The SPECTROTILT™ Dyna – Single Axis Dynamic Electronic Inclinometer, p/n SSY0247-HDS, can be equipped with one of two communication protocols, 9 bit addressing mode or Modbus. It is possible to communicate via a standard RS232 port by use of an external RS232 to RS485 converter module.

**Memory Map**

- 2 reserved
- 4 serial number
- 6 offset for sensor
- 8 reserved
- 10 offset for temp sensor
- 12 reserved
- 14 reserved
- 16 module’s address
- 18 reserved
- 20 offset for pwm output
- 22 baud rate
  - 38400 - 7
  - 28800 - 10
  - 19200 - 15
  - 9600 - 32
  - 4800 - 64
- 100 .. 280 reserved

**1st Protocol**

**Data format:**

- Baud rate* (see memory map, 22)
- Data width = 8 bits plus one bit to indicate data or address (see Visual basic code)
- 1 stop bit
- Module address = 0 (factory set)

**Commands**

Should be sent in (ASCII) format

- 0 (48) // Reset processor
- 3 (51) // read eeprom
- A (65) // read sensor
- C (67) // read temperature
- E (69) // write to eeprom
- F (70) // Sends firmware revision
All commands sent to the unit end with ‘carriage return’ and ‘line feed’. After a command, the unit returns with ‘carriage return’ and ‘line feed’.

EXAMPLES:

‘Reset the Processor’

00 address bit is set
48 cmd reset processor
cr carriage return
lf line feed

‘Read Sensor Data’

00 address module
65 cmd ‘A’
cr
lf

‘Unit Returns 1001’

10
01
cr
lf

‘Write to EEPROM’

00 address
69 cmd ‘E’
34 address 34 note the address has to even number
44 delimits address and data can be a space or ‘;’ or ‘:’ or tab
25
01 write 2501 in memory
cr
lf
Communicating via the RS232 port of PC  (RS232 to RS485 converter required)

The address for the SPECTROTILT™ Dyna is in nine bit format. As RS232 is an eight bit format, we use the parity bit as the missing ninth bit, toggling it 'on' when sending the unit address, and 'off' when sending data. The following is an example of the code in Visual Basic.

**RS232 Communication - Code Example**

' Set serial port
mhSerialPort.Open(Port, 28800, 8, Rs232.DataParity.Parity_None, 1, 200)

' Function: Open
' Parameter: ByVal Port As String
'            ByVal InitStr As String
' Return:    Boolean
'
' Purpose: Open serial port and display message on the main status bar

Public Function Open(ByVal Port As String, ByVal BaudRate As Integer, ByVal DataBit As Integer, ByVal Parity As Rs232.DataParity, ByVal StopBit As Rs232.DataStopBit) As Boolean
Try
  mhSerialPort.Port = getPortNum(Port)
  mhSerialPort.BaudRate = BaudRate
  mhSerialPort.DataBit = DataBit
  mhSerialPort.Parity = Parity
  mhSerialPort.StopBit = StopBit
  mhSerialPort.BufferSize = 4095
  Open = Open()
Catch ex As Exception
  LastMsg = "Unable to connect to Serial Port"
  Return False
End Try
End Function

' The buf() contains only the data the address is defined by bAddress
' bAddress can be between 0 and 255

Public Sub Write(ByVal buf() As Byte, ByVal ilen As Integer)
  Dim bByte As Byte
  Dim addr() As Byte = {bAddress}
  Dim i As Integer
  Try
    ' Send address
    If (cBit.CountBitOn(bAddress) Mod 2 = 0) Then
      mhSerialPort.SetParitybit(Rs232.DataParity.Parity_Odd)
    Else
      mhSerialPort.SetParitybit(Rs232.DataParity.Parity_Even)
    End If
    mhSerialPort.Write(addr)
    System.Threading.Thread.Sleep(1)
    ' Send data
    For i = 0 To ilen - 1
      If (cBit.CountBitOn(buf(i)) Mod 2 = 0) Then
        mhSerialPort.SetParitybit(Rs232.DataParity.Parity_Even)
      Else
        mhSerialPort.SetParitybit(Rs232.DataParity.Parity_Odd)
      End If
      mhSerialPort.Write(buf(i))
    Next
  Catch ex As Exception
    LastMsg = ex.Message
    Return False
  End Try
End Sub
Modbus protocol

**NOTE:** To select this protocol set config 1 to 0x21
This simplifies the communication by eliminating the need to toggle the parity bit, and it adds a check for the communication with the LRC.
The LRC is calculated by adding all the bytes in the message except the ":" at the beginning, and the LF and CR at the end of the message. The addition is done ignoring the carry. Then the result is negated in 2's complement format.

8 bits even parity 1 stop for modbus protocol (standard modbus is 7 bits)

The command starts with a ":" then the address of the module from 1 to 250 then the cmd and then data after the data the Longitudinal Redundancy Check (LRC) then the CR and LF
All the data are sent in ASCII format on an Hex base.

eexample
determine 355 to location 100 in the eeprom of module 2
:0268006401639C cr lf
the module returns
:026830 cr lf
0x65  reset the board
  send back :Address65LRC
0x66    read the sensor
0x67   read temperature
0x68  write to eeprom address and data in HEX
    Address(2B)68EEPROMLocation(4)Data(4)LRC(2)
    send back :Address68LRC
0x69   read eeprom
    ADDRESS(2Bytes)69EEPROMLocation(4Bytes)LRC(2bytes)
0x6A  send revision
Application Note
SPECTROTILT™ Dyna - Communication Protocols

Examples:
Write 15 to location 22 in module address 1
:01680016000F74 cr lf
Read Sensor
:016633 cr lf
Read location 2 on module address 20
:146900026A cr lf