PRL – TN – 2007 - 89

PRL Technical Note

Real Time Data Acquisition
In Windows for
8 Channel Analog Input Module - ADAM 4017+

T.A. Rajesh

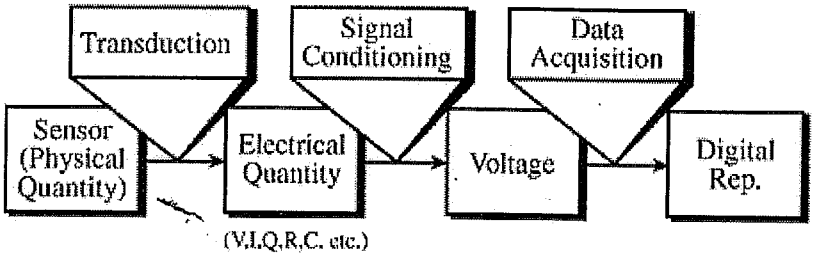
June 2007

Physical Research Laboratory
Navrangpura, Ahmedabad - 380009

Contents

Introduction

Data acquisition system involves gathering signals from measurement sources and digitizing the signal for storage, analysis, and presentation on a PC. The data acquisition (DAQ) system consists of front-end transducer and sensors, DAQ hardware & DAQ driver and application software as shown below. Scientists and engineers can choose from PCI, PXI, PCI Express, PXI Express, PCMCIA, USB, IEEE 1394, parallel, or serial ports for data acquisition in test, measurement, and automation applications.



In the day-to-day laboratory scientific activity many physical phenomenons in electrical signal are to be measured. Acquiring, processing and combining these individual signals is complex and analyze them together for some study is a bit tedious process. Therefore a data acquisition system is needed to simultaneously acquire and process these signals. Considering this, an 8-channel data acquisition system has been designed using the ADAM 4017+ DAQ hardware, which has a flexibility of the type and range of input signals. The present work uses data obtained from various analyzers like ozone, carbon monoxide, nitrous oxides, sulphur dioxide and the solar radiation sensors (Pyranometer, Pyrgeometer) of our Space and Atmospheric Sciences Division. However the data acquisition application with the hardware can be used for the similar applications and the user can interface any electrical analog single ended or differential signal either in voltage (±150mV to ±10V) or in current (±1mA to ±20mA) to the DAQ hardware as per the requirement.

The present work describes the development, implementation and usage of the data acquisition application software developed for the ADAM 4017+ DAQ hardware. The detailed description of the system and its subsystems are discussed below.

DataLog for Windows

The Datalog for Windows (DFW), developed in house, is intelligent, multifunction, high performance data acquisition application software. The DFW, running on a standard computer under Windows, provides a standard way to implement a data acquisition application without doing any windows programming. The DFW system supports 8 analog inputs presently. The analog inputs are supported with an Advantech ADAM 4017+ data acquisition module. The individual analog channel of the module is programmable within the specified input range. The user can configure the each analog channel through the DFW front-end application.

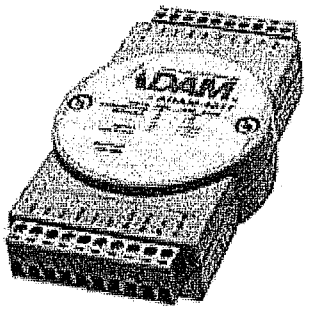
The DFW written in Microsoft Visual Basic 6.0, the system can communicate with the data acquisition module with emulation of the command-response protocol. It has the following features

a) Multi-tasking software – no data is lost during polling.
b) Multi-tasking software allows user interaction without interrupting data collection.
c) Handles 8 analog input channels, presently.
d) Measuring rate –1 sec.
e) Averaging period – user programmable.
f) Dynamic display of the instantaneous and the averaged data.
g) Data acquisition hardware configuration is password protected.
h) System logs the program selection and the error sequence.
i) System checks for the hardware status every time during acquisition.
j) Serial port auto detection at the PC.
k) Daily averaged ASCII data file generation.
l) Instantaneous data logging on selection.

The analog signal either voltage or current is physically interfaced to the ADAM 4017+ analog input channels. The data output frame is in RS-484 protocol that is converted to RS-232 standard using ADAM 4520 for a standard PC interface. The user programmable nalog channels are configured both at the hardware and software level through DFW application to activate the real time data acquisition.

8-Channel Analog Input Module – ADAM 4017+

The ADAM 4017+ is a 16-bit, 8-channel analog input module that provides programmable input ranges on all channels. The module consists of a front-end buffer, programmable gain amplifier, filter and a 8 channel multiplexer. The multiplexer output is optically coupled to an onboard microcontroller, whose serial output is converted to RS485 serial protocol. This module is an extremely cost-effective solution for industrial measurement and monitoring applications. Its opto-isolated inputs provide 3 KVDC of isolation between the analog input and the module, protecting the module and peripherals from damage due to high input-line voltages. ADAM 4017+ features signal conditioning, A/D conversion, ranging and RS-485 digital communication functions. The module protects the equipment from ground loops and power surges by providing optoisolation of A/D input and transformer based isolation up to 3 KVDC. The ADAM 4017+ uses a 16-bit microprocessor-controlled sigma-delta A/D converter to convert sensor voltage or current into digital data. The digital data is then translated into engineering



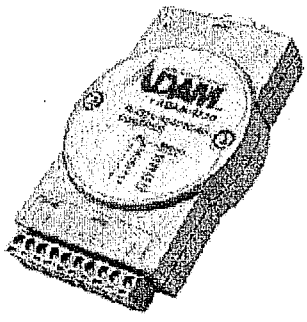
units. When prompted by the host computer, the module sends the data to the host through a standard RS-485 interface.

Specifications:

Effective Resolution: 16-bit
Channels: 6 differential, 2 single-ended
Input Type: mV, V, mA
Input Range: ±150 mV, ±500 mV, ±1 V, ±5 V, ±10 V, ±20 mA
Isolation Voltage: 3 KVDC
Fault and Over Voltage Protection: Withstands over voltage up to ±35 V
Sampling Rate: 10 samples per second (total)
Input Impedance: 20M ohms
Bandwidth: 13.1 Hz @ 50 Hz
Accuracy: ±0.1% or better
Zero Drift: ±6 μV per degree Celsius
Communication Output: RS-485 (2-wire)
Baud Rate: 1200, 2400, 4800, 9600, 19.2K, 38.4K, 57.6K, 115.2K
Data Format: 8 data bits, no parity, 1 stop bit (asynchronous)
Operating Temperature: -10 to 70°C
Power: Unregulated 10-30 VDC (1.2 watts)

Isolated RS-232 to RS-422/485 Converter – ADAM 4520

Most standard computer systems come with standard RS-232 serial ports. Though widely accepted, RS-232 has limited transmission speed, range, and networking capabilities. The RS-422 and RS-485 standards overcome these limitations by using differential voltage lines for data and control signals. The ADAM 4520 converter lets you take advantage of RS-422 and RS-485 on systems originally equipped with RS-232. It transparently converts RS-232 signals into isolated RS-422 or RS-485 signals. The ADAM 4520 builds an industrial grade, long distance communication system with standard PC hardware. It extends the communication distance by1.2 km or increases the maximum number of connected nodes by 32. The RS-485 standard supports half-duplex communication. This means that just two wires are needed to both transmit and receive data. ADAM 4520 is equipped with an I/O circuit which can automatically sense the direction of the data flow. No handshaking with the host (like RTS, Request to Send) is necessary to receive data and forward it in the correct direction. Any software written for half-duplex RS-232 with an ADAM network without modification can be used.



Specifications:

Isolation Voltage: 0.5 KVDC
Communication to host Output: RS-485 (2-wire)
Baud Rate: 1200, 2400, 4800, 9600, 19.2K, 38.4K, RTS control
Data Format: 8 data bits, no parity, 1 stop bit (asynchronous)
Communication speed: 1200, 2400, 4800, 9600, 19.2KBPS
Max. Communication distance: 1.2 KM
Multidrop module: 256 per serial port
Communication to DTE: RS-232
Operating Temperature: -10 to 70°C
Power: Unregulated 10-30 VDC (1.2 watts), protected against power reversal

Implementation and Layout

The DFW graphical user interface was designed in Visual Basic (VB6.0), an event-driven Object-Oriented programming language. The physical data acquisition from the analog input module is implemented via 'timer' event of the VB programming language. The various parameters like Station name, channel header, channel units and channel format are accessed from an ASCII file into the respective modules; this is done using the intrinsic file input-output features of the VB. The individual analog channel configurations are saved in an ACSII file for the next time operation. The various screen shots are shown in appendix 'B'. The DFW has six tab-based module as described below

1. **Setup**: In this module the parameters like station name, serial port ID, serial port parameters, ADAM 4017+ ID, Data folder path and Data averaging time can be set as per the user requirement.

2. **Channel**: In this module user can select the number of ADAM 4017+ channels, channel header, channel units, channel format, channel full scale value, channel voltage/current range and the channel offset if any as per the requirement.

3. **Edit**: The following parameters; station name, channel header, channel unit and channel format can be appended into the corresponding file through this module.

4. **Adam Config**: The individual analog channels of the ADAM 4017+ current input voltage/current range cab be read and set as per the requirement. The selection of this module is password protected.

5. **Data**: This module shows the active ADAM 4017+ channel headers with the station name, data averaging time and the data filename. The user can start the real time data acquisition simply by clicking the green colour [Start] button. Once the module is in active mode it displays the current status also.

6. **Average Data**: This module shows the each channel averaged data in the time domain since 00:00 HRS or the start of the data acquisition.

The interface also has the following menu for the ease of operation

1. **ADAM**: The Adam Utility software (V 2.00) from Advantech can be accessed through this menu, this utility program helps in the ADAM 4017+ individual channel zero and span calibration.
2. **Refresh**: In order to take the changes made in Setup module or Adam Config module to be effective, this menu has to be selected after the setup operation.
3. **Exit**: The application can be properly terminated though this option.

DFW Application – case study

Surface Tracegas Analyser Interface

In the Surface TraceGas Monitoring (STGM) Lab, various tracegases like O_3, CO, NO, NO_x, SO_2 and H_2S being monitored round the clock. These analysers aspires the ambient

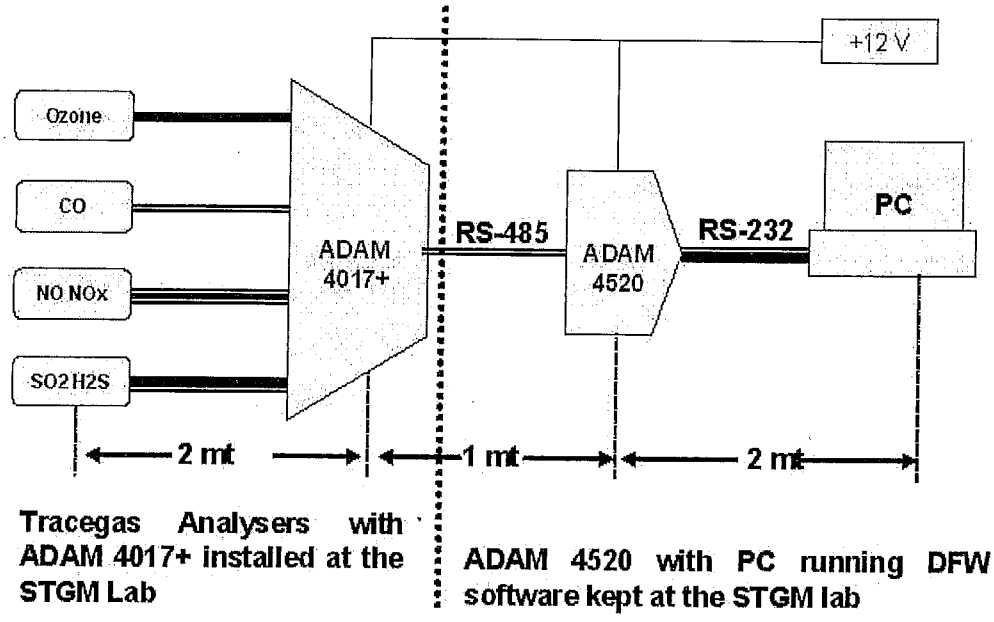


Figure 1: Schematic of Analysers data acquisition hardware interface

air, the microprocessor on board computes the tracegas concentration, displays the

measured concentration and simultaneously gives the analog equivalent of the measured quantity at its physical analog output port. The analog output ports of these analysers are

Instrument	Instrument FS	Instrument Range	4017+ Range	DFW FS	DFW Range	DFW Offset
Ozone Analyser	10 V	1000 PPB	10 V	10	1000	0
CO Analyser	10 V	5000 PPB	10 V	10	5000	0
NO/NOx	1 V	100 PPB	1 V	1	100	0
Analyser	1 V	100 PPB	1 V	1	100	0
SO2/H2S	10 V	100 PPB	10 V	10	100	0
Analyser	10 V	100 PPB	10 V	10	100	0

Table 1:Instrument, ADAM 4017+ & DFW, FS & Range values

physically connected to ADAM 4017+ via a shield cable. The schematic of the surface trace gas analyser interface is shown in figure 1. The ADAM 4017+ and DFW channels are configured with reference to the input fullscale voltage and the respective range, as shown in table 1. In the DFW channel offset were taken as null value this is to nullify the constant offset between the analyzer displayed value and the DFW displayed value.

Solar Radiation Instrument Interface

The solar radiation irradiance which results from the direct solar radiation and from the diffuse radiation incident from the hemisphere above, measuring instruments are installed at the terrace of the PRL main building. Each instrument measures the solar radiation in

Instrument	Instrument Sensitivity	4017+ Range	DFW FS	DFW Range	DFW Offset
Pyrgeo	13.22	150 mV	0.1	13.22	0
Pyrano (T)	10.86	150 mV	0.1	10.86	0
Pyrano (D)	9.51	150 mV	0.1	9.51	0
Pyrgeo-Temp	1	10 V	1	1	0

Table 2: Instrument, ADAM 4017+ & DFW, FS & Range values

the respective band of wavelength and gives the analog output of the measured value via a shield cable of limited length (10 mt). Due the physical separation between the instruments and the PC is more than 10 mt, the ADAM 4017+ is installed near to these

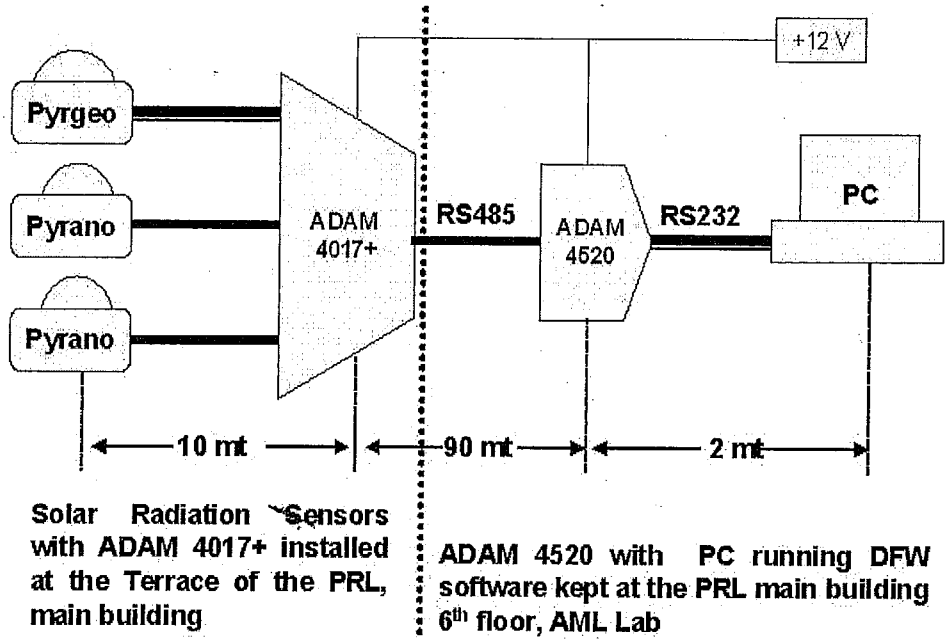


Figure 2: Schematic of Solar sensors data acquisition hardware interface

instruments at the terrace. The schematic of these instruments interface is shown in figure 2. The ADAM 4017+ and DFW channels are configured with reference to the respective sensitivity, as shown in table 2.

Summary & Future Augmentation

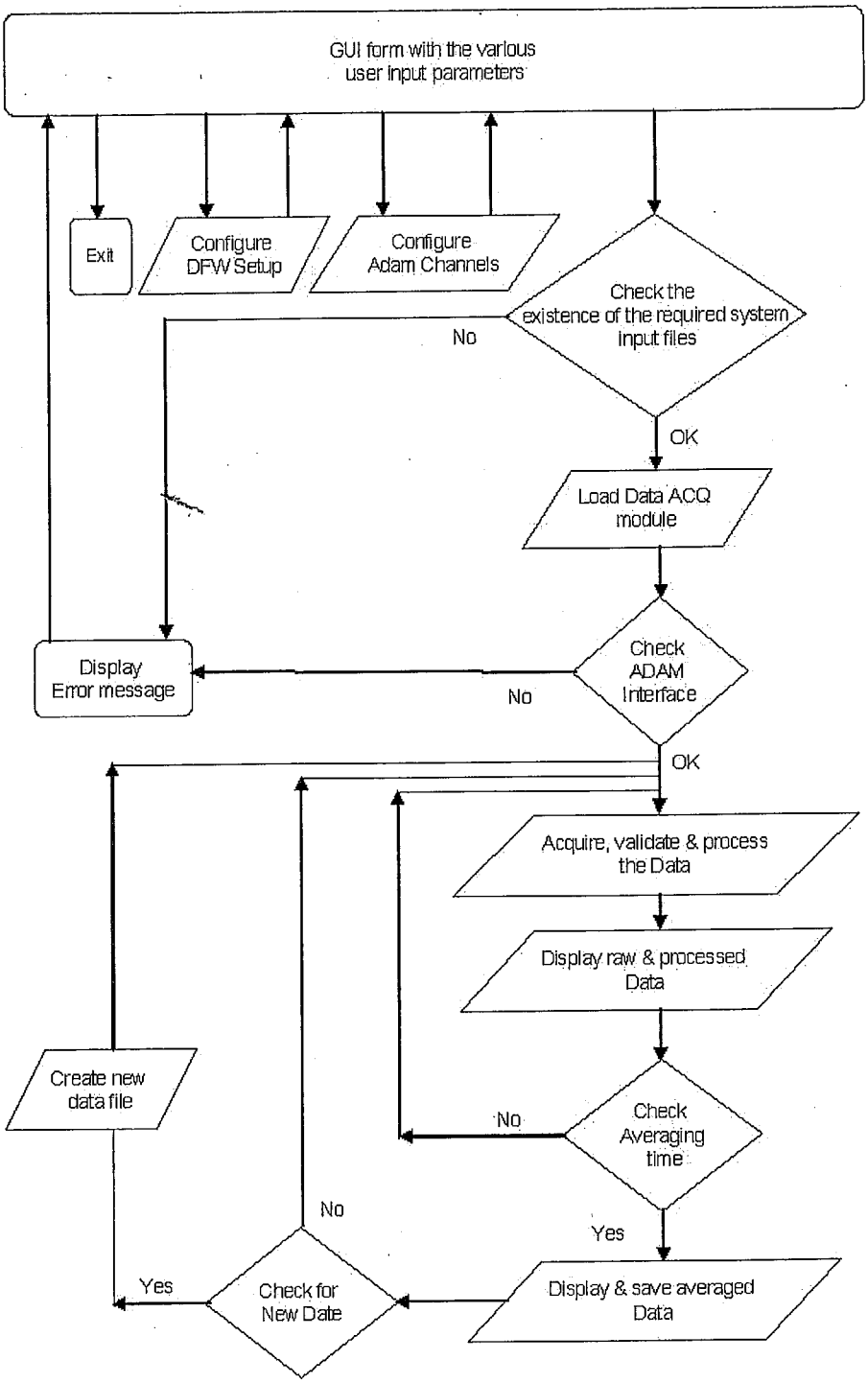
The above described DFW software with the data acquisition hardware can be used for the real time data acquisition of either voltage or current analog signal in the following range: ±150 mV, ±500 mV, ±1 V, ±5 V, ±10 V and ±20 mA only. The user can configure the analog channel as per the requirement from DFW. The data can be logged for the instantaneous or the averaged value, as programmed by the user. The program checks for the hardware status every time and creates a new file at 12:00 AM daily. Apart from the data logging it also logs each and every program activity with the error sequence. It is a user-friendly windows package to be used with ADAM 4017+ only for any mode of data acquisition within the described hardware range. The program in the distribution form is available with the author for any user.

The DFW in its present form can be interfaced to a single ADAM 4017+ only, as ADAM 4520 can support up to 256 multidrop modules, the DFW can be upgraded in future to handle multi ADAM modules. The DFW can be upgraded to display the real time plot of the each analog channel. The DFW can be programmed for hosting the real time data over Ethernet.

Acknowledgement

I am thankful to all the members of the Space and Atmospheric Sciences Division of PRL who have helped directly or indirectly in carrying out this work on data acquisition system.

Appendix: B – Flowchart



Appendix: B – Screen Shots

Surface Tracegas Analysers Interface

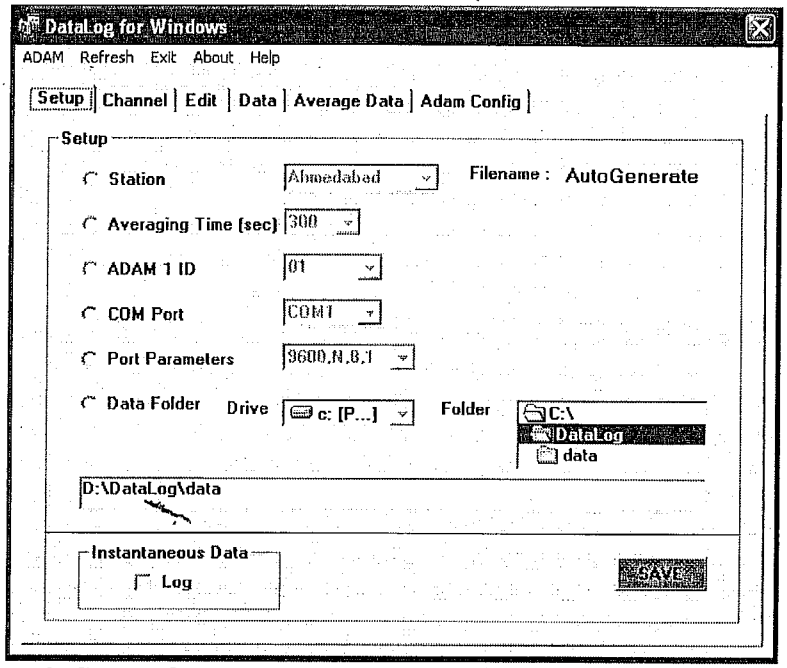


Figure 3: Screen for setting acquisition parameters in the 'Setup' module

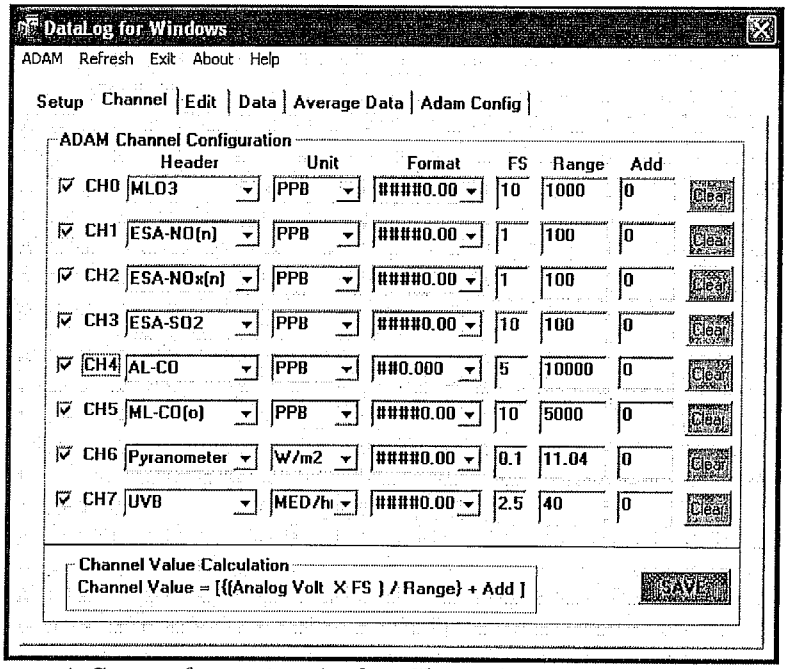


Figure 4: Screen for setting Analog Channels in the 'Channel' module

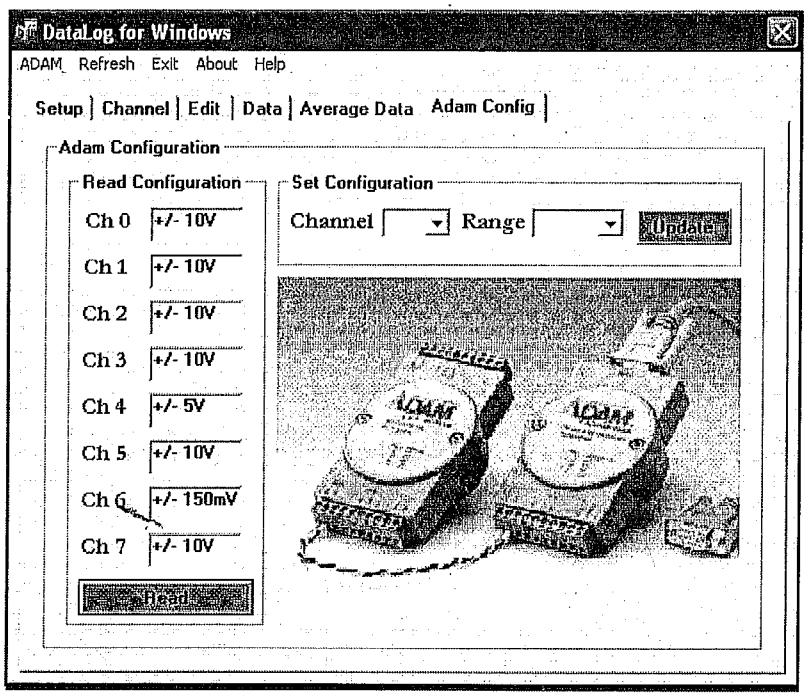


Figure 5: Screen for configuring ADAM 4017+ channels range

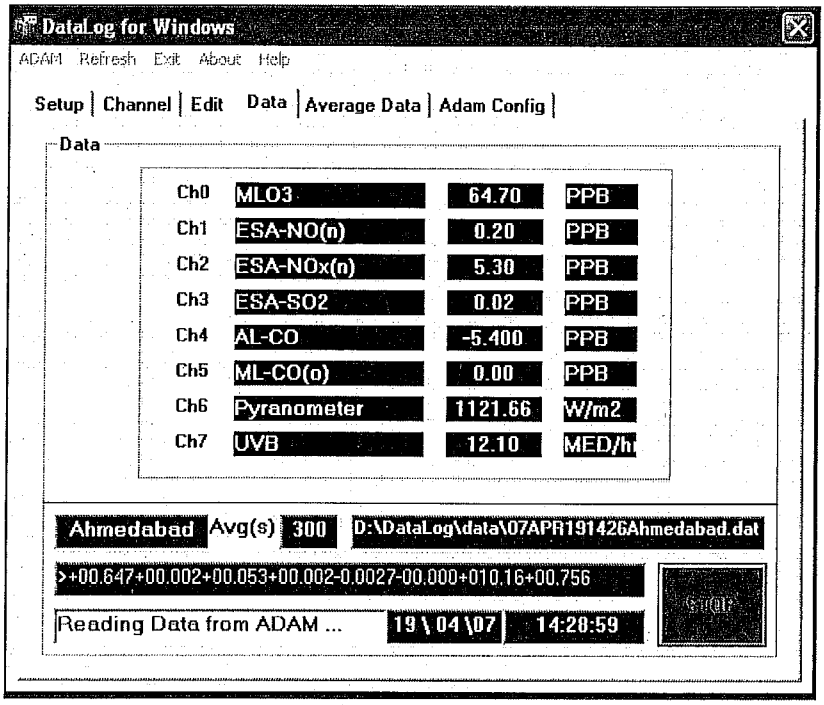


Figure 6: Screen for Real time acquisition mode

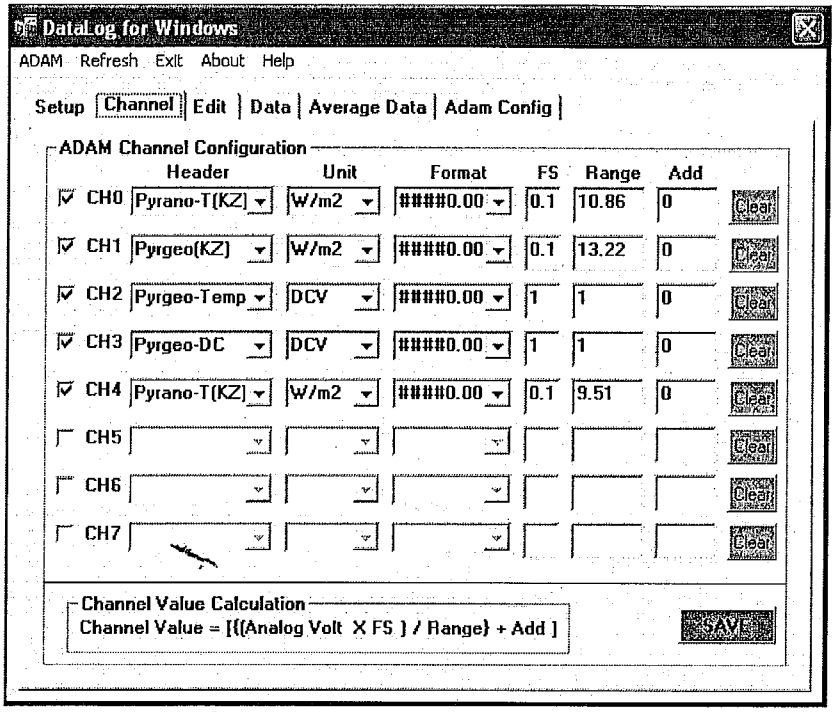


Figure 7: Screen for setting Analog Channels in the 'Channel' module

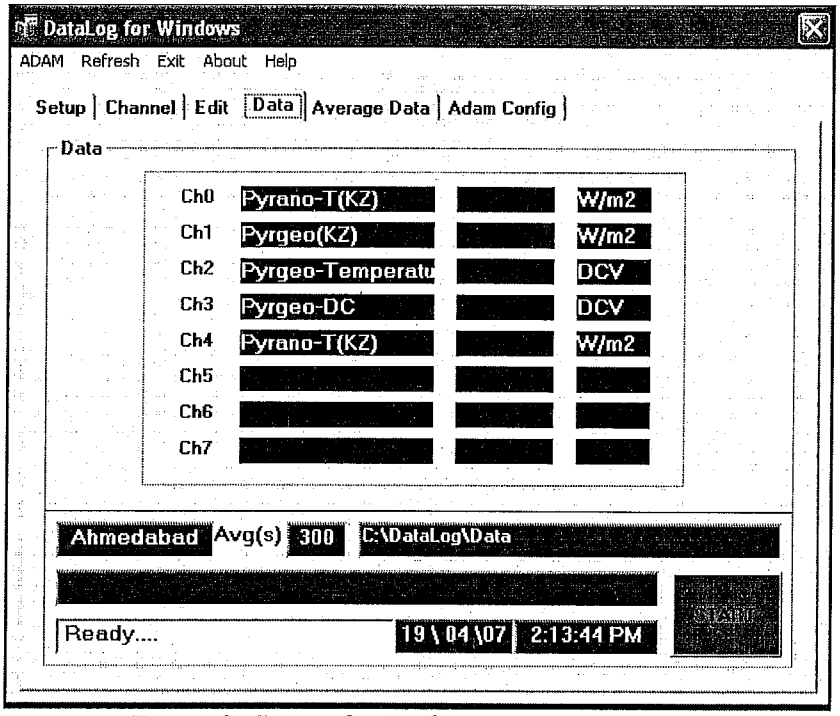


Figure 8: Screen for Real time acquisition mode

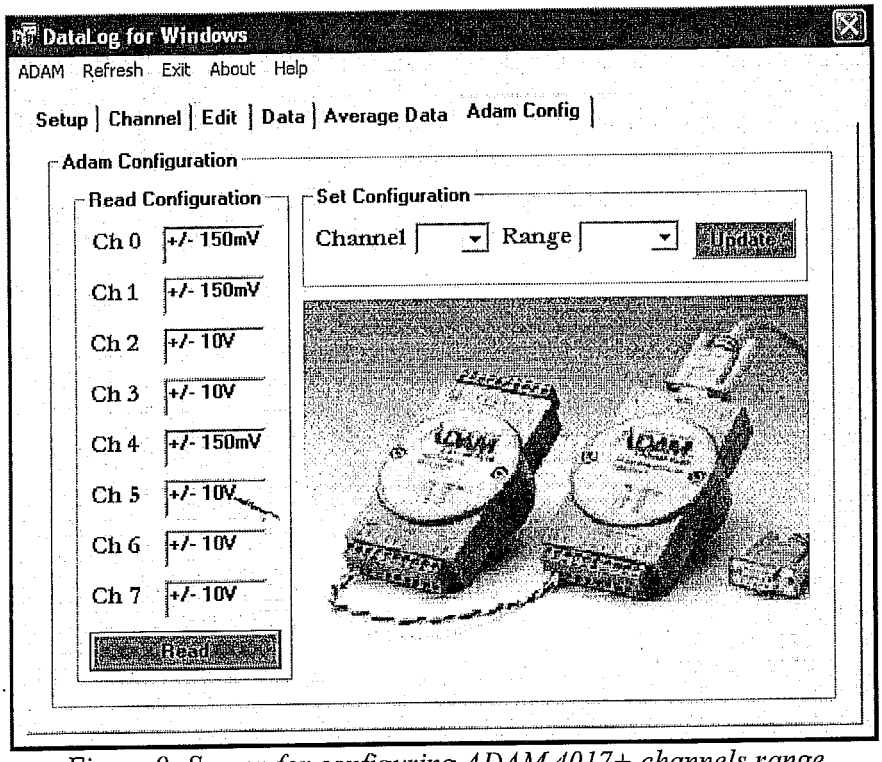


Figure 9: Screen for configuring ADAM 4017+ channels range

Appendix: C – VB6.0 Source code

```
**********module.bas**********
Option Explicit
Public fileName, channel_header(1 To 40), station_name(1 To 20), data, sData As String
Public rawdata, header, test, blink, error, counter, length, adam  As Integer
Public timesec, no_of_data  As Single
Public ch_value(8), ch_sum(8), ch_avg(8) As Double
Public adam_data(8), ch_fs(8), data_range(8), offset(16)  As Single
Public units(1 To 20), formats(1 To 20), file_data, filedata2(1 To 10), data_format(1 To 20) As String
Public data0, data1, data2, data3, data4, data5, data6, data7, path, errmsg As String
Public setup_data1, setup_data2, setup_data3, setup_data4, setup_data5, setup_data6, setup_data7,
setup_data8, setup_data9, setup_data10 As String
Public retval, ch_file, config1, config2, config3  As String
Public hfile, intComPortExist As Long
Public Declare Function Beep Lib "kernel32" (ByVal dwFreq As Long, ByVal dwDuration As Long) As
Long
Public LoginSucceeded As Boolean

Type DCB
    DCBlength As Long
    BaudRate As Long
    fBitFields As Long
    wReserved As Integer
    XonLim As Integer
    XoffLim As Integer
    ByteSize As Byte
    Parity As Byte
    StopBits As Byte
    XonChar As Byte
    XoffChar As Byte
    ErrorChar As Byte
    EofChar As Byte
    EvtChar As Byte
    wReserved1 As Integer
End Type

Type COMMCONFIG
    dwSize As Long
    wVersion As Integer
    wReserved As Integer
    dcbx As DCB
    dwProviderSubType As Long
    dwProviderOffset As Long
    dwProviderSize As Long
    wcProviderData As Byte
End Type
Declare Function GetDefaultCommConfig Lib "kernel32" _
 Alias "GetDefaultCommConfigA" (ByVal lpszName As String, _
        lpCC As COMMCONFIG, lpdwSize As Long) As Long
"This function returns a zero value if the port does not exist. *
Public Function EnumComPorts(port As Integer) As Long
  Dim cp As COMMCONFIG, cpsize As Long
  cpsize = LenB(cp)     'gets the size of COMMCONFIG structure
  EnumComPorts = GetDefaultCommConfig("COM" + Trim(Str(port)) + _
```

```
                    Chr(0), cp, cpsize)
End Function
**********mainform********** START**********
Private Sub Command17_Click() '********** START**********
start = 1
header = 0
start_time = Timer
Label43.Visible = True
e1.Enabled = False
r1.Enabled = False
ad1.Enabled = False
a1.Enabled = False
h1.Enabled = False
If counter <> 1 Then
   k = 1
   Command17.BackColor = &HFF&
   Command17.Caption = "STOP"
   counter = 1
   MSComm1.PortOpen = True
   Timer1.Enabled = True
   hfile = FreeFile
   Open setup_data9 + "\log.txt" For Append As hfile
   Print #hfile, Format(Date$, "dd-mm-yyyy"); " "; Time$, "Programme Started"
   Close hfile
  Else
    start = 0
    k = 2
    k1 = 2
    Command17.BackColor = &HC000&
    Command17.Caption = "START"
    e1.Enabled = True
    r1.Enabled = True
    ad1.Enabled = True
    a1.Enabled = True
    h1.Enabled = True
   counter = 0
   Label43.Caption = "Data Logging Halted !!!"
    If k1 = 2 Then
            Command17.BackColor = &HC000&
            Command17.Caption = "START"
            counter = 0
            MSComm1.PortOpen = False
             Timer1.Enabled = False
    End If
    hfile = FreeFile
    Open setup_data9 + "\log.txt" For Append As hfile
    Print #hfile, Format(Date$, "dd-mm-yyyy"); " "; Time$, "Programme Stoped"
    Close hfile
    End If
    fileName = setup_data9 + "\" + Mid$(Date$, 9, 2) + UCase(Format(Date, "MMM")) + Mid$(Date$, 4, 2)
     + Mid$(Time$, 1, 2) + Mid$(Time$, 4, 2) + Trim$(setup_data1) + ".dat"
     Label12.Caption = fileName
     End Sub
```

```
Private Sub Command18_Click() '******************update adam channel**********************
Select Case Combo8.ListIndex
   Case 0
   config1 = "R08"
   Case 1
   config1 = "R09"
   Case 2
   config1 = "R0A"
   Case 3
   config1 = "R0B"
   Case 4
   config1 = "R0C"
   Case 5
   config1 = "R0D"
End Select
config2 = "$" + Trim$(setup_data3) + "7" + Trim$(Combo7.Text) + config1 + Chr$(13)
MSComm1.PortOpen = True
MSComm1.Output = config2
MSComm1.PortOpen = False
Open setup_data9 + "\log.txt" For Append As hfile
Print #hfile, Format(Date$, "dd-mm-yyyy"); " "; Time$, "Adam Channel Configuration Updated"
Close hfile
End Sub

Private Sub Command20_Click() '**********************edit *********************************
hfile = FreeFile
If Text1(0).Text <> "" Then
   Open "c:\DataLog\station.txt" For Append As hfile
   Print #hfile, Text1(0).Text '1(2)
   Close hfile
Else
   MsgBox "Empty Station name !!!"
End If
hfile = FreeFile
Open setup_data9 + "\log.txt" For Append As hfile
Print #hfile, Format(Date$, "dd-mm-yyyy"); " "; Time$, "New Station Added"
Close hfile
End Sub

Private Sub Command5_Click() '***************SAVE-Channel configuration********************
hfile = FreeFile
Open setup_data9 + "\log.txt" For Append As hfile
Print #hfile, Format(Date$, "dd-mm-yyyy"); " "; Time$, "Channel Configuration Saved"
Close hfile
Dim k1 As Integer
k1 = 0
For k = 24 To 31
If Combo1(k) = "" Then
   k1 = k1 + 1
End If
Next k
hfile = FreeFile
If k1 <> 8 Then
   Open "c:\DataLog\adam_channel.txt" For Output As hfile
   For k = 24 To 31
      If Combo1(k) <> "" Then
```

```
        Write #hfile, k, Check1(k - 16).Value, Combo1(k).Text, Combo1(k + 8).Text, Combo1(k + 16).Text,
Text1(k - 8).Text, Text1(k).Text, Text1(k - 16).Text
End If
Next k
Else
   MsgBox "Empty Channel Header Data"
End If
Close hfile
End Sub

Private Sub Command8_Click() '**********************read adam config**********************
MSComm1.PortOpen = True
For i = 0 To 7
config2 = "$" + Trim$(setup_data3) + "8" + "C" + Trim(Str(i)) + Chr$(13)
MSComm1.Output = config2
sData$ = vbNullString
Do
   sData$ = sData$ & MSComm1.Input
   On Error Resume Next
Loop Until InStr(sData$, vbCr) 'Or Timer - start_time > 0.08
Select Case Mid(sData$, Len(sData$) - 3, 3)
Case "R08"
Label3(i).Caption = "+/- 10V"
Case "R09"
Label3(i).Caption = "+/- 5V"
Case "R0A"
Label3(i).Caption = "+/- 1V"
Case "R0B"
Label3(i).Caption = "+/- 500mV"
Case "R0C"
Label3(i).Caption = "+/- 150mV"
Case "R0D"
Label3(i).Caption = "+/- 20mA"
End Select
Next i
MSComm1.PortOpen = False
Open setup_data9 + "\log.txt" For Append As hfile
Print #hfile, Format(Date$, "dd-mm-yyyy"); " "; Time$, "Read Adam Channel Configuration"
Close hfile
End Sub

Private Sub Form_Load() '********************load main form**********************************
'check the existence of "DataLog" dir
adam = 0
Const ATTR_DIRECTORY = 16
If Dir$("c:\DataLog", ATTR_DIRECTORY) = "" Then
MsgBox "Directory C:\DataLog doesn't exists"
End
End If
'check the existence of "Data" dir
Const ATTR_DIRECTORY1 = 16
If Dir$("c:\DataLog\Data", ATTR_DIRECTORY1) = "" Then
MkDir "c:\DataLog\data"
End If
On Error Resume Next
'Serial port detection
```

```
Dim a1 As Integer
For a1 = 1 To 16 ' VB6 supports maximum 16 physical serial ports
   intComPortExist = EnumComPorts(a1)
   If intComPortExist > 0 Then
      Combo5.AddItem "COM" & Trim(Str(a1))
   End If
Next a1
'Checks the multiple invocation of the application
If App.PrevInstance Then
   MsgBox "Application already in use ..."    ' Flag Error to User
   End                                        ' End the Application
End If
header = 0
error = 0
k = 1
olddate = Date$
Label41.Caption = Mid$(olddate, 4, 2) + " \ " + Left$(olddate, 2) + " \" + Right$(olddate, 2)
Label43.Caption = " Ready...."
Label42.Caption = Time
start_time = Timer
'Call initialize
hfile = FreeFile
'--check the existance of DataLog.txt file---
retval = Dir$("c:\DataLog\stgm.txt")
If retval = "stgm.txt" Then
   Open "c:\DataLog\stgm.txt" For Input As hfile
   '----------Station------average----adam 1id-----adam com port--Adam setings--data folder----data log---
   Input #hfile, setup_data1, setup_data2, setup_data3, setup_data5, setup_data6, setup_data9,
setup_data10
   Close hfile
Else
   MsgBox "stgm.TXT file doesn't exists !!!"
End If
If setup_data10 = 1 Then
Label1(13).Caption = "Instantaneous Data Logging is enabled"
End If
hfile = FreeFile
Open setup_data9 + "\log.txt" For Append As hfile
Print #hfile, Format(Date$, "dd-mm-yyyy"); " "; Time$, "Programme loaded"
Close hfile
Label6.Caption = setup_data1
Label12.Caption = setup_data9
Label8.Caption = setup_data2
length = 0
'--check the existance of adam_channel.txt file---
hfile = FreeFile
retval = Dir$("c:\DataLog\adam_channel.txt")
If retval = "adam_channel.txt" Then
   Open "c:\DataLog\adam_channel.txt" For Input As hfile
   Do Until EOF(hfile)
      length = length + 1 ' to detect number of the channels used
      Input #hfile, data1, data0, data2, data3, data4, data5, data6, data7
      '[data1- ch_number], [data0- checkbox] [data2- Header] [data3- units] [data4- format] [data5- fs]
[data6- range] [data7-Add]
      hdr_disp(length) = data1
      hdr2(length) = data0
```

```
Select Case Val(data1)
  Case 24
        If data0 = 1 Then
        file_no(1) = data1
        file_hdr(1) = data2
        Form3.Label15.Caption = data2
        Form3.Label18.Caption = data3
        ch_fs(1) = Val(data5)
        data_format(1) = data4
        data_range(1) = Val(data6)
        offset(1) = Val(data7)
        End If
   Case 25
        If data0 = 1 Then
        file_no(2) = data1
        file_hdr(2) = data2
        Label46.Caption = data2
        Label47.Caption = data3
        ch_fs(2) = Val(data5)
        data_format(2) = data4
        data_range(2) = Val(data6)
        offset(2) = Val(data7)
        End If
   Case 26
        If data0 = 1 Then
        file_no(3) = data1
        file_hdr(3) = data2
        Label49.Caption = data2
        Label50.Caption = data3
        ch_fs(3) = Val(data5)
        data_format(3) = data4
        data_range(3) = Val(data6)
        offset(3) = Val(data7)
        End If
   Case 27
        If data0 = 1 Then
        file_no(4) = data1
        file_hdr(4) = data2
        Label52.Caption = data2
        Label53.Caption = data3
        ch_fs(4) = Val(data5)
        data_format(4) = data4
        data_range(4) = Val(data6)
        offset(4) = Val(data7)
        End If
   Case 28
        If data0 = 1 Then
        file_no(5) = data1
        file_hdr(5) = data2
        Label55.Caption = data2
        Label56.Caption = data3
        ch_fs(5) = Val(data5)
        data_format(5) = data4
        data_range(5) = Val(data6)
        offset(5) = Val(data7)
        End If
```

```
        Case 29
                If data0 = 1 Then
                file_no(6) = data1
                file_hdr(6) = data2
                Label58.Caption = data2
                Label59.Caption = data3
                ch_fs(6) = Val(data5)
                data_format(6) = data4
                data_range(6) = Val(data6)
                offset(6) = Val(data7)
                End If
        Case 30
                If data0 = 1 Then
                file_no(7) = data1
                file_hdr(7) = data2
                Label61.Caption = data2
                Label62.Caption = data3
                ch_fs(7) = Val(data5)
                data_format(7) = data4
                data_range(7) = Val(data6)
                offset(7) = Val(data7)
                End If
        Case 31
                If data0 = 1 Then
                file_no(8) = data1
                file_hdr(8) = data2
                Label64.Caption = data2
                Label65.Caption = data3
                ch_fs(8) = Val(data5)
                data_format(8) = data4
                data_range(8) = Val(data6)
                offset(8) = Val(data7)
                End If
    End Select
  Loop
  Close hfile
Else
  MsgBox "ADAM_CHANNEL.TXT file doesn't exists !!!"
End If
Timer1.Enabled = False
sData = vbNullString
data = vbNullString
MSComm1.CommPort = Trim$(Right$(setup_data5, 1))
MSComm1.Settings = setup_data6
MSComm1.InputLen = 0
MSComm1.RThreshold = 0 '1 for interrupt
MSComm1.DTREnable = False
End Sub

Private Sub TabStrip1_Click() '********************Tab selection*******************************
Select Case TabStrip1.SelectedItem.Index
Case 1:
  hfile = FreeFile
  Open setup_data9 + "\log.txt" For Append As hfile
  Print #hfile, Format(Date$, "dd-mm-yyyy"); " "; Time$, "Setup Tab Activated"
  Close hfile
```

```
    If start = 0 Then
    Frame1.Visible = True
    Frame1.ZOrder 0 'Setup
    Call tab_setup
    Else
    Frame1.Visible = False
    End If
Case 2:
    hfile = FreeFile
    Open setup_data9 + "\log.txt" For Append As hfile
    Print #hfile, Format(Date$, "dd-mm-yyyy"); " "; Time$, "Channel Tab Activated"
    Close hfile
    If start = 0 Then
    Frame2.Visible = True
    Frame2.ZOrder 0 'Channel
    Call tab_channel
    Else
    Frame2.Visible = False
    End If
Case 3:
    hfile = FreeFile
    Open setup_data9 + "\log.txt" For Append As hfile
    Print #hfile, Format(Date$, "dd-mm-yyyy"); " "; Time$, "Edit Tab Activated"
    Close hfile
    If start = 0 Then
    Frame17.Visible = True
    Frame17.ZOrder 0 'Edit
    Call tab_edit
    Else
    Frame17.Visible = False
    End If
Case 4:
    hfile = FreeFile
    Open setup_data9 + "\log.txt" For Append As hfile
    Print #hfile, Format(Date$, "dd-mm-yyyy"); " "; Time$, "Data Tab Activated"
    Close hfile
    Frame12.ZOrder 0 ' Data
Case 5:
    hfile = FreeFile
    Open setup_data9 + "\log.txt" For Append As hfile
    Print #hfile, Format(Date$, "dd-mm-yyyy"); " "; Time$, "Average Data Tab Activated"
    Close hfile
    Frame20.ZOrder 0 'Avg data
Case 6:
    If start = 0 Then
    adam = 2
    Load frmLogin
    frmLogin.Show
    Else
    Frame5.Visible = False
    End If
End Select
End Sub

Private Sub Timer1_Timer() '**********************Timer object**************************
If Timer - start_time > 6 Then
```

```
blinkmsg ("Error.. No Data from ADAM !!!")
End If
MSComm1.Output = "#" & setup_data3 & Chr$(13)
' adam id "#00" is not to be used it is reserved for initilisation
' keep the selection switch af ADAM-4017+ at normal position.
start_time = Timer
Do
   sData$ = sData$ & MSComm1.Input
   On Error Resume Next
Loop Until InStr(sData$, vbCr) Or Timer - start_time > 0.08
If sData$ <> "" Then
   If Len(sData$) = 58 Then
   error = 0
      Label11.Caption = sData$
      If ((Left$(sData$, 1) = ">") And (Right$(sData$, 1) = Chr(13))) Then
         For i = 1 To 8
            Call extract(sData$)
         Next i
      End If
      Call sum
      For i = 1 To length
      rta = Val(Trim$(hdr_disp(i)))
         ch_value(rta - 23) = Format((((Val(adam_data(rta - 23)) / ch_fs(rta - 23)) * (data_range(rta - 23)))
+ offset(rta - 23)), data_format(rta - 23))
         If Val(hdr2(i)) = 1 Then
         Label5(rta - 15).Caption = ch_value(rta - 23)
         End If
      Next i
   End If
   sData = vbNullString
   If olddate = Date$ Then
      Label42.Caption = " " + Time$
      blinkmsg ("Reading Data from ADAM ...")
      timesec = (Val(Left$(Time$, 2)) * 3600) + (Val(Mid$(Time$, 4, 2)) * 60) + (Val(Right$(Time$, 2)))
      If timesec Mod (Val(setup_data2)) = 0 Then
         Call mod_func
         Call write_data_file
         Call initialize0
      End If
   Else
      RichTextBox1.Refresh
      ch_file = vbNullString
      header = 0
      newdate = Date$
      olddate = newdate
      fileName = setup_data9 + "\" + Mid$(Date$, 9, 2) + UCase(Format(Date, "MMM")) + Mid$(Date$,
4, 2) + Mid$(Time$, 1, 2) + Mid$(Time$, 4, 2) + Trim$(setup_data1) + ".dat"
      Label12.Caption = fileName
   End If
Else
   blinkmsg ("Error.. No Data from ADAM !!!")
   If error = 0 Then
      hfile = FreeFile
      Open setup_data9 + "\error.txt" For Append As hfile
      Print #hfile, Format(Date$, "dd-mm-yyyy"); " "; Time$, "Error"
      Close hfile
```

```
    error = 1
  End If
  Beep 1800, 100 'frequency, duration
End If
If setup_data10 = 1 Then
Call write_data_file_instant 'write instantaneous data
End If
End Sub
Private Sub extract(a As String)
adam_data(i) = Mid$(a, 2 + (i - 1) * 7, 7)
End Sub

Private Sub sum() '**************sum data**********************************************

For i = 1 To 8
   ch_sum(i) = ch_sum(i) + adam_data(i)
Next i
no_of_data = no_of_data + 1
End Sub

Private Sub initialize() '**********************initialize**********************************

rawdata = 0
counter = 0
For i = 1 To 8
   ch_sum(i) = 0
   ch_avg(i) = 0
   ch_fs(i) = 0
   ch_value(i) = 0
Next i
no_of_data = 1
length = 0
j = 0
k = 0
k1 = 1
k2 = 0
ch_file = vbNullString
blink = 0
test = 0
error = 0
End Sub

Private Sub mod_func()
For i = 1 To 8
   ch_avg(i) = ch_sum(i) / (no_of_data)
Next i
End Sub

Private Sub write_data_file() '******************write data file*****************************

hfile = FreeFile
Open fileName For Append As hfile
If header = 0 Then
   Print #hfile, Format(Date, "DD-MMM-YYYY") + " " + Format(Time, "HH:MM:SS") + " " +
setup_data1
   Print #hfile, "TIME",
   For i = 1 To length
   rta = Val(Trim$(hdr_disp(i)))
   rta = rta - 24
```

```
    txtPassword = ""
    LoginSucceeded = True
    Me.Hide
    Shell "c:\DataLog\adam.exe", vbNormalFocus
    Case 2
    txtPassword = ""
    LoginSucceeded = True
    Me.Hide
    hfile = FreeFile
    Open setup_data9 + "\log.txt" For Append As hfile
    Print #hfile, Format(Date$, "dd-mm-yyyy"); " "; Time$, "Adam Config Tab Activated"
    Close hfile
    Form3.Frame5.ZOrder 0 ' Data
    End Select
  Else
    hfile = FreeFile
    Open setup_data9 + "\log.txt" For Append As hfile
    Print #hfile, Format(Date$, "dd-mm-yyyy"); " "; Time$, "ADAM Utility Login Failed"
    Close hfile
    lblLabels(0).Visible = False
    lblLabels(2).Visible = True
    'MsgBox "Invalid Password, try again!", , "stgm"
    txtPassword.SetFocus
    SendKeys "{Home}+{End}"
  End If
End Sub
```

Appendix: D – Package and Deployment

The DFW application contains a standalone exe file, required OCX files and the necessary input ASCII files, which have to be packaged into a setup disk for the distribution. The setup disk on execution should install the DFW application in the required folder and to copy the OCX files into the Windows/System32 folder. The DFW application packaging and distribution have been implemented using the **Inno Setup version 5.1.8** utility program.

Inno Setup is an open source script-driven installation system. It is a feature-packed installation builder. Features include a wizard interface, creation of a single EXE for easy online distribution, support for disk spanning, full uninstall capabilities, customizable setup types, integrated file compression, support for installing shared files and OCX's, and creation of Start Menu icons, INI entries, and registry entries. The script generated by the Inno setup wizard is listed below

```
; Script generated by the Inno Setup Script Wizard.
; SEE THE DOCUMENTATION FOR DETAILS ON CREATING INNO SETUP SCRIPT FILES!
[Setup]
AppName=DataLog for Windows
AppVerName=2.01A
AppPublisher=SPA-SC, PRL
DefaultDirName=c:\DataLog
DefaultGroupName=DataLog
OutputDir=C:\Setup
OutputBaseFilename=setup
Password=tracegas
Compression=lzma
SolidCompression=yes
[Languages]
Name: "english"; MessagesFile: "compiler:Default.isl"
[Tasks]
Name: "desktopicon"; Description: "{cm:CreateDesktopIcon}"; GroupDescription:
"{cm:AdditionalIcons}"; Flags: unchecked
[Files]
Source: "C:\DataLog\DataLog for Windows.exe"; DestDir: "{app}"; Flags: ignoreversion
Source: "C:\DataLog\units.txt"; DestDir: "{app}"; Flags: ignoreversion
Source: "C:\DataLog\adam.log"; DestDir: "{app}"; Flags: ignoreversion
Source: "C:\DataLog\ADAM.exe"; DestDir: "{app}"; Flags: ignoreversion
Source: "C:\DataLog\adam_channel.txt"; DestDir: "{app}"; Flags: ignoreversion
Source: "C:\DataLog\channel_header.txt"; DestDir: "{app}"; Flags: ignoreversion
Source: "C:\DataLog\formats.txt"; DestDir: "{app}"; Flags: ignoreversion
Source: "C:\DataLog\Module1.bas"; DestDir: "{app}"; Flags: ignoreversion
Source: "C:\DataLog\station.txt"; DestDir: "{app}"; Flags: ignoreversion
Source: "C:\DataLog\stgm.txt"; DestDir: "{app}"; Flags: ignoreversion
```

Source: "C:\windows\system32\MSCOMM32.OCX"; DestDir: "C:\windows\system32"; Flags: ignoreversion
Source: "C:\windows\system32\RichTx32.OCX"; DestDir: "C:\windows\system32"; Flags: ignoreversion
Source: "C:\windows\system32\COMCTL32.OCX"; DestDir: "C:\windows\system32"; Flags: ignoreversion
Source: "C:\windows\system32\DBGRID32.OCX"; DestDir: "C:\windows\system32"; Flags: ignoreversion
Source: "C:\windows\system32\COMCT232.OCX"; DestDir: "C:\windows\system32"; Flags: ignoreversion
Source: "C:\windows\system32\COMDLG32.OCX"; DestDir: "C:\windows\system32"; Flags: ignoreversion
; NOTE: Don't use "Flags: ignoreversion" on any shared system files

[Icons]
Name: "{group}\DataLog"; Filename: "{app}\DataLog.exe"
Name: "{group}\{cm:UninstallProgram,DataLog}"; Filename: "{uninstallexe}"
Name: "{userdesktop}\DataLog"; Filename: "{app}\DataLog for Windows.exe"; Tasks: desktopicon

[Run]
Filename: "{app}\DataLog for Windows.exe"; Description: "{cm:LaunchProgram,DataLog}"; Flags: nowait postinstall skipifsilent

References:

1. Serial Port Complete, Jan Axelson
2. Visual Basic 6 Programming Black Book, Steven Holzner
3. Visual Basic 6 Programming Bible, Eric A Smith and others
4. Mastering Visual Basic 6, Evangelos Petroutsos
5. Visual Basic 6 Desktop Applications, Michael Mckelvy
6. www.advantech.com/products/
7. www.msdn.microsoft.com/vbasic/
8. www.devarticles.com/c/b/Visual-Basic/
9. www.vbexplorer.com/VBExplorer/
10. www.vb-helper.com/
11. www.beyondlogic.org/serial/
12. www.lookrs232.com/com_port_programming/
13. www.members.home.nl/albartus/inno/
14. www.fredshack.com/docs/inno.html