```
'************************************************************
' Caravan System Monitor for 12V Battery System              '
' Written by Doug Pankhurst May 2017, based on ideas from    '
' Tom Pankhurst and various Back Shed Forum members.         '                                    '
' Written for MMBasic V5.4.08  , copyright Geoff Graham 2012 '
' Program runs on PIC32MX470 with an SSD1930 7" touch screen '
' This program is free for anyone to use or modify as they   '
'  choose however, if you wish to re-distribute, please      '
'  acknowledge authors and contributors above.               '
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' File Version csmbe410-gui.bas  14 April 2018
' - this version moves all display function to GUI
' last known good version text 4.10  (live in caravan)
' For a detailed explanation of the program, it's
' construction and rationale, please read the
' CSM4READ.TXT file.
'************************************************************

' Initial Entry to Program
EntryPoint:
  SETTICK 0,0        ' Disable timer interrupt for now
  OPTION EXPLICIT
  OPTION AUTORUN ON

  ' Define constants for gui controls
  CONST BGround_Box = 20
  CONST Head_Box  = 51
  CONST Time_Box  = 52
  CONST Hint_1 = 53
  CONST Back_Arrow = 54 ' back arrow box
  CONST Help_Button = 55 ' question mark box
  CONST DB_Hdr_1 = 30 ' 4 small box display boxes
  CONST DB_Hdr_2 = 31
  CONST DB_Hdr_3 = 32
  CONST DB_Hdr_4 = 33
  CONST DB_Val_5 = 34
  CONST DB_Val_6 = 35
  CONST DB_Val_7 = 36
  CONST DB_Val_8 = 37
  CONST Menu_1 = 38  ' 4 lower menu selection boxes
  CONST Menu_2= 39   ' - note these have touch areas overlayed
  CONST Menu_3= 40
  CONST Menu_4= 41
  CONST Help_Box= 22 ' used for help messages
  CONST LDB_Hdr_1= 43 ' 2 large display boxes
  CONST LDB_Hdr_2= 44
  CONST LDB_Val_1= 45
  CONST LDB_Val_2= 46
  ' touch areas for display boxes
  CONST Back_Arrow_Touch = 1
  CONST Help_Button_Touch = 2
  CONST Hint_Touch = 4
  CONST Led_1 = 9
  CONST Menu_1_Touch = 5
  CONST Menu_2_Touch= 6
  CONST Menu_3_Touch= 7
  CONST Menu_4_Touch= 8

'Define variables
  ' Global Variables - initialised to default values
  DIM F_Ver = 4.10    ' File Version csmbe410-gui.bas  14 April 2018
  ' Voltage and current of caravan battery
  DIM BVolts = 12.8  '  Battery Voltage
  DIM BVArray(29)    ' Battary Volts 30 element array - 1 each 2 secs
  DIM AverageBV=12.8 ' Average battery voltage over last 60 seconds
  DIM PreviousABV=12.8 '
  DIM OldABV = 13    ' so we can calculate ABV correctly
  DIM DeltaBV = 0.1  ' Battery voltage variation over 60 seconds. The
  ' difference between the first and the last used to
  ' calculate the "Time to 30% discharge point"
  DIM Time2Discharge = 20 ' Time to 30% discharge in minutes
  DIM BatCharge = 100' Battery state of charge as a percentage where
  ' 12.75V or higher is 100%, down to 12.05V being 35% - danger level
  ' Voltage and current from Solar Panels
  DIM SVolts = 20    ' Solar Array voltage
  DIM SVArray(29)    ' Solar Volts 30 element array - 1 each two secs
  DIM AverageSV = 21 ' Average Solar Panel volts over last minute
  DIM SAmps = 0 ' current from Solar Array into D250
  DIM SAArray(29)    ' solar panel current array - 1 each two secs
  DIM AverageSC = 0  ' average solar panerl current last minute
```

```
  ' Voltage and current from towing vehicle
  DIM CVolts = 13.8  ' Vehicle battery volts
  DIM CVArray(29)    ' Car volts 30 element array  - 1 each two secs
  DIM AverageCV = 13 ' Average vehicle volts ove rlast 30 60 secs
  DIM CAmps = 0 ' current from vehicle into D250
  DIM CAArray(29)    ' car current array  - 1 each two secs
  DIM AverageCC = 0  ' average car current last minute
  ' Input (charging), output (load) and battery charge/discharge currents.
  ' Battery charge/discharge is calculated from input minus load currents.
  ' - instantaneous values, minute averaged value currents
  ' - daily accumulating input, output and battery amp hour useage
  ' - last 4 days battery charge/discharge currents and rolling total
  DIM ACAmps = 0 ' Charging current from XPS25000 240VAC charger
  DIM DCAmps = 0 ' Charging current from D250 Solar Charger
  DIM ACAArray(29)   ' XPS25000 240VAC minute current array - 1 each two secs
  DIM DCAArray(29)   ' D250 minute current array  - 1 each 2 secs
  DIM AverageACC = 0 ' average D250 output current last minute
  DIM AverageDCC = 0 ' average X25000 output current last minute
  ' Input current -  a calculated sum of AC Charger and DC charger currents
  DIM IAmps = 18  ' Charging current into battery/load
  DIM AverageIC = 18 ' Average input current
  DIM LAmps = 12 ' Load current - supplied from batteries/charging systems
  DIM LAArray(29)    ' Load Current 60 element array
  DIM AverageLC = 12 ' Average load current
  ' Battery current - a calculated value; if positive, = a charging current to
  ' the battery; if negative, equals a discharge current from the battery.
  DIM BAmps = 6     ' Input charging current minus load current
  DIM AverageBC = 6      ' Average charge/discharge current
  ' daily accumulating values
  DIM InputWH = 0        ' Input Watt Hours
  DIM LoadWH = 0         ' Load Watt Hours
  ' previous days values
  DIM PrevIWH = 12 ' yesterdays Input Current ahs
  DIM PrevLWH = 10 ' yesterdays Load Current ahs
  DIM BatWH = 0   ' Bat charge (+ value) or discharge (- value) in Whs
  DIM PrevBWH = 2 ' yesterdays Bat Current Wh
  DIM Day2BWH = 20      ' last 4 days battery Watt hour usage
  DIM Day3BWH = 30
  DIM Day4BWH = 40
  DIM TotalBWH = 90     ' rolling 4 day watt hour total usage
  ' Flags
  DIM NewDataFlag = 0   ' Two second flag - set every
  ' 2 seconds by the interrupt routine when new voltage
  ' and current values are collected.
  DIM MinFlag = 0 ' Minute flag - set once every min at 00 seconds
  ' by sec interrupt process and cleared by min processing code.
  DIM EODFlag = 0        ' End of Day Flag - set at 6AM
  DIM hrs = 6
  DIM mins = 0
  DIM secs = 0
  DIM DataVals$ LENGTH 64 '
  DIM DataValid = 1      ' 0 = bad data, anything else is good data
  ' misc variables
  DIM Help$          ' string to display in help box
  DIM ActivePage = 1 'default to main GUI page
  DIM ReEnter = 0    ' flag to control re-entry into GUI display
  DIM Ax             'loop counter
  DIM S_Bright = 5   ' initial display brightness
  ' some variables to simulate a 6Amp discharge rate
  DIM cf1 = 14      ' AverageBC  - offset of x from zero
  DIM cf2 = 0.15    ' shape of curve
  DIM cf3 = 11.3    ' 0% charge point
  DIM tbv = 13
  DIM twosecled = 1 ' LED flashes on 2 sec interrupt
  DIM Simulate = 1  ' for testing
''''''''''''''''''
' Carry out initialisation routines here - enable interrupt last
StartPoint:
  BACKLIGHT S_Bright  ' set initially for 25%, tap heading to increase
  COLOUR RGB(WHITE),RGB(WHITE)   'fg and bg - needed for page switch
  GUI INTERRUPT TouchDown ' subroutine to handle touch int
  RTC GETTIME        ' date and time from DS3231
  ' Data values come in from front end via COM1
  ' Note: Front end action could be included in this
  '  program if the system is close enough to read
  '  values directly
  OPEN "COM1:9600,128,GetVals, 1" AS #1

  ' set up any GUI controls
  InitGUI ' routine to set GUI controls
```

```
  ' preset some display boxes
  CTRLVAL(Head_Box) = "Caravan System Monitor"
  CTRLVAL(Time_Box)  = LEFT$(TIME$,5)+"     "+DATE$
  CTRLVAL(Back_Arrow) = "<" ' left arrow
  CTRLVAL(Help_Button) = "?" ' home
  ' Fill the the minute average arrays
  ' just so initial values make sense
  FOR Ax=0 TO 29
    BVArray(Ax) = 12.8
    SVArray(Ax) = 19
    CVArray(Ax) = 13
    SAArray(Ax) = 10
    CAArray(Ax) = 6
    ACAArray(Ax) = 0
    DCAArray(Ax) = 16
    LAArray(Ax) = 12
  Next Ax
  ' preset some variables
  Help$ = "Caravan System Monitor v4.10 14 April 2018~"
  Help$ = Help$ + "Created by D.Pankhurst~"
  Help$ = Help$ + "Select action from menu boxes or < to restore main page~~"
  Help$ = Help$ + "Repeatedly tap header to increase brightness"
  ctrlval(Help_Box) = Help$
  ctrlval(Hint_1) = ".... press < for large display"
  ' Display all static data
  ActivePage = 1
  ReEnter = 1
  UpdateGUIScreen  ' initial update GUI screen then only if
                   ' ReEnter is set which is only done when
                   ' a touch detected or at 2 sec processing

  ' Wait until 00 seconds to start for clean entry.
  Do While Val(Mid$(Time$,7,2)) < 58
  Loop

  ' Enable timer tick interrupts to start data collection at 00 seconds
  SetTick 2000,RequestData  ' Call every other second, send ACK to
  ' front end that initiates a data dump into COM2 from the front end.
  '
  ' end of initialisation routines
  ''''''''''''''''''''

Main:
  Do    'Main program loop, for every 2 sec interrupt, get the data, do
    '  2 second, minute and end of day processing as appropriate
    Do While NewDataFlag = 0   ' wait until data request sent
      ' from back end to front end which then generates fresh data
      ' and sends this to back end
    Loop
    ' fresh data received so now process
    TwoSecProcess  ' real time data
    If MinFlag = 1 Then
      MinProcess   ' minute average data
    EndIf
    If EODFlag = 1 Then
      EODProcess   ' end of day tidy up
    EndIf
  Loop
  ' end of main process loop

Sub TwoSecProcess
  ' entered with the following instantaneous values from front end
  ' battery volts, solar volts, solar amps, vehicle amps
  ' 240VAC charger amps, solar/car charger amps
  ' total input amps (solar/car + 240VAC), load amps
  ' battery amps (display as green charging, red discharging)
  Local Ax2  'array counter
  ' 2 second processing code - every interrupt sets 2 sec flag
  ' Push most recent values into arrays for minute averaging
  ' then set up for minute averaging
  For Ax2 = 29 To 1 Step -1  ' shuffle everyone down one spot
    BVArray(Ax2) = BVArray(Ax2-1)
    SVArray(Ax2) = SVArray(Ax2-1)
    CVArray(Ax2) = CVArray(Ax2-1)
    SAArray(Ax2) = SAArray(Ax2-1)
    CAArray(Ax2) = CAArray(Ax2-1)
    ACAArray(Ax2) = ACAArray(Ax2-1)
    DCAArray(Ax2) = DCAArray(Ax2-1)
    LAArray(Ax2) = LAArray(Ax2-1)
  Next Ax2
```

```
  BVArray(0) = BVolts ' then plug in new values to first spot
  SVArray(0) = SVolts
  CVArray(0) = CVolts
  SAArray(0) = SAmps
  CAArray(0) = CAmps
  ACAArray(0) = ACAmps
  DCAArray(0) = DCAmps
  LAArray(0) = LAmps
  ' Battery Current is IAmps-LAmps - if positive
  '  it is a charging current into battery, if negative, it is a
  '  discharging current out of battery
  IAmps = ACAmps + DCAmps  ' from both mains and solar/car chargers
  BAmps = IAmps-LAmps

  ' Convert both input, output & battery instantaneous
  '  currents into WHrs
  InputWH = InputWH + ((IAmps/1800)*BVolts)
  LoadWH = LoadWH + ((LAmps/1800)*BVolts)
  BatWH = InputWH-LoadWH
  ' Note: All AH accumulating totals are plugged into
  ' last day values then reset to zero at 6AM
  ' - now update terminal display with all values
  BatChargeUpdate  ' sub to calculate display battery charge
  UpDateGUIVals  ' display current data
  NewDataFlag = 0  ' clear new data flag ready for next interrupt
  ' End of two second processing code
End Sub

  ''''''''''''''''''''''''''''''''''
Sub MinProcess
  Local Ax1m
  ' Check for minute flag, calculate the average of the last 60 seconds of
  ' readings, plug them into the log file array at the log file array pointer
  ' calculate averages for last minute
  PreviousABV = AverageBV   ' save for discharge calculation purposes
  ' clear ready for averaging
  AverageBV = 0
  AverageSV = 0
  AverageCV = 0
  AverageSC = 0
  AverageCC = 0
  AverageACC = 0
  AverageDCC = 0
  AverageLC = 0
  For Ax1m = 0 To 29   'don't forget only sampled every 2 secs
    AverageBV = AverageBV + BVArray(Ax1m)
    AverageSV = AverageSV + SVArray(Ax1m)
    AverageCV = AverageCV + CVArray(Ax1m)
    AverageSC = AverageSC + SAArray(Ax1m)
    AverageCC = AverageCC + CAArray(Ax1m)
    AverageACC = AverageACC + ACAArray(Ax1m)
    AverageDCC = AverageDCC + DCAArray(Ax1m)
    AverageLC = AverageLC + LAArray(Ax1m)
  Next Ax1m
  ' now average
  AverageBV = AverageBV / 30
  AverageSV = AverageSV / 30
  AverageCV = AverageCV / 30
  AverageSC = AverageSC / 30
  AverageCC = AverageCC / 30
  AverageACC = AverageACC / 30
  AverageDCC = AverageDCC / 30
  AverageLC = AverageLC / 30

  AverageIC = AverageACC + AverageDCC  ' sum X250 car/solar plus
  ' X25000 mains chargers
  AverageBC = AverageIC - AverageLC    ' sum of amps in (+) and amps out(-)
'  rgUpdate         ' sub to paint out running graph values

  MinFlag = 0   ' reset the min flag
End Sub  ' End of minute code
  ''''''''''''''''''''''''''''''''''''''
  ' End of Day process - save values for historical records,
  ' tidy up and reset. Note EOD currently 0600 Hrs.
Sub EODProcess  ' Check for end of day
  ' - for new day starting at 6AM
  RTC GETTIME        ' date and time from DS3231
  PrevIWH = InputWH
  InputWH = 0
  PrevLWH = LoadWH
```

```
  LoadWH = 0
  Day3BWH = Day2BWH
  Day2BWH = PrevBWH
  TotalBWH = PrevBWH + Day2BWH + Day3BWH
  PrevBWH = BatWH
  BatWH = 0
  TotalBWH = PrevBWH + Day2BWH + Day3BWH
  EODFlag = 0    ' clear end of day flag until next 6AM
End Sub  ' End of EOD code
  '''''''''''''''''

  '''''''''''''''''''''''''''
  '  2 Second Interrupt Routine Handling to send data requestto front end
Sub RequestData      ' Timer Interrupt handler
  ReEnter = 1  ' allow GUI screen update
  ' timer = 0
  ' Send data request to front end
  Print #1,Chr$(6);     ' the ACK character
  Print "Data request sent at ",time$ 'debug use
  CtrlVal(Time_Box)  = left$(Time$,5)+"     "+Date$
  DataValid = 0  ' reset ready for new data
  'flip twosecled on-off - just an 'I'm alive' indicator
  if twosecled = 1 then
    ctrlval(Led_1) = 1
    twosecled = 0
  else
    ctrlval(Led_1) = 0
    twosecled = 1
  endif
End Sub

  '   Interrupt to receive data string from front end with all data values
Sub GetVals  ' Receive character interrupt handler
  Local BytesReady
  ' structure of DataVal$ -
  ' battery volts vv.vv, solar volts vv.v, car volts vv.vv,
  ' solar amps ii.i, car amps ii.i,
  ' ac charge amps ii.i, dc charge amps ii.i,
  ' load amps ii.i, data valid zz.z  in the form
  ' vv.vv,vv.v,vv.vv,ii.i,ii.i,ii.i,ii.i,ii.i,zz.z
  ' total 46 bytes
  pause 50    ' wait for data in buffer to stabilise
  BytesReady = Loc(#1)
  Print BytesReady,    " characters waiting"
  DataVals$ = Input$(BytesReady,#1)
  ' if the back end Tx Rx are looped, plug in data values to simulate
  ' data from the front end
  if asc(left$(DataVals$,1)) = 6 then  ' look for ACK
    print "ASCII 6 loopback character received" ' debug use
    DataVals$ = "12.52,19.5,13.85,11.5,12.6,14.1,10.1,14.8,zz.z"
  endif
  Print DataVals$   ' debug use
  ' so now extract the front end data
  BVolts = Val(Mid$(DataVals$,1,5))
  SVolts = Val(Mid$(DataVals$,7,4))
  CVolts = Val(Mid$(DataVals$,12,5))
  SAmps  = Val(Mid$(DataVals$,18,4))
  CAmps = Val(Mid$(DataVals$,23,4))
  ACAmps = Val(Mid$(DataVals$,28,4))
  DCAmps= Val(Mid$(DataVals$,33,4))
  LAmps = Val(Mid$(DataVals$,38,4))
  If Mid$(DataVals$,43,4) = "zz.z" Then
    ' very simple checksum test
    print "Data valid" ' debug use
    DataValid = 1
  Else
    DataValid = 0
  EndIf
  IAmps = DCAmps + ACAmps
  BAmps = IAmps - LAmps
  ' grab time into component variables
  hrs = Val(left$(Time$,2))
  mins = Val(Mid$(Time$,4,2))
  secs = Val(right$(Time$,2))
  ' Get the current time, extract the minutes component
  If secs = 0 Then
    MinFlag = 1
  Else
    MinFlag = 0
  EndIf
```

```
  ' Get the current time, test for 06:00
  If hrs = 6 And mins = 0 And secs = 0 Then
    EODFlag = 1
  Else
    EODFlag = 0
  EndIf
  NewDataFlag = 1 ' set to indicate fresh data
End Sub ' Returns with the following values:-
  ' BVolts = 0 to 15 equalling battery volts
  ' SVolts = 0 to 25 equalling solar Array volts
  ' CVolts = 0 to 25 equalling solar Array volts
  ' SAmps = 0 to 50 equalling 0 to 50Amps
  ' CAmps = 0 to 50 equalling 0 to 50Amps
  ' ACAmps=0 to 50 equalling 0 to 50Amps
  ' DCAmps=0 to 50 equalling 0 to 50Amps
  ' LAmps=0 to 50 equalling 0 to 50Amps
  ' plus flags set/clear
  'End of Interrupt code
  ''''''''''''''''''''

  ' Defined Subroutines
  ''''''''''''''''''''
  sub UpdateGUIScreen ' update GUI display
    CtrlVal(Time_Box)  = left$(Time$,5)+"     "+Date$
    if ReEnter = 1 then
      If ActivePage = 1 then     'startup page with help
        page 1,3
      elseif ActivePage = 2 then ' battery volts, current, time to 40%
        page 1,2
        CtrlVal(Hint_1) = "Battery Details"
        ctrlval(DB_Hdr_1) = "Battery~Volts"
        ctrlval(DB_Hdr_2) = "Battery~Amps"
        ctrlval(DB_Hdr_3) = "Load~Amps"
        ctrlval(DB_Hdr_4) = "Battery~Charge %"
      elseif ActivePage = 3 then ' running input and load watt hours
        page 1,2
        CtrlVal(Hint_1) = "Power Details"
        ctrlval(DB_Hdr_1) = "Input~WHrs Today"
        ctrlval(DB_Hdr_2) = "Load~WHrs Today"
        ctrlval(DB_Hdr_3) = "Battery~WHrs Today"
        ctrlval(DB_Hdr_4) = "Time to~Battery 40%"
      elseif ActivePage = 4 then ' watt hours over previous 4 days
        page 1,2
        CtrlVal(Hint_1) = "Power History"
        ctrlval(DB_Hdr_1) = "WattHours~Yesterday"
        ctrlval(DB_Hdr_2) = " -2 Days~WattHours"
        ctrlval(DB_Hdr_3) = " -3 Days~WattHours"
        ctrlval(DB_Hdr_4) = " 3Day Total~WattHours"
      elseif ActivePage = 5 then ' input voltage and current summary
        page 1,2
        CtrlVal(Hint_1) = "Input Summary"
        ctrlval(DB_Hdr_1) = "Solar~Volts"
        ctrlval(DB_Hdr_2) = "Car~Volts"
        ctrlval(DB_Hdr_3) = "Input~Amps"
        ctrlval(DB_Hdr_4) = "Load~Amps"
      elseif ActivePage = 6 then ' help display
        page 1,3
        ctrlval(Help_Box) = Help$
      elseif ActivePage = 7 then ' large display battery volts and charge
        page 1,7
        ctrlval(Hint_1) = " 2 second values "
        ctrlval(LDB_Hdr_1) = "Battery~Volts"
        ctrlval(LDB_Hdr_2) = "Battery~Charge %"
      endif
    endif
    ReEnter = 0
  end sub

  sub UpdateGUIVals ' update GUI display values
    if ReEnter = 1 then
      CtrlVal(Time_Box)  = left$(Time$,5)+"     "+Date$
      If ActivePage = 1 then
         ' do nothing
      elseif ActivePage = 2 then
        gui fColour rgb(black),DB_Val_5
        gui BColour rgb(80,80,255),DB_Val_5
        CtrlVal(DB_Val_5) = str$(BVolts,2,2)
        if BAmps < 0 then
          gui fColour rgb(black),DB_Val_6
          gui BColour rgb(red),DB_Val_6
```

```
      CtrlVal(DB_Val_6) = str$(BAmps,2,1)
    else
      gui fColour rgb(black),DB_Val_6
      gui BColour rgb(GREEN),DB_Val_6
      CtrlVal(DB_Val_6) = str$(BAmps,2,1)
    endif
      gui fColour rgb(black),DB_Val_7
      gui BColour rgb(red),DB_Val_7
    CtrlVal(DB_Val_7) = str$(LAmps,2,1)
      gui fColour rgb(black),DB_Val_8
      gui BColour rgb(255,255,100),DB_Val_8
    ctrlval(DB_Val_8) = str$(BatCharge)
  elseif ActivePage = 3 then
      gui fColour rgb(black),DB_Val_5
      gui BColour rgb(GREEN),DB_Val_5
    CtrlVal(DB_Val_5) = str$(InputWH,3,0)
      gui fColour rgb(black),DB_Val_6
      gui BColour rgb(red),DB_Val_6
    CtrlVal(DB_Val_6) = str$(LoadWH,3,0)
    if BatWH < 0 then
      gui fColour rgb(black),DB_Val_7
      gui BColour rgb(red),DB_Val_7
      CtrlVal(DB_Val_7) = str$(BatWH,3,0)
    else
      gui fColour rgb(black),DB_Val_7
      gui BColour rgb(GREEN),DB_Val_7
      CtrlVal(DB_Val_7) = str$(BatWH,3,0)
    endif
      gui fColour rgb(black),DB_Val_8
      gui BColour rgb(255,255,100),DB_Val_8
    ctrlval(DB_Val_8) = str$(Time2Discharge)
  elseif ActivePage = 4 then
    if PrevBWH < 0 then
      gui fColour rgb(black),DB_Val_5
      gui BColour rgb(red),DB_Val_5
    CtrlVal(DB_Val_5) = str$(PrevBWH,3,0)
    else
      gui fColour rgb(black),DB_Val_5
      gui BColour rgb(GREEN),DB_Val_5
    CtrlVal(DB_Val_5) = str$(PrevBWH,3,0)
    endif
    if Day2BWH < 0 then
      gui fColour rgb(black),DB_Val_6
      gui BColour rgb(red),DB_Val_6
    CtrlVal(DB_Val_6) = str$(Day2BWH,3,0)
    else
      gui fColour rgb(black),DB_Val_6
      gui BColour rgb(GREEN),DB_Val_6
    CtrlVal(DB_Val_6) = str$(Day2BWH,3,0)
    endif
    if Day3BWH < 0 then
      gui fColour rgb(black),DB_Val_7
      gui BColour rgb(red),DB_Val_7
    CtrlVal(DB_Val_7) = str$(Day3BWH,3,0)
    else
      gui fColour rgb(black),DB_Val_7
      gui BColour rgb(GREEN),DB_Val_7
    CtrlVal(DB_Val_7) = str$(Day3BWH,3,0)
    endif
    if Day4BWH < 0 then
      gui fColour rgb(black),DB_Val_8
      gui BColour rgb(red),DB_Val_8
    CtrlVal(DB_Val_8) = str$(TotalBWH,3,0)
    else
      gui fColour rgb(black),DB_Val_8
      gui BColour rgb(GREEN),DB_Val_8
    CtrlVal(DB_Val_8) = str$(TotalBWH,3,0)
    endif
  elseif ActivePage = 5 then
      gui fColour rgb(black),DB_Val_5
      gui BColour rgb(cyan),DB_Val_5
    CtrlVal(DB_Val_5) = str$(SVolts,2,2)
      gui fColour rgb(black),DB_Val_6
      gui BColour rgb(magenta),DB_Val_6
    CtrlVal(DB_Val_6) = str$(CVolts,2,2)
      gui fColour rgb(black),DB_Val_7
      gui BColour rgb(GREEN),DB_Val_7
    CtrlVal(DB_Val_7) = str$(IAmps,2,2)
      gui fColour rgb(black),DB_Val_8
      gui BColour rgb(red),DB_Val_8
```

```
        ctrlval(DB_Val_8) = str$(LAmps,2,2)
      elseif ActivePage = 6 then
        ctrlval(Help_Box) = Help$
      elseif ActivePage = 7 then
        CtrlVal(LDB_Val_1) = str$(BVolts,2,2)
        ctrlval(LDB_Val_2) = str$(BatCharge)
     endif
    endif
    ReEnter = 0
end sub

  '--------------------------------------------------
  ' screen touch routines for GUI interface
Sub TouchDown
  ReEnter = 1
  Select Case Touch(REF)
    Case Menu_1_Touch
      ActivePage = 2
      Print "Battery details Touch detected "
    Case Menu_2_Touch
      ActivePage = 3
      Print "Power Details Touch detected "
    Case Menu_3_Touch
      ActivePage = 4
      Print "Power History Touch detected "
    Case Menu_4_Touch
      ActivePage = 5
      Print "Input Summary Touch detected "
    Case Hint_Touch
      ActivePage = 6
      ctrlval (Hint_1) = "Be kind to your mother!"
      Print "Hint area Touch detected "
    Case Back_Arrow_Touch
      ActivePage = 7
      Print "< symbol Touch detected "
      ctrlval (Hint_1) = "Battery Voltage and State of Charge"
    Case Help_Button_Touch
      ActivePage = 6
      Print "? symbol Touch detected "
      ctrlval (Hint_1) = ".... press < for large display"
    Case Else
      ReEnter = 0
      Print "?? Touch detected "
      If S_Bright < 96 then
        S_Bright = S_Bright + 5
        backlight S_Bright
      else
        S_Bright = 0
        backlight S_Bright
      endif
  End Select
  UpdateGUIScreen
End Sub
''''''''''''
'
Sub BatChargeUpdate
  ' Calculate battery change as percentage and fill charge state graph
  ' now calculate charge and paint capacity graph
  ' 12.1V equates to 30% charge - note this is a rough calculation
  '  - it should be done under no load.
  ' *** work needed here ***
  '  - this code attemps to simulate a discharge curve for the purposes
  '    of calculating state of charge under load
  local RemainBC
  local range = 29       ' shape of discharge curve
  local offset = 10.9   ' offset to position range of values
  local curve = 6       ' shape determined by discharge current
  ' ... some rule of thumb values to position values in the
  ' correct range and with a discharge curve approximating a discharge
  ' rate of between 4 and 10 amps
'  curve = 10-(AverageLC/2.1)
  BatCharge = (((BVolts-offset)^2)*range)+ curve
  ' now set up to display charge graphically
  BatCharge = Int(BatCharge)
  If BatCharge < 1 Then BatCharge = 1
  If BatCharge > 100 Then BatCharge = 100
  ' Calculate time to 30% discharge of battery
  RemainBC = BatCharge - 30 ' rough convert from 30% - 100%
  ' to 0Ahrs to 70AHrs remaining charge
  If RemainBC < 1 Then
```

```
    RemainBC = 1
  EndIf
  AverageBC = 10
  Time2Discharge = RemainBC/AverageBC  'if we have 70 AHrs and we
  ' discharge at 10AHrs
  ' then we have 7 hours of usage till 30%
  ' - again these are only rough approximations

End Sub

  '-----------------------------------------------------
  ' GUI Setup routine - only called once but safer in a subroutine
sub InitGUI  ' initialise graphics
  gui setup 1 ' same for all pages
    GUI led Led_1,"",770,30,10,RGB(green) ' flash every 2 secs
    Font 4,2
    GUI displaybox Head_Box,2,1,796,55,RGB(black),RGB(white)
    font 6
    GUI Displaybox Time_Box,2,60,796,60,RGB(black),RGB(green)
    font 4,2
    GUI Displaybox Back_Arrow,5,418,60,60,RGB(black),RGB(white)
    GUI Displaybox Help_Button,738,418,60,60,RGB(black),RGB(white)
    font 4
    GUI Displaybox Menu_1,10,350,180,60,RGB(black),RGB(cyan)
    GUI Displaybox Menu_2,212,350,180,60,RGB(black),RGB(cyan)
    GUI Displaybox Menu_3,412,350,180,60,RGB(black),RGB(cyan)
    GUI Displaybox Menu_4,612,350,180,60,RGB(black),RGB(cyan)
    GUI displaybox Hint_1,100,418,600,60,RGB(black),RGB(white)
    CtrlVal(Head_Box) = "Caravan System Monitor"
    ctrlval(Menu_1) = "Battery~Details"
    ctrlval(Menu_2) = "Power~Details"
    ctrlval(Menu_3) = "Power~History"
    ctrlval(Menu_4) = "Input~Summary"
    ctrlval(Hint_1) = " .. waiting for minute rollover"
    ' areas for boxes above
    GUI area Back_Arrow_Touch,5,418,60,60 '
    GUI area Help_Button_Touch,738,418,60,60 '
    gui area Hint_Touch,100,418,600,60 '
    gui area Menu_1_Touch,10,350,180,60
    GUI area Menu_2_Touch,212,350,180,60
    GUI area Menu_3_Touch,412,350,180,60
    GUI area Menu_4_Touch,612,350,180,60
  gui setup 2 ' small display battery information
    font 4
    GUI Displaybox DB_Hdr_1,10,130,180,60,RGB(black),RGB(220,220,220)
    GUI Displaybox DB_Hdr_2,212,130,180,60,RGB(black),RGB(220,220,220)
    GUI Displaybox DB_Hdr_3,412,130,180,60,RGB(black),RGB(220,220,220)
    GUI Displaybox DB_Hdr_4,612,130,180,60,RGB(black),RGB(220,220,220)
    font 6
    GUI Displaybox DB_Val_5,10,200,180,60,RGB(black),RGB(magenta)
    GUI Displaybox DB_Val_6,212,200,180,60,RGB(black),RGB(green)
    GUI Displaybox DB_Val_7,412,200,180,60,RGB(black),RGB(red)
    GUI Displaybox DB_Val_8,612,200,180,60,RGB(black),RGB(yellow)
    'areas for boxes above
  gui setup 3 ' help screen
    font 2
    GUI Displaybox Help_Box,10,130,782,210,RGB(black),RGB(220,220,220)
  gui setup 4 '
  gui setup 5
  gui setup 6
  gui setup 7 ' large battery volts and battery charge
    font 4
    GUI Displaybox LDB_Hdr_1,10,130,382,60,RGB(black),RGB(220,220,220)
    GUI Displaybox LDB_Hdr_2,412,130,380,60,RGB(black),RGB(220,220,220)
    font 6,2
    GUI Displaybox LDB_Val_1,10,200,382,140,RGB(black),RGB(220,220,255)
    GUI Displaybox LDB_Val_2,412,200,380,140,RGB(black),RGB(255,230,50)
  gui setup 8 '
  ' end of GUI setups
end sub
  '-------------------------------------------------------------


  '''''''''
  ' End program.
  ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```