Contents

5.1 Introduction to Universal Serial Bus (USB) ................. 122
5.2 Design Description ........................................ 124
5.3 Hardware Design ........................................... 125
  5.3.1 PCB Schematic .......................................... 125
  5.3.2 PCB Fabrication ......................................... 125
5.4 Software Design ............................................ 126
  5.4.1 Firmware Development ............................... 126
  5.4.2 Application Program Development ................. 131
5.5 Experimental Observations .................................. 136
References ..................................................... 141
5.1 Introduction to Universal Serial Bus (USB)

Universal Serial Bus (USB) is an industry standard developed in the mid-1990s, this defines the cables, connectors and communications protocols used in a bus for connection, communication and power supply between computers and electronic devices.

A group of seven companies began the development of USB in 1994: Compaq, DEC, IBM, Intel, Microsoft, NEC and Nortel. The goal was to make it fundamentally easier to connect external devices to PCs by replacing the multitude of connectors at the back of PCs, addressing the usability issues of existing interfaces, and simplifying software configuration of all devices connected to USB, as well as permitting greater data rates for external devices. The first silicon for USB was made by Intel in 1995. USB connector types and their pin connections are shown in Fig. 5.1 and Fig. 5.2 respectively.

![USB pin types](image1)

*Fig. 5.1: USB pin types*

![USB types A and B pins with their connections](image2)

*Fig. 5.2: USB types A and B pins with their connections*
USB is not a true bus, meaning only the root hub sees every signal on the bus. This implies there is no method to monitor upstream communications from a downstream device.

USB was designed to standardize the connection of computer peripherals, such as keyboards, pointing devices, digital cameras, printers, portable media players, disk drives and network adapters to personal computers, both to communicate and to supply electric power. It has become commonplace on other devices, such as smartphones, PDAs and video game consoles. USB has effectively replaced a variety of earlier interfaces, such as serial and parallel ports, as well as separate power chargers for portable devices [1].

The original USB 1.0 specification, which was introduced in January 1996, defined data transfer rates of 1.5 Mbit/s "Low Speed"and 12 Mbit/s "Full Speed". The first widely used version of USB was 1.1, which was released in September 1998. The 12 Mbit/s data rate was intended for higher-speed devices such as disk drives, and the lower 1.5 Mbit/s rate for low data rate devices such as joysticks.

The USB 2.0 specification was released in April 2000 and was ratified by the USB Implementers Forum (USB-IF) at the end of 2001. Hewlett-Packard, Intel, Lucent Technologies (now Alcatel-Lucent), NEC and Philips jointly led the initiative to develop a higher data transfer rate, with the resulting specification achieving 480 Mbit/s, a forty times increase over the original USB 1.1 specification.

The USB 3.0 specification was published on 12 November 2008. Its main goals were to increase the data transfer rate (up to 5 Gbit/s), to decrease power consumption, to increase power output, and to be backwards-compatible with USB 2.0. USB 3.0 includes a new, higher speed bus called SuperSpeed in parallel with the USB 2.0 bus. For this reason, the new version is also called SuperSpeed. The first USB 3.0 equipped devices were presented in January 2010.
5.2 Design Description

A data acquisition system (DAS) has been designed and developed using PIC18F4550 [2], which is a 40/44-Pin, high-performance, enhanced flash, USB microcontrollers with nano watt technology. Fig. 5.6 shows the schematic diagram of the system so designed. It uses 8 (eight) analog input channels (AN0 through AN07) having 10-bit resolution ADC, in which the entire operation is controlled by the firmware. A B-Type USB socket is used to communicate with the USB port of the PC. The PCB layout of the circuit is designed using DIPTrace and the DAS is fabricated.

Signal strength coming out from the temperature sensor IC LM35 and humidity sensor IC HIH 4000 are not suitable for feeding it to the DAS directly. For these, signal conditioning circuits have been designed using OpAmps OP07 and implemented as shown in Fig. 5.3. To protect the over voltage due to accidental or over range, a 5.1 volt zener diode is connected in series with a series current limiter resistance. The output is taken across the zener diode (by limiting a maximum voltage to 5.1 Volt) and fed to the input lines of the microcontroller.

![Fig. 5.3: Schematic diagram of the amplifier circuit used for signal conditioning](image)

For proper functioning of the data acquisition system, a firmware is developed and downloaded to the microcontroller, and an application program has also been developed in Visual Basic.

A firmware program is written in ‘C’ using MikroC IDE for channel selections and proper ADC conversion, then sending the digitized data to the PC through the USB.
The program is compiled and generates a hex file. The hex file so generated has been downloaded to the PIC18F4550 microcontroller using PICkit2 programmer [3]. The flowchart of the program so developed is given in Fig. 5.7.

For proper acquisition of the data by the PC, graphical display and saving into a file, an application program has been developed in Visual Basic. For preventing data missed, polling technique is used; that does not require a hardware interrupt. The data so acquired is split into four digit numbers per channel, and displayed. Since it has 10 bit resolution it can read an analog input value from 0 to 5 Volt, giving a digital output having 1023 distinct steps for a channel. Thus, it has an accuracy of 4.88 mV.

For the temperature monitoring, the range is set between 0 to 50°C. This is converted into voltage from 0 to 5 Volt, by the signal conditioning circuit. The range of the relative humidity is from 10 to 90 % RH. Then the signal conditioning circuit converted this voltage range of 0 - 5 Vclt. The screen shot of the application program is shown in the following Fig. 5.9.

5.3 Hardware Design

The hardware design consists of the proper selection of the components as per the requirement, drawing of the schematics for the PCB layout. It involves the component selection, component description and hardware details of the system designed. Development of the circuit board assembly and its testing has to be carried out.

5.3.1 PCB Schematic

The schematic for the PCB layout for Eight Channel USB based DAS is shown in Fig. 5.4. It has been drawn by using PCB CAD tool called KiCAD.

5.3.2 PCB Fabrication

PCB layout for the schematic of Fig. 5.4 is made using CAD tools, as shown in Fig. 5.5. PCB layout is transferred to the glass epoxy copper clad by screen printing technique.
Fig. 5.4: Schematic of the Eight Channel USB based DAS

Then, the printed PCB is etched in the ferric chloride solution. After etching is over, holes are made by drilling with appropriate drill-bit sizes. Thus, the corresponding circuit assembly is made accordingly as shown in Fig. 5.6.

5.4 Software Design

Software design can be done into the following stages viz. (i) firmware development and (ii) application program development. Both the firmware and application program can be developed in various ways using various programming languages. We can develop it by using 'C' or 'Visual Basic'. Firmware of the microcontroller has been developed in 'C'. The following sections discuss about the development of firmware and application programs.

5.4.1 Firmware Development

Firmware development involves writing program in high level language and converts it to hexadecimal (HEX) or binary (BIN) files, which is suitable for downloading it to
the microcontroller, for proper functioning. It can be carried out as follows:- steps (i) writing algorithm of the problem, then (ii) drawing the flowchart for this algorithm. (iii) if the development of algorithm and flowchart have been completed, then the cor-
responding program can be developed in any language viz basic or C or Pascal.

The following sections give the development of algorithm and flowchart for the firmware program, which is to be developed. Programs have been demonstrated by giving appropriate pseudo codes.

**Algorithm**

*Algorithm of the firmware for PIC18F4550 microcontroller*

1. Start
2. Define I/O settings;
3. Define VID, PID;
4. ADC channel = 0
5. Set Configuration bits
6. Set ADC Channel
7. Read ADC Data
8. Store ADC Data into byte array
9. Increase ADC channel by 1
10. If ADC channel < 8, then goto 5; else
    Send byte array through USB;
11. Clear byte array, then goto 4;
12. Stop
13. End

**Flowchart**

Flowchart for developing the firmware for PIC18F4550 to acquire analog data, to convert it into digital form and send it serially to the USB is given in Fig. 5.7
Fig. 5.7: Flowchart for the firmware of PIC18F4550

START

Define I/O settings
Define VID, PID

ADC Channel = 0

Set ADC Channel

Read ADC Data

Store ADC Data into byte array

Increase ADC Channel by 1

If ADC channel < 8

No

Send byte array through USB

Clear byte array

Yes

Se: Configuration bits
USB Firmware pseudo code in C

unsigned char num, i, j;
unsigned char op[12], read_data[34], Read_buffer[34];
unsigned long Vin;

void interruptO()
{
    HID_InterruptProcO;
    TMR0L = 100;
    INTCON.TMR0IF = 0;
}

void mainO()
{
    int it = 0;
    int adr[];
    ADCON1 = 0x07;
    TRISA = 0xFF;
    TRISE = 0x07;
    INTCON = 0;
    INTCON2 = 0xF5;
    INTCON3 = 0xC0;
    RCON.IPEN = 0;
    PIE1 = 0;
    PIE2 = 0;
    PIR1 = 0;
    PIR2 = 0;
    T0CON = 0x47;
    TMR0L = 100;
    INTCON.TMR0IE = 1;
    T0CON.TMR0ON = 1;
    INTCON = 0xE0;
    Hid_Enable(&Read_buffer, &read_data);
    Delay_ms(1000);
    for(;;)
    {
        for(j = 0; j < 34; j++) read_data[j] = ' ';
        j = 0;
        for(it = 0; it < 8; it++)
{ 
ADCON0 = adr[it];
Vin = Adc_Read(it);
LongToStr(Vin,op);
for(i=0;i<4;i++)
{
    read_data[j]=op[i+7];
    j++;
}
}
while(!Hid_Write(&read_data,34));
Delay_ms(5);
Hid_Disable();
}

5.4.2 Application Program Development

Application program development involves the development of the logic of the pro-
gramming so as to create algorithm of the problem, then drawing the corresponding
flowchart for this algorithm. If the development of algorithm and flowchart have been
completed, then the corresponding program can be developed in any language, like
Visual Basic.

The following sections give the development of algorithm and flowchart for the
application program. Programs have been demonstrated by giving appropriate pseudo
codes.

Algorithm of the Application Program for USB DAS

1. START
2. Load Form
3. Initialize Graphics
4. If VID,PID matches then Connect to USB port
   Else goto 4
5. Read data
6. Display 8 channel data
7. If checkbox of channel checked then Plot the data point in the chart and join with previous data with line.
8. If Start Acquisition button is pressed
   Create a temporary file and save the data
9. If Save As button is pressed
   Save the current temporary file to user specified file
10. If EXIT button pressed then go to ; else goto 5
11. Close the opened file
12. Disconnect the port;
13. Unload form
14. Stop
15. END

Flowchart

Flowchart for developing the application program for the designed DAS, to acquire data, and to display and store it into the hard disk, is given in Fig. 5.8.
Fig. 5.8: Flowchart for the Application Program in VB for Serial Port based DAS
Pseudo code for application program in Visual Basic

USB OnPlugged Event in Visual Basic

Public Sub OnPlugged(ByVal pHandle As Integer)
    If hidGetVendorID(pHandle) = VendorID And 
    hidGetProductID(pHandle) = ProductID Then
        lblstatus.Text = "USB DAQ Plugged in.."
    End If
End Sub

USB OnUnplugged Event in Visual Basic

Public Sub OnUnplugged(ByVal pHandle As Integer)
    If hidGetVendorID(pHandle) = VendorID And 
    hidGetProductID(pHandle) = ProductID Then
        hidSetReadNotify(hidGetHandle(VendorID, ProductID), False)
        lblstatus.Text = "USB DAQ Unplugged..."
    End If
End Sub

USB OnRead Event in Visual Basic

Public Sub OnRead(ByVal pHandle As Integer)
    If hidRead(pHandle, BufferIn(0)) Then
        daqdata1 = Chr(BufferIn(1)) & Chr(BufferIn(2)) & 
                   Chr(BufferIn(3)) & Chr(BufferIn(4))
        daqdata2 = Chr(BufferIn(5)) & Chr(BufferIn(6)) & 
                   Chr(BufferIn(7)) & Chr(BufferIn(8))
        daqdata3 = Chr(BufferIn(9)) & Chr(BufferIn(10)) & 
                   Chr(BufferIn(11)) & Chr(BufferIn(12))
        daqdata4 = Chr(BufferIn(13)) & Chr(BufferIn(14)) & 
                   Chr(BufferIn(15)) & Chr(BufferIn(16))
        daqdata5 = Chr(BufferIn(17)) & Chr(BufferIn(18)) & 
                   Chr(BufferIn(19)) & Chr(BufferIn(20))
        daqdata6 = Chr(BufferIn(21)) & Chr(BufferIn(22)) & 
                   Chr(BufferIn(23)) & Chr(BufferIn(24))
        daqdata7 = Chr(BufferIn(25)) & Chr(BufferIn(26)) & 
                   Chr(BufferIn(27)) & Chr(BufferIn(28))
        daqdata8 = Chr(BufferIn(29)) & Chr(BufferIn(30)) & 
                   Chr(BufferIn(31)) & Chr(BufferIn(32))
    End If
End Sub
chlda = Val(daqdata1)
ch2da = Val(daqdata2)
ch3da = Val(daqdata3)
ch4da = Val(daqdata4)
ch5da = Val(daqdata5)
ch6da = Val(daqdata6)
ch7da = Val(daqdata7)
ch8da = Val(daqdata8)

If aquirestatus = True Then
    If counter = gap Then
        startACQ()
        counter = 0
    ElseIf counter < gap Then
        counter = counter + 1
    ElseIf counter > gap Then
        counter = 0
    End If
End If
Label26.Text = TimeValue(Now)

End Sub

startACQ() program in Visual Basic

Private Sub startACQ()
    Dim textout As String
    textout = Now & vbTab & daqdata1 & vbTab & daqdata2 & vbTab
    & daqdata3 & vbTab & daqdata4 & vbTab & daqdata5 & vbTab
    & daqdata6 & vbTab & daqdata7 & vbTab & daqdata8
    Dim sw As StreamWriter
    sw = File.AppendText("C:\DAQ\DAT\temp.dat")
    sw.WriteLine(textout)
    sw.Flush()
    sw.Close()
End Sub
Application Program in Visual Basic

Fig. 5.9: Screen shot of the Visual Basic Program for USB DAS

5.5 Experimental Observations

For real time temperature, humidity and light intensity monitoring

An IC temperature sensor LM35 [4] and a relative humidity sensor HIH 4000 Series [5] were used. LM35 is pre-calibrated in Degree Celsius. The characteristic graph of the LM35 is made from the experimental observation, as shown in Fig. 5.10, which is linear. Also, the characteristic graph of the humidity sensor HIH 4000 is shown in Fig. 5.11.

Fig. 5.12 shows the screen shot of the Visual Basic program for online monitoring of the temperature, humidity and light intensity respectively using USB based DAS.

Fig. 5.13 shows the reconstructed waveform for 25 minutes of online monitoring of the temperature, humidity and light intensity respectively, using USB based DAS. To visualize the variations in the record clearly, one reconstructed waveform for only
6 second is shown in Fig. 5.14. In the graph black, blue and red colour represents the temperature, light intensity and humidity respectively, in digital value (AU). The actual value can be calculated by multiplying the digital value by a constant multiplier, whose value is depending on the sensor used and it's range.
USB Based DAS

Fig. 5.12: Screen shot of the Visual Basic program for online monitoring, using USB based DAS

Fig. 5.13: Reconstructed waveform for a duration of 25 (twenty five) minutes
According to our channel selections, graphs are plotted for the selected channels. All the data are stored into the hard disk of the PC into comma separated value (.csv) format, if save button is pressed. The data so stored can be used for future use and analysis, which enables reconstruction of the wave form. For temperature, humidity...
and light intensity one representative curve for 25 minutes is shown. Observations are taken for one month by measuring humidity continuously 10 hours a day, one representative curve for a day long observation is presented in Fig. 5.15. The slight fluctuation in the observation may be due to fluctuation in air flow of the room and the variation is less than ±1%, where 5% is considerable in general. Thus, the designed system gives better result.

This system will be useful in research and practical laboratories where acquisition for the measurement, monitoring, analysis and storage of temperature and relative humidity etc. are necessary. The designed system is a low cost with 10 bit resolution having eight channel with accuracy of 4.88 mV and compatible to PC and laptops [6].
REFERENCES


