHZ,
Clk_1MHz=SPI_CLK_1MHZ,
Clk_500kHz=SPI_CLK_500KHZ,
Clk_250kHz=SPI_CLK_500KHZ,
Clk_125kHz=SPI_CLK_125KHZ,
Ss_H=SS_H,
Ss_HL=SS_HL,
Ss_HHL=SS_HHL,
Ss_HHHL=SS_HHHL,
Ss_HHHHL=SS_HHHHL,
Ss_LH=SS_LH,
Ss_LLH=SS_LLH,
Ss_LLLH=SS_LLLH,
Ss_LLLLH=SS_LLLLH,
Ss_LO=SS_LO,
Ss_HiZ=SS_HiZ

Class Rs

RS232/485 related constants

Constants:
RxEnable=RS_RX_ENABLE,
TxEnable=RS_TX_ENABLE,
Char5=RS_CHAR_5,
Char6=RS_CHAR_6,
Char7=RS_CHAR_7,
Char8=RS_CHAR_8,
Char9=RS_CHAR_9,
ParityNone=RS_PARITY_NONE,
ParityEven=RS_PARITY_EVEN,
ParityOdd=RS_PARITY_ODD,

Stop1=RS_STOP_1,
Stop2=RS_STOP_2,

RxBeforeTx=RS_RX_BEFORE_TX,
RxAfterTx=RS_RX_AFTER_TX

Class Fifo

FIFO related constants

Constants:
SelectSpi=FIFO_SELECT_SPI,
SelectUart=FIFO_SELECT_UART,
Clear=FIFO_CLEAR

Class Fpwm

Description:
FPWM related constants

Constants:
Enable=FPWM_ENABLE,
En0=FPWM_EN0,
En1=FPWM_EN1,
En2=FPWM_EN2

Class Adc

ADC related constants

Constants:

Enable=ADC_ENABLE,
RefVcc=ADC_REF_VCC,
Ref256=ADC_REF_2_56,
S0=ADC_S0,
S1=ADC_S1,
S2=ADC_S2,
S3=ADC_S3,
S4=ADC_S4,
S5=ADC_S5,
S6=ADC_S6,
S7=ADC_S7,
D10_10X=ADC_D10_10X,
D10_200X=ADC_D10_200X,
D32_10X=ADC_D32_10X,
D32_200X=ADC_D32_200X,
D01=ADC_D01,
D21=ADC_D21,
D31=ADC_D31,
D41=ADC_D41,
D51=ADC_D51,
D61=ADC_D61,

D71=ADC_D71,
D02=ADC_D02,
D12=ADC_D12,
D32=ADC_D32,
D42=ADC_D42,
D52=ADC_D52

Class BB_I2C

Description:
I2C Bit-bang related constants
BB_I2C.FastPlus, for 1000 KHz
BB_I2C.Fast, for 400 KHz
BB_I2C.Std, for 100 KHz

Class MDIO

Mdio22Read=SUB_MDIO22_READ,
Mdio22Write=SUB_MDIO22_WRITE,
Mdio45Addr=SUB_MDIO45_ADDR,
Mdio45Write=SUB_MDIO45_WRITE,
Mdio45Pria=SUB_MDIO45_PRIA,
Mdio45Read=SUB_MDIO45_READ,
MdioCfpMsa=SUB_CFP_MSA

Class SystemByteArray

Used as a container of the System.Array to use in VB6 and VBA application

Public Properties:
System.Array array

Public Methods:

void CreateInstance(Integer Length)

Initializes a new instance of the System.Array class

Parameters:
Integer Length
Length of the array

Return Value:
None

See also:
System::Array.CreateInstance

Byte GetValue(Integer Index)
Gets the value of the specified element in the current Array

Parameters:
Integer Index - Zero based element's index

Return Value:
Element's Value

See also:
System::Array.GetValue

Boolean SetValue(Byte Value, Integer Index)

Sets the specified element in the current Array to the specified value.

Parameters:
Byte Value - Value to be set
Integer Index - Zero based element's index

Return Value:
True in success, false otherwise

See also:
System::Array.SetValue