DM2200 Temperature Sensor Demonstration

by Frank Perkins

The example application shown in Figure 1 measures temperature using a thermistor-resistor circuit connected to GP6 on a DM2200 field node. The application is written in Microsoft Visual Basic 6.0 (SP6). The sensor used is a Vishay BCcomponents 2322 640 66103 NTC thermistor, which has a nominal resistance of 10K at 77 °F (25 °C). This thermistor is readily available from Digi-Key and other electronic component distributors. The data sheet for the thermistor was located at www.vishay.com-docs-29049-23226403.pdf. The thermistor is connected between the 3 V regulated output on the DM2200 and the ADC input, and a 10K, 1% resistor is connected from the ADC input to ground. The thermistor and the 10K resistor form a voltage divider to drive the ADC input, where the voltage increases with increasing temperature, as shown in Figure 2. The photograph in Figure 3 shows the thermistor and resistor installed on an IM2200 interface module.

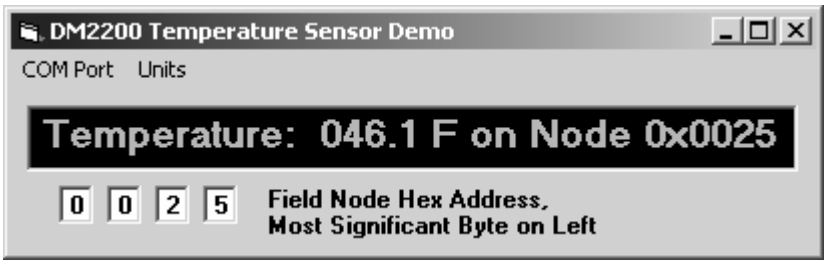


Figure 1: DM2200 Themistor Demo

The example program is contained in a single Visual Basic form. The heart of the program is *Timer1_Timer*, which is launched once every two seconds by a timer control. This routine first checks the serial port input buffer. If the input buffer is holding one or more bytes, the *GetRX* subroutine is called to collect and process them. *Timer1_Timer* then calls *TXCmd* to output a *CMD_IO IO_ADC* (read ADC), which propagates through the DM2200 network to the addressed field node. The field nodes makes the ADC measurement and sends a response back to be read on the next *Timer1_Timer* cycle.

Each time *GetRX* is called, it collects the newly received bytes in *ComData$* and calls *RXRsp*. The function of *RXRsp* is to extract messages from the stream of received bytes. *RXRsp* adds *ComData$* to the end of the *FIFO$* string. The *FIFO$* string is then scanned and complete messages are extracted. Any partial message at the end of the FIFO$ string is saved for future processing.

Each time *RXRsp* extracts a complete network message, it calls *ShowT*. This subroutine filters out all network messages except *CMD_IO IO_ADC* response messages. The ADC measurements are then converted to temperature readings and displayed. The *GetTemp* subroutine does the temperature conversion using the *TmpArry* table values and interpolation. The *TmpArry* table values were derived from a resistance versus temperate table in the BCcomponents data sheet. These resistance

values were, in turn, used to compute the *TmpArry* table of output voltage ratios, in terms of ADC counts, for the thermistor-resistor voltage divider.

TXCmd builds and sends command messages. When the application is started and a valid COM port is entered, a *CMD_MASTER* (enable master beacon) is immediately sent. Then *TXCmd* sends commands to disable network sleep mode and to set the master beacon interval to 10 seconds. Following these initialization messages, TXCmd sends *CMD_IO IO_ADC* messages every two seconds. Note that the *op_ref* (message ID) byte is changed for every transmitted message. This is required for normal network operation.

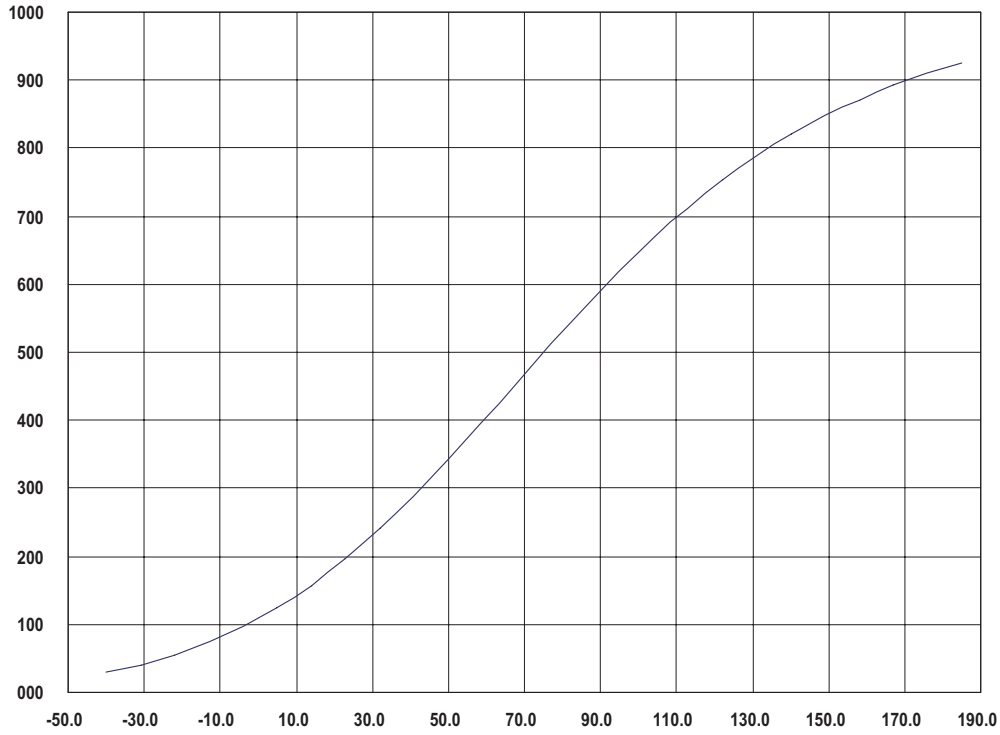


Figure 2: ADC Count$_{10}$ versus Temperature in °F

The temperature sensor example has been kept simple to avoid obscuring the basic DM2200 network command-response program methodology. Nevertheless, this simple program can be used to remotely monitor a temperatures over an outdoor path of more than a mile using the four nodes in the DM2200-916-DK development kit.

An application install package, the project files and a spreadsheet tabulation of the temperature sensor voltage divider can be downloaded from Cirronet's web site at www.cirronet.com., and is also included on the DM2200-916-DK development kit.

After the application is installed, launch it and select the COM port. For PCs with a serial port, this is usually *COM1*. In the case of USB operation, a virtual COM port is assigned to the IM2200 USB port by the associated USB virtual COM port driver. *COM5* is shown in the example in Figure 4. Next enter the address of the field node that has the

thermistor-resistor voltage divider installed (*Node 0x0025* is shown in Figure 1). The *Units* menu selects Fahrenheit or Celsius for the temperature display.

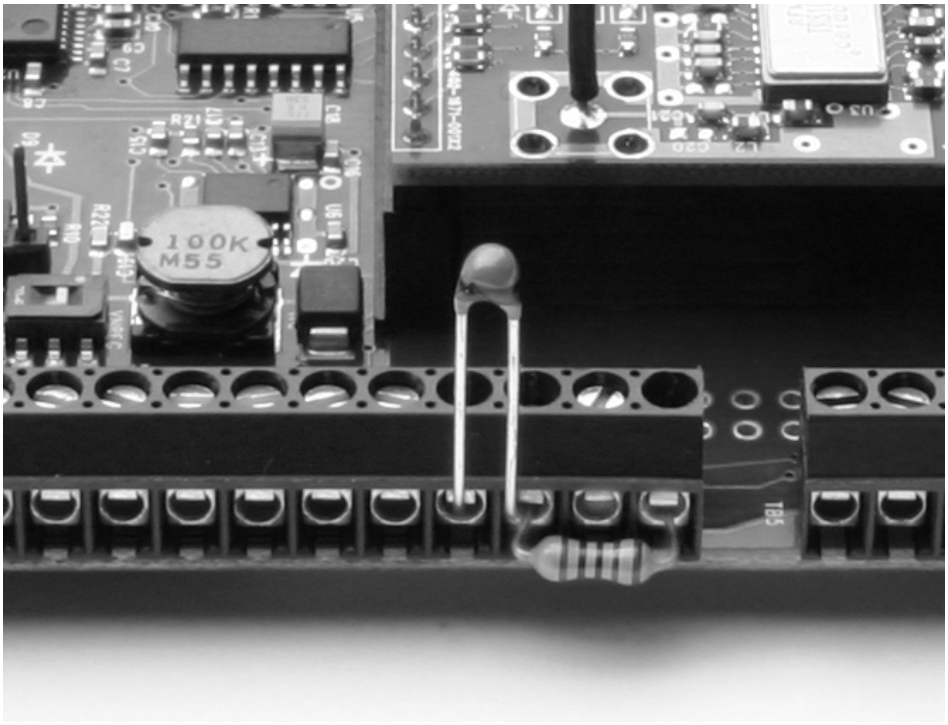


Figure 3: IM2200 Thermistor-Resistor Installation

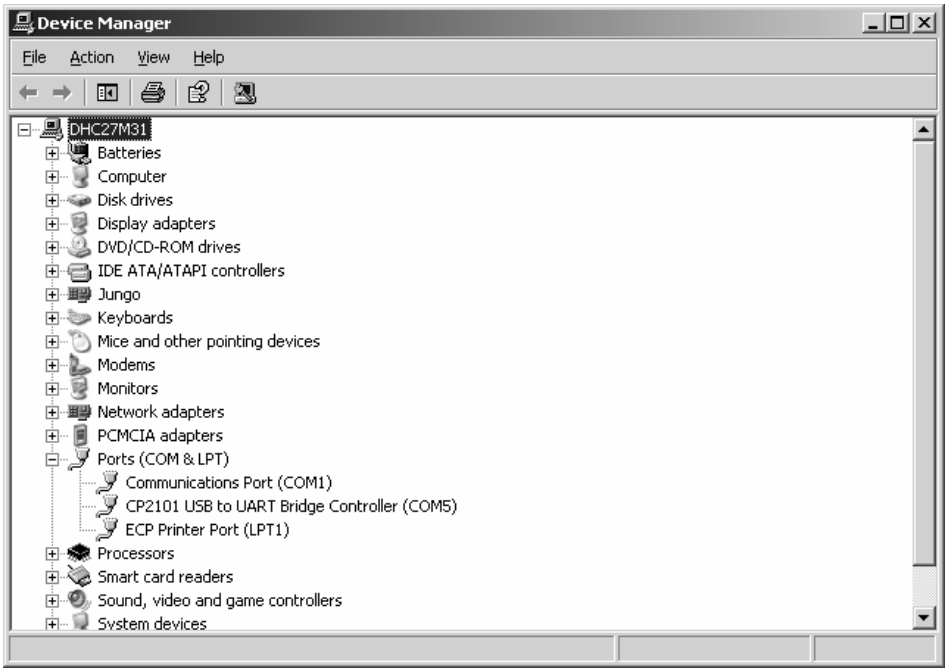


Figure 4: COM Port Verification

DM2200 Temperature Sensor Application Source Code

```
VERSION 5.00
Object = "{648A5603-2C6E-101B-82B6-000000000014}#1.1#0"; "MSCOMM32.OCX"
Begin VB.Form Form1
   Caption         =   "DM2200 Temperature Sensor Demo"
   ClientHeight    =   1215
   ClientLeft      =   165
   ClientTop       =   735
   ClientWidth     =   6030
   ForeColor       =   &H00000000&
   LinkTopic       =   "Form1"
   ScaleHeight     =   1215
   ScaleWidth      =   6030
   StartUpPosition =   3  'Windows Default
   Begin VB.TextBox Text4
      Alignment       =   2  'Center
      BeginProperty Font
         Name            =   "MS Sans Serif"
         Size            =   8.25
         Charset         =   0
         Weight          =   700
         Underline       =   0   'False
         Italic          =   0   'False
         Strikethrough   =   0   'False
      EndProperty
      Height          =   285
      Left            =   1440
      TabIndex        =   5
      Text            =   "0"
      Top             =   720
      Width           =   255
   End

   Begin VB.TextBox Text3
      Alignment       =   2  'Center
      BeginProperty Font
         Name            =   "MS Sans Serif"
         Size            =   8.25
         Charset         =   0
         Weight          =   700
         Underline       =   0   'False
         Italic          =   0   'False
         Strikethrough   =   0   'False
      EndProperty
      Height          =   285
      Left            =   1080
      TabIndex        =   4
      Text            =   "0"
      Top             =   720
      Width           =   255
   End
   Begin VB.TextBox Text2
      Alignment       =   2  'Center
      BeginProperty Font
         Name            =   "MS Sans Serif"
         Size            =   8.25
         Charset         =   0
         Weight          =   700
         Underline       =   0   'False
         Italic          =   0   'False
         Strikethrough   =   0   'False
      EndProperty
      Height          =   285
      Left            =   720
      TabIndex        =   3
      Text            =   "0"
      Top             =   720
```

```
   Width           =   255
End
Begin VB.TextBox Text1
   Alignment       =   2  'Center
   BeginProperty Font
      Name            =   "MS Sans Serif"
      Size            =   8.25
      Charset         =   0
      Weight          =   700
      Underline       =   0   'False
      Italic          =   0   'False
      Strikethrough   =   0   'False
   EndProperty
   Height          =   285
   Left            =   360
   TabIndex        =   2
   Text            =   "0"
   Top             =   720
   Width           =   255
End
Begin MSCommLib.MSComm MSComm1
   Left            =   4560
   Top             =   120
   _ExtentX        =   1005
   _ExtentY        =   1005
   _Version        =   393216
   DTREnable       =   -1  'True
   BaudRate        =   19200
End
Begin VB.Timer Timer1
   Enabled         =   0   'False
   Interval        =   1000
   Left            =   4080
   Top             =   120
End
Begin VB.Label Label2
   Caption         =   "Field Node Hex Address, Most Significant Byte on Left"
   BeginProperty Font
      Name            =   "MS Sans Serif"
      Size            =   8.25
      Charset         =   0
      Weight          =   700
      Underline       =   0   'False
      Italic          =   0   'False
      Strikethrough   =   0   'False
   EndProperty
   Height          =   495
   Left            =   1920
   TabIndex        =   1
   Top             =   720
   Width           =   2535
End
Begin VB.Label Label1
   BackColor       =   &H00000000&
   BorderStyle     =   1  'Fixed Single
   Caption         =   " Select COM Port"
   BeginProperty Font
      Name            =   "MS Sans Serif"
      Size            =   13.5
      Charset         =   0
      Weight          =   700
      Underline       =   0   'False
      Italic          =   0   'False
      Strikethrough   =   0   'False
   EndProperty
   ForeColor       =   &H0000FF00&
   Height          =   495
```

```
      Left            =   120

      TabIndex        =   0
      Top             =   120
      Width           =   5775
   End
   Begin VB.Menu mnuSerial
      Caption         =   "COM Port"
      Begin VB.Menu mnuCOM1
         Caption         =   "COM 1"
      End
      Begin VB.Menu mnuCOM2
         Caption         =   "COM 2"
      End
      Begin VB.Menu mnuCOM3
         Caption         =   "COM 3"
      End
      Begin VB.Menu mnuCOM4
         Caption         =   "COM 4"
      End
      Begin VB.Menu mnuCOM5
         Caption         =   "COM 5"
      End
      Begin VB.Menu mnuCOM6
         Caption         =   "COM 6"
      End
      Begin VB.Menu mnuCOM7
         Caption         =   "COM 7"
      End
      Begin VB.Menu mnuCOM8
         Caption         =   "COM 8"
      End
   End
   Begin VB.Menu mnuUnits
      Caption         =   "Units"
      Begin VB.Menu mnuF
         Caption         =   "Deg F"
         Checked         =   -1  'True
      End
      Begin VB.Menu mnuC
         Caption         =   "Deg C"
      End
   End
End

Attribute VB_Name = "Form1"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False

' DM2200TD_10.FRM, 2007.07.25 @ 11:45 CDT
' DM2200 tutorial software
' Thermistor temperature sensor demo
' NO representation is made that this
' software is suitable for any purpose
' Compiled in Microsoft Visual Basic 6.0
```

```
' declare global variables and constants:
  Dim ComData$                                          ' COM input string
  Dim FIFO$                                             ' RX character FIFO
  Dim RPkt$                                             ' response message
  Dim BOT$                                              ' message start character
  Dim EOT$                                              ' message end character
  Dim MSG$                                              ' read ADC on GP6 6 of node 2
  Dim MST$                                              ' master node beacon ON command
  Dim ADC As Single                                     ' get ADC count command
  Dim TmpArry(32, 2) As Single                          ' temperature conversion array
  Dim TempFF As Variant                                 ' formatted temperature, F
  Dim TempCF As Variant                                 ' formatted temperature, C
  Dim P As Integer                                      ' COM port
  Dim T As Integer                                      ' temperature F or C select flag
  Dim NN$                                               ' node ID string
  Dim ID As Long                                        ' numeric node ID
  Dim OP As Integer                                     ' op_ref (packet ID)
  Dim HT As Integer                                     ' high TX address byte
  Dim LT As Integer                                     ' low TX address byte
  Dim A0 As Long                                        ' address nibble 0
  Dim A1 As Long                                        ' address nibble 1
  Dim A2 As Long                                        ' address nibble 2
  Dim A3 As Long                                        ' address nibble 3
  Dim MND$                                              ' master node sleep disable command
  Dim MNB$                                              ' master node beacon interval command
  Dim S As Integer                                      ' TX state variable

Private Sub Form_Load()
' initialize global variables and constants:
  ComData$ = ""                                         ' clear COM input string
  FIFO$ = ""                                            ' clear FIFO string
  RPkt$ = ""                                            ' clear response string
  BOT$ = Chr$(&H0)                                      ' message start character
  EOT$ = Chr$(&H4)                                      ' message end character
  Call LoadMsg                                          ' load command messages
  ADC = 0                                               ' clear ADC
  Call LoadArray                                        ' load temperature conversion array
  TempFF = 0                                            ' clear TempFF
  TempCF = 0                                            ' clear TempCF
  P = 1                                                 ' default COM port number
  T = 0                                                 ' set display default as deg F
  NN$ = ""                                              ' clear node ID string
  ID = 0                                                ' clear node ID value
  OP = 0                                                ' clear op_ref value
  HT = 0                                                ' clear TX high address byte
  LT = 0                                                ' clear TX low address byte
  A0 = 0                                                ' clear address nibble 0
  A1 = 0                                                ' clear address nibble 1
  A2 = 0                                                ' clear address nibble 2
  A3 = 0                                                ' clear address nibble 3
  S = 1                                                 ' set TX state to 1
  Form1.Left = (Screen.Width - Form1.Width) / 1.2       ' put form at
  Form1.Top = (Screen.Height - Form1.Height) / 5        '   upper right
  Timer1.Interval = 2000                                ' set timer interval to 2 seconds
  Timer1.Enabled = False                                ' start timer when COM port selected
End Sub

Private Sub Timer1_Timer()
  If MSComm1.InBufferCount > 0 Then                     ' if bytes in COM input buffer
    Call GetRX                                          '   call to get RX bytes
  End If
  Call TXCmd                                            ' call to send TX command
End Sub
```

```
Public Sub TXCmd()
  If (S = 1) Then                                          ' if TX state 1
    MSComm1.Output = MND$                                  '   disable master node sleep cycle
    S = S + 1                                              '   increment state
  ElseIf (S = 2) Then                                      ' if TX state 2
    MSComm1.Output = MNB$                                  '   set master beacon period to 10 s
    S = S + 1                                              '   increment state
  ElseIf (S = 3) Then                                      ' if TX state 3
    OP = OP + 1                                            '   increment op_ref
    If (OP = 9) Then                                       '   if op_ref is 9
      OP = 1                                               '     reset op_ref to 1
    End If
    Mid(MSG$, 4, 3) = Chr$(OP) + Chr$(LT) + Chr$(HT)       '   update op_ref and address
    MSComm1.Output = MSG$                                  '   send command to read ADC
  End If
End Sub

Public Sub GetRX()
  On Error Resume Next                                     ' set up error handler
  ComData$ = MSComm1.Input                                 ' load ComData$ from COM input buffer
  Call RXRsp                                               ' extract RX response message(s)
End Sub

Public Sub RXRsp()
  Dim I As Integer                                         ' index to next BOT$
  Dim L As Integer                                         ' length byte value
  Dim Q As Integer                                         ' FIFO length
  FIFO$ = FIFO$ & ComData$                                 ' add ComData$ to end of FIFO$ string
  ComData$ = ""                                            ' clear ComData$
  RPkt$ = ""                                               ' clear RPkt$
  I = 0                                                    ' clear index
  Q = Len(FIFO$)                                           ' load Q with number in FIFO$
  If (Q > 256) Then                                        ' if FIFO$ is overflowing (garbage)
    FIFO$ = ""                                             '   clear FIFO$
    Exit Sub                                               '   exit
  End If
  If (Q < 6) Then                                          ' if FIFO$ less than 6 characters
    Exit Sub                                               '   exit
  End If
  For Y = 1 To 8                                           ' loop processes up to 8 msgs per call
    Q = Len(FIFO$)                                         ' update FIFO$ length
    I = CInt(InStr(FIFO$, BOT$))                           ' find start of next message (BOT$)
    If (I = 0) Then                                        ' if no BOT$ in FIFO$ string
      Exit For                                             '   exit For
    End If
    If (Q > I) Then                                        ' if FIFO$ characters to the right
      FIFO$ = Right$(FIFO$, (Q - I))                       '   remove BOT$ and bytes to the left
      Q = Len(FIFO$)                                       '   update FIFO length
    Else                                                   ' else
      Exit For                                             '   exit For
    End If
    L = Asc(Left$(FIFO$, 1))                               ' get message length up to EOT$
    If (Q >= (L + 2)) Then                                 ' if FIFO$ length >= message length
      If (Mid$(FIFO$, (L + 2), 1) = EOT$) Then             '   if EOT$ properly located
        RPkt$ = Mid$(FIFO$, 2, L)                          '     copy core message
        FIFO$ = Right$(FIFO$, (Q - (L + 2)))               '     delete message from FIFO$
      Else                                                 '   else
        Exit For                                           '     exit For
      End If
    Else                                                   ' else
      Exit For                                             '   exit For
    End If
```

```
    If RPkt$ <> "" Then                                 ' if RPkt$ is not null
      Call ShowT                                        '   call to display temperature
      RPkt = ""                                         '   clear RPkt$
    End If
  Next Y
End Sub

Public Sub ShowT()
' filter for valid ADC response
  Dim RH As Integer                                     ' high RX address byte
  Dim RL As Integer                                     ' low RX address byte
  Dim RT As Integer                                     ' 16-bit RX address
  Dim HB As Integer                                     ' high ADC byte
  Dim HL As Integer                                     ' low ADC byte
  Dim VL As Integer                                     ' ADC integer value
  If (Len(RPkt$) <> 9) Then                             ' if wrong length
    Exit Sub                                            '   exit
  ElseIf (Asc(Mid$(RPkt$, 3, 1)) <> &H1) Then           ' else if invalid response
    Exit Sub                                            '   exit
   ElseIf (Asc(Mid$(RPkt$, 1, 1)) <> &HB) Then          ' else if not I/O response
    Exit Sub                                            '   exit
  ElseIf (Asc(Mid$(RPkt$, 6, 1)) <> &H2) Then           ' else if not ADC response
    Exit Sub                                            '   exit
  End If
' calculate and show new temperature
  RH = Asc(Mid$(RPkt$, 5, 1))                           ' get high address byte
  RL = Asc(Mid$(RPkt$, 4, 1))                           ' get low address byte
  RT = 256 * RH + RL                                    ' calculate 16-bit address
  NN$ = Hex(RT)                                         ' convert address to string
  If (Len(NN$) = 1) Then                                ' if one hex character
    NN$ = " 0x000" + NN$                                '   add 0x000 to from
  ElseIf (Len(NN$) = 2) Then                            ' else if two hex characters
    NN$ = " 0x00" + NN$                                 '   add 0x00 to front
  ElseIf (Len(NN$) = 3) Then                            ' else if three hex characters
    NN$ = " 0x0" + NN$                                  '   add 0x0 to front
  Else                                                  ' else
    NN$ = " 0x" + NN$                                   '   add 0x to front
  End If
  HB = Asc(Mid$(RPkt$, 9, 1))                           ' get high ADC byte
  LB = Asc(Mid$(RPkt$, 8, 1))                           ' get low ADC byte
  VL = 256 * HB + LB                                    ' calculate total ADC value
  VL = VL / 4                                           ' scale to 10 bits
  ADC = CSng(VL)                                        ' covert ADC value to single
  Call GetTemp                                          ' convert to temperature
  If (T = 0) Then                                       ' if temperature flag clear
    Label1.Caption = " Temperature:  " + TempFF _
    + " F on Node" + NN$                                '   show F temperature
  Else                                                  ' else
    Label1.Caption = " Temperature:  " + TempCF _
    + " C on Node" + NN$                                '   show C temperature
  End If
End Sub

Public Sub GetTemp()
' interpolate from table look-up
  Dim CH As Single                                      ' high ADC interpolation count
  Dim CL As Single                                      ' low ADC interpolation count
  Dim TH As Single                                      ' high temperature interpolation value
  Dim TL As Single                                      ' low temperature interpolation value
  Dim TempF As Single                                   ' raw temperature, F
  Dim TempC As Single                                   ' raw temperature, C
  Dim X As Integer                                      ' loop counter
```

```
  For X = 2 To 25
    If (ADC <= TmpArry(X, 1)) Then
      CH = TmpArry(X, 1)
      CL = TmpArry((X - 1), 1)
      TH = TmpArry(X, 2)
      TL = TmpArry((X - 1), 2)
      TempF = TL + (((ADC - CL) * (TH - TL)) / (CH - CL))
      TempFF = Format(TempF, "000.0")
      TempC = (5 / 9) * (TempF - 32)
      TempCF = Format(TempC, "000.0")
      Exit For
    End If
  Next X
  End Sub

Public Sub LoadArray()
' load table look-up array
  TmpArry(1, 1) = 30
  TmpArry(1, 2) = -40
  TmpArry(2, 1) = 41
  TmpArry(2, 2) = -31
  TmpArry(3, 1) = 55
  TmpArry(3, 2) = -22
  TmpArry(4, 1) = 74
  TmpArry(4, 2) = -13
  TmpArry(5, 1) = 96
  TmpArry(5, 2) = -4
  TmpArry(6, 1) = 124
  TmpArry(6, 2) = 5
  TmpArry(7, 1) = 157
  TmpArry(7, 2) = 14
  TmpArry(8, 1) = 196
  TmpArry(8, 2) = 23
  TmpArry(9, 1) = 241
  TmpArry(9, 2) = 32
  TmpArry(10, 1) = 290
  TmpArry(10, 2) = 41
  TmpArry(11, 1) = 343
  TmpArry(11, 2) = 50
  TmpArry(12, 1) = 398
  TmpArry(12, 2) = 59
  TmpArry(13, 1) = 455
  TmpArry(13, 2) = 68
  TmpArry(14, 1) = 512
  TmpArry(14, 2) = 77
  TmpArry(15, 1) = 567
  TmpArry(15, 2) = 86
  TmpArry(16, 1) = 619
  TmpArry(16, 2) = 95
  TmpArry(17, 1) = 668
  TmpArry(17, 2) = 104
  TmpArry(18, 1) = 712
  TmpArry(18, 2) = 113
  TmpArry(19, 1) = 753
  TmpArry(19, 2) = 122
  TmpArry(20, 1) = 788
  TmpArry(20, 2) = 131
  TmpArry(21, 1) = 820
  TmpArry(21, 2) = 140
  TmpArry(22, 1) = 847
  TmpArry(22, 2) = 149
  TmpArry(23, 1) = 871
  TmpArry(23, 2) = 158
  TmpArry(24, 1) = 892
  TmpArry(24, 2) = 167
  TmpArry(25, 1) = 910
  TmpArry(25, 2) = 176
  TmpArry(26, 1) = 925
```

```
  TmpArry(26, 2) = 185
End Sub

Private Sub mnuCOM1_Click()
  P = 1                                          ' COM port 1
  Call SetCOM                                    ' open COM port
  Call ResetCOM                                  ' clear all checkmarks
  mnuCOM1.Checked = True                         ' set this checkmark
End Sub

Private Sub mnuCOM2_Click()
  P = 2                                          ' COM port 2
  Call SetCOM                                    ' open COM port
  Call ResetCOM                                  ' clear all checkmarks
  mnuCOM2.Checked = True                         ' set this checkmark
End Sub

Private Sub mnuCOM3_Click()
  P = 3                                          ' COM port 3
  Call SetCOM                                    ' open COM port
  Call ResetCOM                                  ' clear all checkmarks
  mnuCOM3.Checked = True                         ' set this checkmark
End Sub

Private Sub mnuCOM4_Click()
  P = 4                                          ' COM port 4
  Call SetCOM                                    ' open COM port
  Call ResetCOM                                  ' clear all checkmarks
  mnuCOM4.Checked = True                         ' set this checkmark
End Sub

Private Sub mnuCOM5_Click()
  P = 5                                          ' COM port 5
  Call SetCOM                                    ' open COM port
  Call ResetCOM                                  ' clear all checkmarks
  mnuCOM5.Checked = True                         ' set this checkmark
End Sub

Private Sub mnuCOM6_Click()
  P = 6                                          ' COM port 6
  Call SetCOM                                    ' open COM port
  Call ResetCOM                                  ' clear all checkmarks
  mnuCOM6.Checked = True                         ' set this checkmark
End Sub

Private Sub mnuCOM7_Click()
  P = 7                                          ' COM port 7
  Call SetCOM                                    ' open COM port
  Call ResetCOM                                  ' clear all checkmarks
  mnuCOM7.Checked = True                         ' set this checkmark
End Sub

Private Sub mnuCOM8_Click()
  P = 8                                          ' COM port 8
  Call SetCOM                                    ' open COM port
  Call ResetCOM                                  ' clear all checkmarks
  mnuCOM8.Checked = True                         ' set this checkmark
End Sub
```

```
Public Sub SetCOM()
  On Error GoTo Skip                                ' invalid COM port recovery
  If (MSComm1.PortOpen = True) Then                 ' if COM port open
    MSComm1.PortOpen = False                        '   close it
    Timer1.Enabled = False                          '   stop timer
  End If
  MSComm1.CommPort = P                              ' initialize COM port
  MSComm1.Settings = "19200,N,8,1"                  ' at 19200 b/s
  MSComm1.RThreshold = 0                            ' poll only, no interrupts
  MSComm1.InputLen = 0                              ' read all bytes
  MSComm1.PortOpen = True                           ' open COM port
  Label1.Caption = " Select Node ID:  "             ' update display
  MSComm1.Output = MST$                             ' master beacon command
  Timer1.Enabled = True                             ' enable timer
  Exit Sub
Skip:
  Label1.Caption = " Invalid COM  "                 ' update display
End Sub

Public Sub ResetCOM()
  mnuCOM1.Checked = False                           ' clear checkmark
  mnuCOM2.Checked = False                           ' clear checkmark
  mnuCOM3.Checked = False                           ' clear checkmark
  mnuCOM4.Checked = False                           ' clear checkmark
  mnuCOM5.Checked = False                           ' clear checkmark
  mnuCOM6.Checked = False                           ' clear checkmark
  mnuCOM7.Checked = False                           ' clear checkmark
  mnuCOM8.Checked = False                           ' clear checkmark
End Sub

Private Sub mnuC_Click()
  mnuC.Checked = True                               ' check deg C
  mnuF.Checked = False                              ' uncheck deg F
  T = 1                                             ' show deg C
End Sub

Private Sub mnuF_Click()
  mnuC.Checked = False                              ' uncheck deg C
  mnuF.Checked = True                               ' check deg F
  T = 0                                             ' show deg F
End Sub

Private Sub Text1_Change()
  Dim N As Long                                     ' nibble
  If (Len(Text1.Text) > 0) Then                     ' if new input
    N = Asc(UCase$(Text1.Text))                     '   get first character
  End If
  If (48 <= N) And (N <= 57) Then                   ' convert 0 to 9 ASCII
    A0 = N - 48                                     '   to number
  ElseIf (65 <= N) And (N <= 70) Then               ' convert A to F ASCII
    A0 = N - 55                                     '   to number
  Else                                              ' else
    A0 = 0                                          '   default number
  End If
  Call NodeID                                       ' calculate node address
End Sub

Private Sub Text2_Change()
  Dim N As Long                                     ' nibble
  If (Len(Text2.Text) > 0) Then                     ' if new input
    N = Asc(UCase$(Text2.Text))                     '   get first character
  End If
```

```
  If (48 <= N) And (N <= 57) Then                    ' convert 0 to 9 ASCII
    A1 = N - 48                                      '   to number
  ElseIf (65 <= N) And (N <= 70) Then                ' convert A to F ASCII
    A1 = N - 55                                      '   to number
  Else                                               ' else
    A1 = 0                                           '   default number
  End If
  Call NodeID                                        ' calculate node address
End Sub

Private Sub Text3_Change()
  Dim N As Long                                      ' nibble
  If (Len(Text3.Text) > 0) Then                      ' if new input
    N = Asc(UCase$(Text3.Text))                      '   get first character
  End If
  If (48 <= N) And (N <= 57) Then                    ' convert 0 to 9 ASCII
    A2 = N - 48                                      '   to number
  ElseIf (65 <= N) And (N <= 70) Then                ' convert A to F ASCII
    A2 = N - 55                                      '   to number
  Else                                               ' else
    A2 = 0                                           '   default number
  End If
  Call NodeID                                        ' calculate node address
End Sub

Private Sub Text4_Change()
  Dim N As Long                                      ' nibble
  If (Len(Text4.Text) > 0) Then                      ' if new input
    N = Asc(UCase$(Text4.Text))                      '   get first character
  End If
  If (48 <= N) And (N <= 57) Then                    ' convert 0 to 9 ASCII
    A3 = N - 48                                      '   to number
  ElseIf (65 <= N) And (N <= 70) Then                ' convert A to F ASCII
    A3 = N - 55                                      '   to number
  Else                                               ' else
    A3 = 0                                           '   default number
  End If
  Call NodeID                                        ' calculate node address
End Sub

Public Sub NodeID()
  ID = 0                                             ' reset ID
  ID = ID + (A0 * 4096)                              ' add nibble to total
  ID = ID + (A1 * 256)                               ' add nibble to total
  ID = ID + (A2 * 16)                                ' add nibble to total
  ID = ID + (A3)                                     ' add nibble to total
  HT = ID \ 256                                      ' HT is integer quotient
  LT = ID - (256 * HT)                               ' LT is remainder
  Label1.Caption = " Temperature:  "                 ' update display
End Sub

Public Sub LoadMsg()

  MST$ = Chr$(&H0) + Chr$(&H4) + Chr$(&H3) _
  + Chr$(&H0) + Chr$(&H0) + Chr$(&H0) + Chr$(&H4)    ' turn on master beacon command

  MND$ = Chr$(&H0) + Chr$(&H7) + Chr$(&H2) _
  + Chr$(&H0) + Chr$(&H0) + Chr$(&H1) + Chr$(&H1E) _
  + Chr$(&H0) + Chr$(&H0) + Chr$(&H4)                ' master node sleep cycle disable

  MNB$ = Chr$(&H0) + Chr$(&H6) + Chr$(&H2) _
  + Chr$(&H0) + Chr$(&H0) + Chr$(&H1) + Chr$(&H9) _
  + Chr$(&HA) + Chr$(&H4)                            ' master node 10 s beacon interval
```

```
  MSG$ = Chr$(&H0) + Chr$(&H8) + Chr$(&HB) _
  + Chr$(&H0) + Chr$(&H0) + Chr$(&H0) + Chr$(&H2) _
  + Chr$(&H26) + Chr$(&HA) + Chr$(&H0) + Chr$(&H4)          ' read ADC on GP6 command

End Sub

Private Sub Form_Unload(Cancel As Integer)
  If (MSComm1.PortOpen = True) Then                         ' if COM port open
    MSComm1.PortOpen = False                                '   close it
  End If
  End                                                       ' done
End Sub
```