DM1800/DM1810 Sleep Duty Cycle Demonstration

by Frank Perkins

The example application shown in Figure 1 measures temperature using a thermistor-resistor circuit connected to the ADC input on a DM1800 or DM1810 field node. The field node can be set for a 15 seconds active, 15 to 300 seconds sleep duty cycle. This power management method can greatly extend field node battery life where data measurements are acceptable at intervals greater than 30 seconds. The application is written in Microsoft Visual Basic 6.0 (SP6). The sensor is a Vishay BCcomponents 2322 640 66103 NTC thermistor, which has a resistance of 10K at 77 °F (25 °C). This thermistor is available from Digi-Key and other electronic component distributors. The data sheet for the thermistor is located at www.vishay.com-docs-29049-23226403.pdf. The thermistor is connected from the 3 V regulated output on the DM1800/DM1810 to 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 4. Figure 5 shows the thermistor and resistor installed on an IM1800 interface module. These components are built into the IM1800-1 interface board.

The DM1800/DM1810 has a number of sleep cycle options, as discussed in Section 3.1 of Application Notes AN1800-1-23/4/5 and AN1810-1-23/4/5. In this example, the sleep cycle is orchestrated with a *Write Dynamic Power Management Parameters* command, where *power management* is enabled and *report on wake up* event messaging is enabled. The sleep period can be set for 15 seconds, 60 seconds, or 300 seconds. This configuration causes the addressed field node to remain awake about 15 seconds after receiving the command, and then to sleep for approximately the selected sleep period.

At the end of the sleep period, the node sends a *wake up* event message. When the program receives the *wake up* event message, it resends the sleep command and then sends *Digital Out High* and *Read ADC* commands during the 15 second interval before the next sleep period begins. The ADC reading is interpreted as a temperature and is displayed as shown in Figure 1, until the next *wake up* event message is received. At this point *Node Awake* is displayed as shown in Figure 2. The temperature reading is then refreshed and replaces the *Node Awake* message after a few seconds.

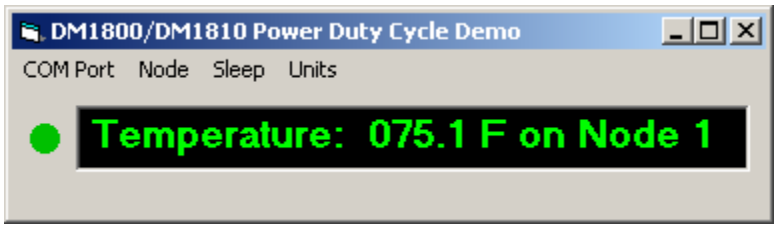


Figure 1: Reading Node Temperature

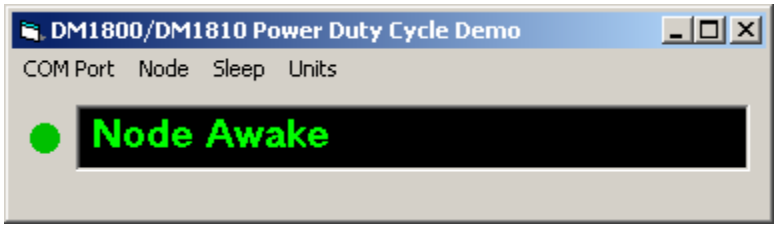


Figure 2: Node Awake Event Message

Figure 3 shows the sleep period menu. The default sleep period is 15 seconds. Note the active interval is fixed at 15 seconds for any sleep period selection.

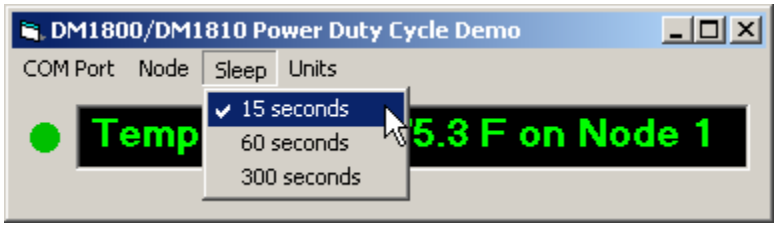


Figure 3: Setting the Sleep Period

The example program is contained in a single Visual Basic form. The heart of the program is *Timer1_Timer*, which is launched once every 500 ms 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.*

Each time *TXCmd* sends a command it sets a response pending flag, *RPFLag*, and does not send out another command until this flag is reset by *GetRX* when the response is received, or after a five second gross time out. The sequence of commands sent by the *TXCmd* subroutine is controlled by state variable *S*. When *S* is 1, *TXCmd* sends the *Power Management* sleep command to the selected field node, using utility subroutine *SndTX* which adds bi-sync framing and a checksum to the command. When S is 2, a *Digital Output High* command is sent. This selects the thermistor-resistor circuit built into the IM1800-1 if this interface board is being used. When S is 3, a *Read ADC* command is issued. *S* continues to be incremented until reset to 1 by the reception of a *wake up* event message, or when *S* reaches a safety time out value related to the selected sleep period.

Each time *GetRX* is called, it collects the newly received bytes in *ComData$* and adds these bytes to the end of the *FIFO$* string. The *FIFO$* string is then scanned byte-by-byte, removing bi-sync framing and escape characters from each complete message found in the *FIFO$* string. Local command strings are ignored, and any partial messages at the end of the FIFO$ string are saved for future processing.

Each time a complete message is extracted, *GetRX* resets *RPFLag* and checks the message length. If its length is seven bytes, *ChkWkUp* is called to confirm the message is a *wake up* event message for the addressed field node. If so, *ChkWkUp* resets state

variable *S* to 1 and displays the *Node Awake* message. If the extracted message length is 10 bytes, *GetRX* calls *ShowT*. This subroutine filters out all messages except a *Read ADC Input with 10-bit Resolution* response message from the addressed field node. The ADC measurement is then converted to a temperature reading 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.

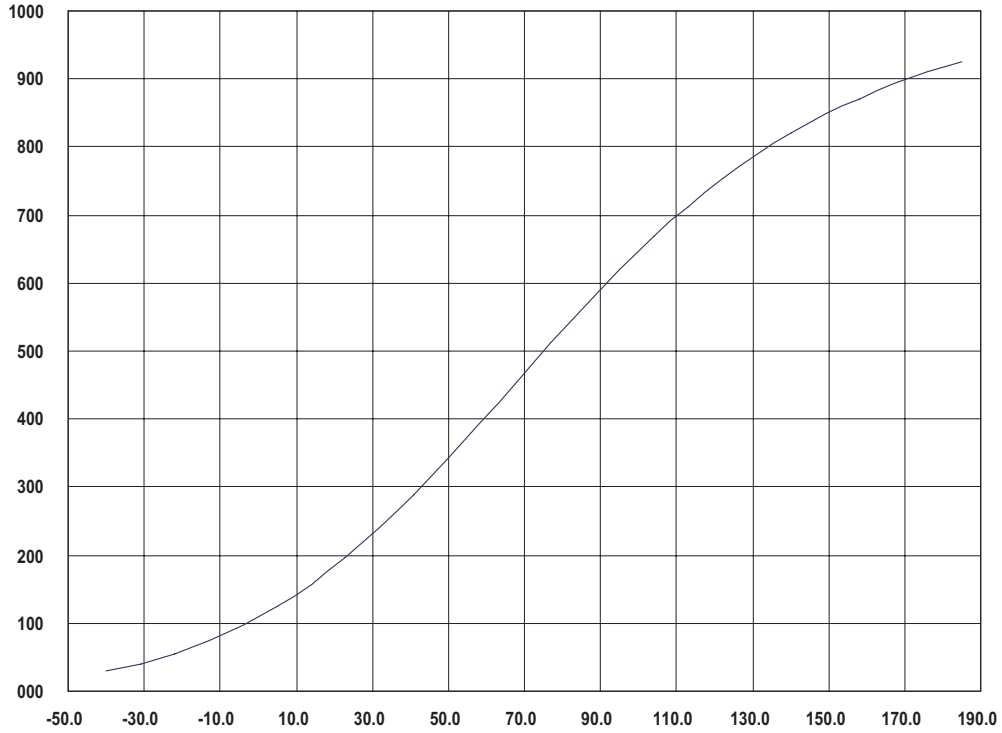


Figure 4: ADC Count$_{10}$ versus Temperature in °F

This example application program has been kept simple to avoid obscuring the basic DM1800/DM1810 network command-response program methodology. Nevertheless, the program can be used to remotely monitor a temperatures over an outdoor path of more than a mile using a field node and three router nodes from a DM1810-916-DK Pro development kit. Compared to continuous field node operation, the sleep period options will increase field node battery life by a factor of about 1.8 for a 15 second sleep period, 4.5 for a 60 second sleep period, and 18 for a 300 second sleep period.

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.

After the application is installed, launch it and select the sleep period first, the field node AID address next, and then the COM port assigned to the IM1800 or IM1800-1 USB

port. *COM 5* is shown in the example in Figure 6. The *Units* menu selects Fahrenheit or Celsius for the temperature display.

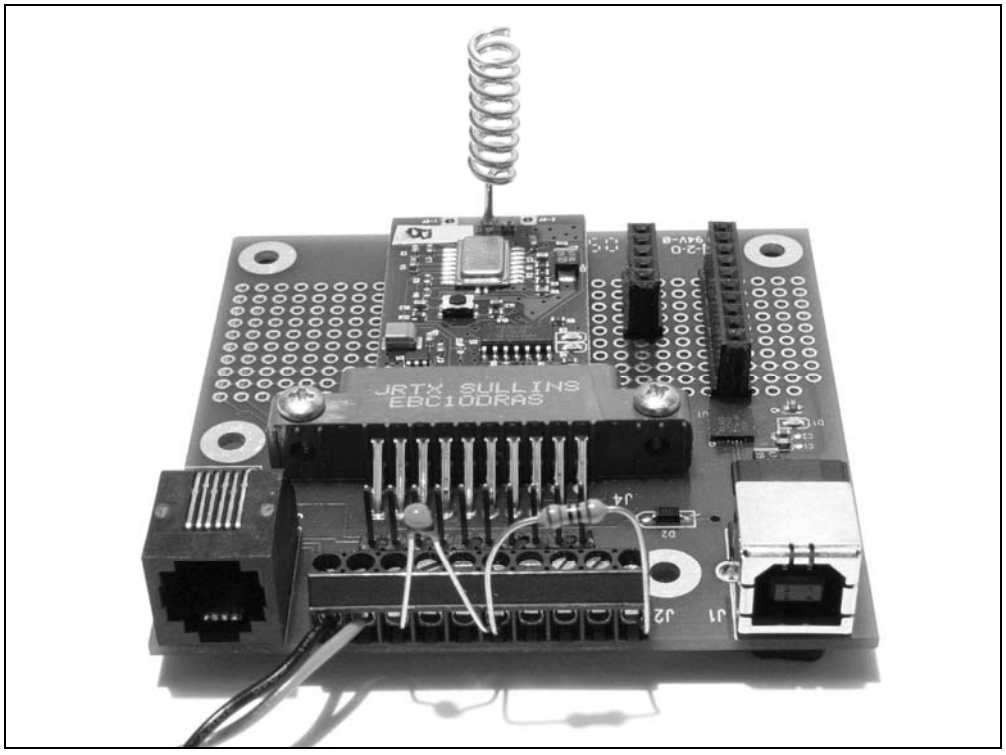


Figure 5: IM1800 Thermistor-Resistor Installation

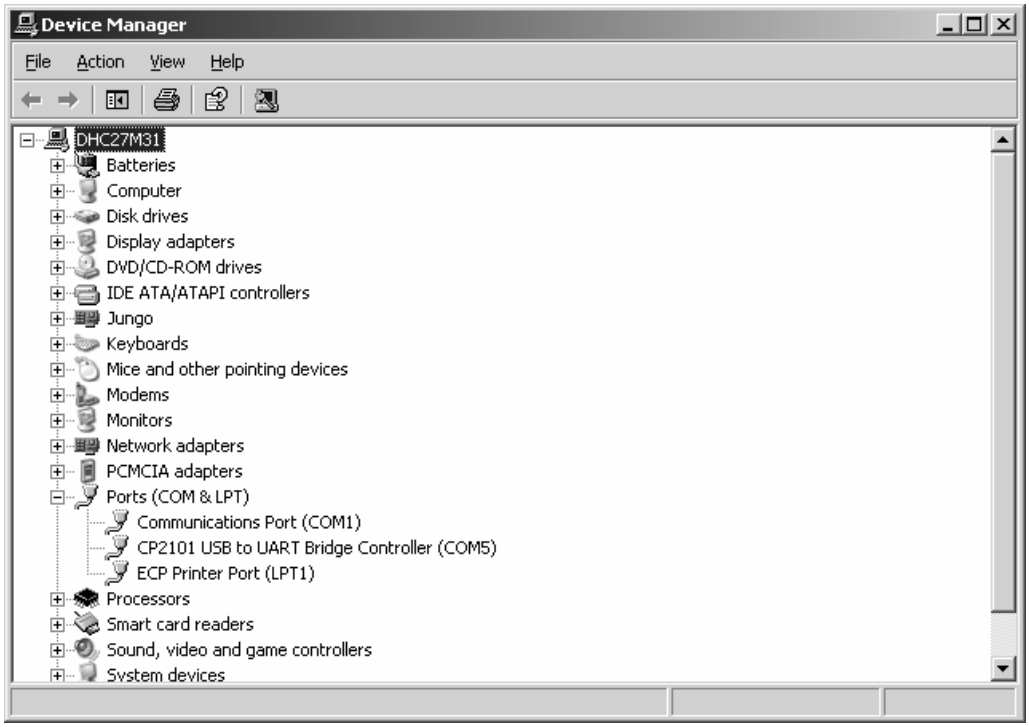


Figure 6: COM Port Verification

DM1800/DM1810 Sleep Cycle Demonstration Source Code

```
VERSION 5.00
Object = "{648A5603-2C6E-101B-82B6-000000000014}#1.1#0"; "MSCOMM32.OCX"
Begin VB.Form Form1
   Caption         =   "DM1800/DM1810 Power Duty Cycle Demo"
   ClientHeight    =   945
   ClientLeft      =   165
   ClientTop       =   735
   ClientWidth     =   5655
   ForeColor       =   &H00000000&
   LinkTopic       =   "Form1"
   ScaleHeight     =   945
   ScaleWidth      =   5655
   StartUpPosition =   3  'Windows Default
   Begin MSCommLib.MSComm MSComm1
      Left            =   4920
      Top             =   120
      _ExtentX        =   1005
      _ExtentY        =   1005
      _Version        =   393216
      DTREnable       =   -1  'True
   End
   Begin VB.Timer Timer1
      Enabled         =   0   'False
      Interval        =   1000
      Left            =   4440
      Top             =   120
   End
   Begin VB.Shape LED
      BackColor       =   &H00000000&
      BackStyle       =   1  'Opaque
      BorderColor     =   &H80000004&
      FillColor       =   &H00008000&
      FillStyle       =   0  'Solid
      Height          =   255
      Left            =   120
      Shape           =   3  'Circle
      Top             =   240
      Width           =   255
   End
   Begin VB.Label Label1
      BackColor       =   &H00000000&
      BorderStyle     =   1  'Fixed Single
      Caption         =   " Select Sleep, Node, 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            =   480
      TabIndex        =   0
      Top             =   120
      Width           =   5055
   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 mnuNode
      Caption         =   "Node"
      Begin VB.Menu mnuNode1
         Caption         =   "Node1"
         Checked         =   -1  'True
      End
      Begin VB.Menu mnuNode2
         Caption         =   "Node 2"
      End
      Begin VB.Menu mnuNode3
         Caption         =   "Node 3"
      End
      Begin VB.Menu mnuNode4
         Caption         =   "Node 4"
      End
      Begin VB.Menu mnuNode5
         Caption         =   "Node 5"
      End
      Begin VB.Menu mnuNode6
         Caption         =   "Node 6"
      End
      Begin VB.Menu mnuNode7
         Caption         =   "Node 7"
      End
      Begin VB.Menu mnuNode8
         Caption         =   "Node 8"
      End
      Begin VB.Menu mnuNode9
         Caption         =   "Node 9"
      End
      Begin VB.Menu mnuNode10
         Caption         =   "Node 10"
      End
   End
   Begin VB.Menu mnuSleep
      Caption         =   "Sleep"
      Begin VB.Menu mnu15s
         Caption         =   "15 seconds"
         Checked         =   -1  'True
      End
      Begin VB.Menu mnu60s
         Caption         =   "60 seconds"
      End
      Begin VB.Menu mnu300s
         Caption         =   "300 seconds"
      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

' DM1800-DC07.FRM, 2007.03.27 @ 19:15 CDT
' DM1800/DM1810 V2.3 tutorial software
' Sleep cycle demo
```

```
' NO representation is made that this
' software is suitable for any purpose
' Compiled in Microsoft Visual Basic 6.0

' declare form variables:
  Dim DLE$                                          ' bi-sync DLE
  Dim STX$                                          ' bi-sync STX
  Dim ETX$                                          ' bi-sync ETX
  Dim S As Integer                                  ' TX message state variable
  Dim StPDC$                                        ' set sleep duty cycle command
  Dim DOut$                                         ' digital output high command
  Dim RdADC$                                        ' read ADC command
  Dim FIFO$                                         ' character FIFO
  Dim RPkt$                                         ' response message
  Dim AID As Byte                                   ' AID address (low byte)
  Dim ADC As Single                                 ' ADC count
  Dim TempArray(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
  Dim L As Integer                                  ' time out limit
  Dim Sto As Integer                                ' safety time out
  Dim RPFlag As Integer                             ' response pending flag
  Dim TOut As Integer                               ' response pending time out counter

Private Sub Form_Load()
' initialize form variables:
  DLE$ = Chr$(&H10)                                 ' DLE = &H10
  STX$ = Chr$(&H2)                                  ' STX = &H02
  ETX$ = Chr$(&H3)                                  ' ETX = &H03
  S = 1                                             ' set TX state to 1
  Call LoadMsg                                      ' load messages
  FIFO$ = ""                                        ' clear FIFO string
  RPkt$ = ""                                        ' clear response string
  AID = 1                                           ' default node address
  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                                             ' display default deg F
  Sto = 100                                         ' 50 second safety time out
  L = Sto                                           ' load default sleep time out
  RPFlag = 0                                        ' clear response pending flag
  TOut = 0                                          ' clear response time out counter
' initialize form and timer
  Form1.Left = (Screen.Width - Form1.Width) / 1.2   ' put form at
  Form1.Top = (Screen.Height - Form1.Height) / 5    ' upper right
  LED.FillColor = &HC000&                           ' green LED OFF
  Timer1.Interval = 500                             ' set timer interval to 0.5 s
  Timer1.Enabled = False                            ' start timer when COM port selected
End Sub

Private Sub Timer1_Timer()
  LED.FillColor = &HC000&                           ' green LED OFF
  If MSComm1.InBufferCount > 0 Then                 ' if bytes in COM input buffer
    Call GetRX                                      '   call to process RX bytes
  End If
  Call TXCmd                                        ' call to send TX command
End Sub

Public Sub TXCmd()
  If (RPFlag = 0) Then                             'if no response pending
    If (S = 1) Then                                ' if TX state 1
      RPFlag = 1                                   '   set response pending flag
      TOut = 10                                    '   initialize 5 s response time out
      L = Sto                                      '   update sleep time out limit
      Mid(StPDC$, 5, 1) = Chr$(AID)                '   update command address
      Call SndTX(StPDC$)                           '   send sleep command
      S = S + 1                                    '   increment state
    ElseIf (S = 2) Then                            ' else if TX state 2
      RPFlag = 1                                   '   set response pending flag
      TOut = 10                                    '   initialize 5 s response time out
      Mid(DOut$, 5, 1) = Chr$(AID)                 '   update command address
      Call SndTX(DOut$)                            '   send digital out high command
      S = S + 1                                    '   increment state
```

```
    ElseIf (S = 3) Then                                 ' else if TX state 3
      RPFlag = 1                                        '   set response pending flag
      TOut = 10                                         '   initialize 5 s response time out
      Mid(RdADC$, 5, 1) = Chr$(AID)                     '   update command address
      Call SndTX(RdADC$)                                '   send read ADC command
      S = S + 1                                         '   increment state
    Else                                                ' else
      S = S + 1                                         '   increment safety time out
      If (S = L) Then                                   '   if count reaches time out limit
        S = 1                                           '     set TX state back to 1
      End If
    End If
  Else                                                  ' else if response pending
    TOut = TOut - 1                                     '   decrement resp pending time out
    If (TOut = 0) Then                                  '   if time out reaches 0
      RPFlag = 0                                        '     reset response pending flag
      S = 1                                             '     reset sleep cycle
      Label1.Caption = " No response"                   '   display message
    End If
  End If
End Sub

Public Sub SndTX(Cmd$)
  Dim N As Integer                                      ' command length
  Dim M As Integer                                      ' sum
  Dim Y As Integer                                      ' loop counter
  Dim CK As Byte                                        ' checksum
  Dim C$                                                ' character buffer
  Dim GTPkt$                                            ' bi-sync framed command
  N = Len(Cmd$)                                         ' get length of command
  M = 0                                                 ' clear sum
  For Y = 1 To N                                        ' loop to add Cmd$ bytes
    M = M + Asc(Mid$(Cmd$, Y, 1))                       '   add value of each byte to sum
  Next Y
  CK = CByte(M Mod 256)                                 ' checksum is lower byte of sum
  GTPkt$ = DLE$ + STX$                                  ' start GTPkt$ with DLE$ + STX$
  For Y = 1 To N                                        ' loop to build GTPkt$ from Cmd$
    C$ = Mid$(Cmd$, Y, 1)                               '   load C$ with next Cmd$ character
    If (C$ = DLE$) Then                                 '   if character is DLE$
      GTPkt$ = GTPkt$ + C$ + DLE$                       '     add character plus DLE$ escape
    Else                                                '   else
      GTPkt$ = GTPkt$ + C$                              '     just add character
    End If
  Next Y
  If (Chr$(CK) = DLE$) Then                             ' if checksum is DLE$
    GTPkt$ = GTPkt$ + Chr$(CK) + DLE$                   '   add checksum plus DLE$ escape
  Else                                                  ' else
    GTPkt$ = GTPkt$ + Chr$(CK)                          '   just add checksum
  End If
  GTPkt$ = GTPkt$ + DLE$ + ETX$                         ' complete GTPkt$ with DLE$ + ETX$
  MSComm1.Output = GTPkt$                               ' send GTPkt$ command to read ADC
End Sub

Public Sub GetRX()
  Dim J As Integer                                      ' RPkt$ string position
  Dim K As Integer                                      ' current message length
  Dim Q As Integer                                      ' FIFO$ length
  Dim R As Integer                                      ' DLE$ flag
  Dim U As Integer                                      ' build message flag
  Dim Y As Integer                                      ' FOR loop counter
  Dim ComData$                                          ' COM input string
  Dim C$                                                ' character buffer
  RPkt$ = ""                                            ' clear RPkt$
  J = 1                                                 ' point J to first character position
  K = 0                                                 ' clear current message length
  R = 0                                                 ' clear DLE$ flag
  U = 0                                                 ' clear build message flag
  On Error Resume Next                                  ' set up COM error handler
  ComData$ = MSComm1.Input                              ' load ComData$ from COM input buffer
  FIFO$ = FIFO$ + ComData$                              ' add ComData$ to end of FIFO$ string
  Q = Len(FIFO$)                                        ' load Q with number of characters in
FIFO$
  If (Q > 256) Then                                     ' if FIFO$ is overflowing (garbage)
    FIFO$ = ""                                          '   clear FIFO$
    Exit Sub                                            '   and exit
  End If
  If (Q < 2) Then                                       ' if FIFO$ less than 2 characters
    Exit Sub                                            '   exit
  End If
```

```
  For Y = Q To 1 Step -1                                  ' else loop/scan all FIFO$ characters
    C$ = Mid$(FIFO$, J, 1)                                '   load C$ with next character
    If (R = 0) And (C$ = DLE$) Then                       '   if character is a DLE$
      R = 1                                               '     set DLE$ flag
    ElseIf (R = 1) And (C$ = STX$) Then                   '   else if DLE$ + STX$ sequence
      R = 0                                               '     clear DLE$ flag
      U = 1                                               '     set build message flag
      RPkt$ = ""                                          '     clear RPkt$
    ElseIf (R = 0) And (U = 1) Then                       '   else if build flag set and DLE$
flag clear
      RPkt$ = RPkt$ + C$                                  '     add character to message
    ElseIf (R = 1) And (U = 1) And (C$ = DLE$) Then       '   else if DLE$ + DLE$ sequence
      R = 0                                               '     reset DLE$ flag
      RPkt$ = RPkt$ + C$                                  '     add only one DLE$ to RPkt$
    ElseIf (R = 1) And (U = 1) And (C$ = ETX$) Then       '   else if DLE$ + ETX$ sequence
      R = 0                                               '     clear DLE$ flag
      U = 0                                               '     clear build message flag
      K = J                                               '     load K with message length
      If (RPkt$ <> "") Then                               '     if RPkt$ is not null
        RPFlag = 0                                        '       reset response pending flag
        If (Len(RPkt$) = 7) Then                          '       if the length is 7
          Call ChkWkUp                                    '       call/check for wake up event
        ElseIf (Len(RPkt$) = 10) Then                     '     else if the length is 10
          Call ShowT                                      '       call ShowT to show temperature
        End If
        RPkt = ""                                         '     clear RPkt$
      End If
    ElseIf (R = 1) Then                                   '   else if DLE$ sequence not here
      R = 0                                               '     clear DLE$ flag and move on
    End If
    J = J + 1                                             '   increment character position
  Next Y                                                  ' do FOR loop Q times
  If (K < Q) Then                                         ' if K is less Q after looping
    FIFO$ = Right$(FIFO$, (Q - K))                        '   load FIFO$ with remaining char's
  Else                                                    ' else
    FIFO$ = ""                                            '   clear FIFO$
  End If
End Sub

Public Sub ChkWkUp()
  Dim N As Integer                                        ' command length
  Dim M As Integer                                        ' sum
  Dim Y As Integer                                        ' loop counter
  Dim CK As Byte                                          ' checksum
' filter for valid power up event
  N = Len(RPkt$)                                          ' get response length
  M = 0                                                   ' clear sum
  For Y = 1 To (N - 1)                                    ' for all bytes but received checksum
    M = M + Asc(Mid$(RPkt$, Y, 1))                        '   add bytes to sum
  Next Y
  CK = CByte(M Mod 256)                                   ' checksum is lower byte
  If (CK <> Asc(Right$(RPkt$, 1))) Then                   ' if bad checksum
    Exit Sub                                              '   exit
  ElseIf (Asc(Mid$(RPkt$, 5, 1)) <> AID) Then             ' else if not from addressed node
    Exit Sub                                              '   exit
  ElseIf ((Asc(Mid$(RPkt$, 1, 1)) And &HF0) <> &HF0) Then  ' else if not event
    Exit Sub                                              '   exit
  ElseIf (Asc(Mid$(RPkt$, 6, 1)) And &H8F) <> &H8A Then    ' else if not wake up
    Exit Sub                                              '   exit
  End If
  LED.FillColor = &HFF00&                                 ' green LED ON
  S = 1                                                   ' reset state to 1 (send sleep cmd)
    Label1.Caption = " Node Awake"                        ' signal node is awake
End Sub

Public Sub ShowT()
  Dim N As Integer                                        ' command length
  Dim M As Integer                                        ' sum
  Dim Y As Integer                                        ' loop counter
  Dim HB As Integer                                       ' ADC high byte
  Dim LB As Integer                                       ' ADC low byte
  Dim VL As Integer                                       ' total ADC value
  Dim CK As Byte                                          ' checksum
  Dim NN$                                                 ' node number
' filter for valid 10-bit ADC response
  N = Len(RPkt$)                                          ' get response length
  M = 0                                                   ' clear sum
  For Y = 1 To (N - 1)                                    ' for all bytes but received checksum
    M = M + Asc(Mid$(RPkt$, Y, 1))                        ' add bytes to sum
  Next Y
```

```
  CK = CByte(M Mod 256)                                    ' checksum is lower byte
  If (CK <> Asc(Right$(RPkt$, 1))) Then                    ' if bad checksum
    Exit Sub                                               '   exit
  ElseIf (Asc(Mid$(RPkt$, 5, 1)) <> AID) Then              ' else if not from addressed node
    Exit Sub                                               '   exit
  ElseIf (Asc(Mid$(RPkt$, 1, 1)) And &HF0) <> &HE0 Then    ' else if not application response
    Exit Sub                                               '   exit
  ElseIf (Asc(Mid$(RPkt$, 6, 1)) <> &H9) Then              ' else if not valid response
    Exit Sub                                               '   exit
  ElseIf (Asc(Mid$(RPkt$, 7, 1)) And &HC1) <> &H41 Then    ' else if not 10-bit ADC reading
    Exit Sub                                               '   exit
  End If
' calculate and show new temperature
  NN$ = Str(Asc(Mid$(RPkt$, 5, 1)))                        ' get node AID number
  HB = Asc(Mid$(RPkt$, 8, 1))                              ' get high ADC byte
  LB = Asc(Mid$(RPkt$, 9, 1))                              ' get low ADC byte
  VL = 256 * HB + LB                                       ' calculate total ADC value
  ADC = CSng(VL)                                           ' covert ADC value to single
  LED.FillColor = &HFF00&                                  ' green LED ON
  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()
  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
' interpolate from table
  For X = 2 To 25
    If (ADC <= TempArray(X, 1)) Then
      CH = TempArray(X, 1)
      CL = TempArray((X - 1), 1)
      TH = TempArray(X, 2)
      TL = TempArray((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()
  TempArray(1, 1) = 30
  TempArray(1, 2) = -40
  TempArray(2, 1) = 41
  TempArray(2, 2) = -31
  TempArray(3, 1) = 55
  TempArray(3, 2) = -22
  TempArray(4, 1) = 74
  TempArray(4, 2) = -13
  TempArray(5, 1) = 96
  TempArray(5, 2) = -4
  TempArray(6, 1) = 124
  TempArray(6, 2) = 5
  TempArray(7, 1) = 157
  TempArray(7, 2) = 14
  TempArray(8, 1) = 196
  TempArray(8, 2) = 23
  TempArray(9, 1) = 241
  TempArray(9, 2) = 32
  TempArray(10, 1) = 290
  TempArray(10, 2) = 41
  TempArray(11, 1) = 343
  TempArray(11, 2) = 50
  TempArray(12, 1) = 398
  TempArray(12, 2) = 59
  TempArray(13, 1) = 455
```

```
  TempArray(13, 2) = 68
  TempArray(14, 1) = 512
  TempArray(14, 2) = 77
  TempArray(15, 1) = 567
  TempArray(15, 2) = 86
  TempArray(16, 1) = 619
  TempArray(16, 2) = 95
  TempArray(17, 1) = 668
  TempArray(17, 2) = 104
  TempArray(18, 1) = 712
  TempArray(18, 2) = 113
  TempArray(19, 1) = 753
  TempArray(19, 2) = 122
  TempArray(20, 1) = 788
  TempArray(20, 2) = 131
  TempArray(21, 1) = 820
  TempArray(21, 2) = 140
  TempArray(22, 1) = 847
  TempArray(22, 2) = 149
  TempArray(23, 1) = 871
  TempArray(23, 2) = 158
  TempArray(24, 1) = 892
  TempArray(24, 2) = 167
  TempArray(25, 1) = 910
  TempArray(25, 2) = 176
  TempArray(26, 1) = 925
  TempArray(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 P
  MSComm1.Settings = "9600,N,8,1"                ' at 9600 b/s
  MSComm1.RThreshold = 0                         ' poll only, no interrupts
  MSComm1.InputLen = 0                           ' read all bytes
  MSComm1.PortOpen = True                        ' open COM port
  Label1.Caption = " Valid COM:  "               ' update display
  Timer1.Enabled = True                          ' start 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 mnuNode1_Click()
  AID = 1                                        ' node 1
  Call ResetNode                                 ' clear all checkmarks
  mnuNode1.Checked = True                        ' set this checkmark
End Sub

Private Sub mnuNode2_Click()
  AID = 2                                        ' node 2
  Call ResetNode                                 ' clear all checkmarks
  mnuNode2.Checked = True                        ' set this checkmark
End Sub

Private Sub mnuNode3_Click()
  AID = 3                                        ' node 3
  Call ResetNode                                 ' clear all checkmarks
  mnuNode3.Checked = True                        ' set this checkmark
End Sub

Private Sub mnuNode4_Click()
  AID = 4                                        ' node 4
  Call ResetNode                                 ' clear all checkmarks
  mnuNode4.Checked = True                        ' set this checkmark
End Sub

Private Sub mnuNode5_Click()
  AID = 5                                        ' node 5
  Call ResetNode                                 ' clear all checkmarks
  mnuNode5.Checked = True                        ' set this checkmark
End Sub

Private Sub mnuNode6_Click()
  AID = 6                                        ' node 6
  Call ResetNode                                 ' clear all checkmarks
  mnuNode6.Checked = True                        ' set this checkmark
End Sub

Private Sub mnuNode7_Click()
  AID = 7                                        ' node 7
  Call ResetNode                                 ' clear all checkmarks
  mnuNode7.Checked = True                        ' set this checkmark
End Sub
```

```
Private Sub mnuNode8_Click()
  AID = 8                                                ' node 8
  Call ResetNode                                         ' clear all checkmarks
  mnuNode8.Checked = True                                ' set this checkmark
End Sub

Private Sub mnuNode9_Click()
  AID = 9                                                ' node 9
  Call ResetNode                                         ' clear all checkmarks
  mnuNode9.Checked = True                                ' set this checkmark
End Sub

Private Sub mnuNode10_Click()
  AID = 10                                               ' node 10
  Call ResetNode                                         ' clear all checkmarks
  mnuNode10.Checked = True                               ' set this checkmark
End Sub

Public Sub ResetNode()
  mnuNode1.Checked = False                               ' clear checkmark
  mnuNode2.Checked = False                               ' clear checkmark
  mnuNode3.Checked = False                               ' clear checkmark
  mnuNode4.Checked = False                               ' clear checkmark
  mnuNode5.Checked = False                               ' clear checkmark
  mnuNode6.Checked = False                               ' clear checkmark
  mnuNode7.Checked = False                               ' clear checkmark
  mnuNode8.Checked = False                               ' clear checkmark
  mnuNode9.Checked = False                               ' clear checkmark
  mnuNode10.Checked = False                              ' clear checkmark
End Sub

Private Sub mnu15s_Click()
  StPDC$ = Chr$(&H60) + Chr$(&H0) _
    + Chr$(&H0) + Chr$(&H0) + Chr$(&H1) + Chr$(&HF1) _
    + Chr$(&H80) + Chr$(&H40) + Chr$(&H1)                ' set sleep duty cycle command, 15 s
sleep
  Sto = 100                                              ' 50 second safety time out
  Call ResetSleep                                        ' clear sleep checkmarks
  mnu15s.Checked = True                                  ' set this checkmark
End Sub

Private Sub mnu60s_Click()
  StPDC$ = Chr$(&H60) + Chr$(&H0) _
    + Chr$(&H0) + Chr$(&H0) + Chr$(&H1) + Chr$(&HF1) _
    + Chr$(&H80) + Chr$(&H40) + Chr$(&H4)                ' set sleep duty cycle command, 60 s
sleep
  Sto = 240                                              ' 120 second safety time out
  Call ResetSleep                                        ' clear sleep checkmarks
  mnu60s.Checked = True                                  ' set this checkmark
End Sub

Private Sub mnu300s_Click()
  StPDC$ = Chr$(&H60) + Chr$(&H0) _
    + Chr$(&H0) + Chr$(&H0) + Chr$(&H1) + Chr$(&HF1) _
    + Chr$(&H80) + Chr$(&H40) + Chr$(&H14)               ' set sleep duty cycle command, 300 s
sleep
  Sto = 960                                              ' 480 second safety time out
  Call ResetSleep                                        ' clear sleep checkmarks
  mnu300s.Checked = True                                 ' set this checkmark
End Sub

Public Sub ResetSleep()
  mnu15s.Checked = False                                 ' clear checkmark
  mnu60s.Checked = False                                 ' clear checkmark
  mnu300s.Checked = False                                ' clear checkmark
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 mnuC_Click()
  mnuC.Checked = True                                    ' check deg C
  mnuF.Checked = False                                   ' uncheck deg F
  T = 1                                                  ' show deg C
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

Public Sub LoadMsg()
  StPDC$ = Chr$(&H60) + Chr$(&H0) _
    + Chr$(&H0) + Chr$(&H0) + Chr$(&H1) + Chr$(&HF1) _
    + Chr$(&H80) + Chr$(&H40) + Chr$(&H1)              ' set sleep duty cycle command, 15 s
sleep
  DOut$ = Chr$(&H60) + Chr$(&H0) _
    + Chr$(&H0) + Chr$(&H0) + Chr$(&H1) + Chr$(&H31) _
    + Chr$(&H81) + Chr$(&H0)                           ' set digital out high command
(IM1800-1)
  RdADC$ = Chr$(&H60) + Chr$(&H0) _
    + Chr$(&H0) + Chr$(&H0) + Chr$(&H1) + Chr$(&HA9)   ' read 10-bit ADC command
End Sub
```

Updated 2007.08.25