PC OSCILLOSCOPE

Anubhav Singla and Manish Kumar Singh
Electronics Club, IIT Kanpur

***Abstract* –The problem of obtaining a portable, student friendly *device* for plotting the graphs or testing the circuits for any competition as in the technical fests. This report describes our *device* - which is an easy way to achieve this by the combination of PC, microcontroller and a custom made GUI.**

I. INTRODUCTION

The objective of this project was to design something for practical use, which would also give us exposure to new frontiers. As a result we zeroed on making a PC-Oscilloscope. The basic concept is very straight forward – sample the signal , send the samples, plot it – the required analog signal can be sampled using ADC, these digital samples are sent to computer which plots the samples so as to mimic the given signal. ADC sampling at *appropriate* rate and its subsequent transmission is done using an IC circuit (*Device*), the samples are handled and plotted by a custom made GUI inside *comp*, their details are as follows:-

- First part deals with *Device* Component
- Second part deals with Software Component
- Third part deals with Detailed Implementation

II. *DEVICE* COMPONENTS

Processor

We used Atmega16 – an Atmel AVR series microcontroller. Atmega16 has been used because of availability of compatible C compiler with automatic code generator tool (Code Wizard AVR). The processor clock was set to 8MHz internal oscillator.

Analog to digital converter

In the *device* we used Atmega16's inbuilt ADC. Its 8bit mode provides sufficient accuracy of 0.02V (5/256 V). With clock frequency of 4MHz given to ADC, single conversion takes about 10µs, and 15µs is taken for other processing. That means 25μs per sample which gives maximum of 40,000 samples per second. Our *device* supports at max two channels, as only one ADC *device* is present in the μC, dual channel mode is implemented using the multiplexers inbuilt in the μC.

Universal Asynchronous Receiver Transmitter (UART)

This feature is inbuilt in Atmega16 – digitalized samples are sent to PC using this UART *device* through RS232 serial port. A maximum baud rate of 115200 bits per second can be achieved using this mode of communication. The voltage level of UART of μC is 0V to 5V where for computer's UART it is -12V to +12V; as a result the voltage converter IC MAX232 is used. As laptops and many of the PC's don't have RS232 serial port, commercially available USB - Serial converters can be used. Voltage Level Converter ADC of Atmega16 can tolerate analog voltage of 0 to 5 V only.

Voltage Level Converter

For voltage range exceeding the above limit we need some way to fit this within 0 to 5 V. In our design we achieved this using op-amp circuitry as shown in figure 1. Buffer at input side ensures that no power is drawn from source of the signal. Buffer at output side is provided with 0 to 5V so that it eliminates any rare chances of output going out of safety range of 0to5V .

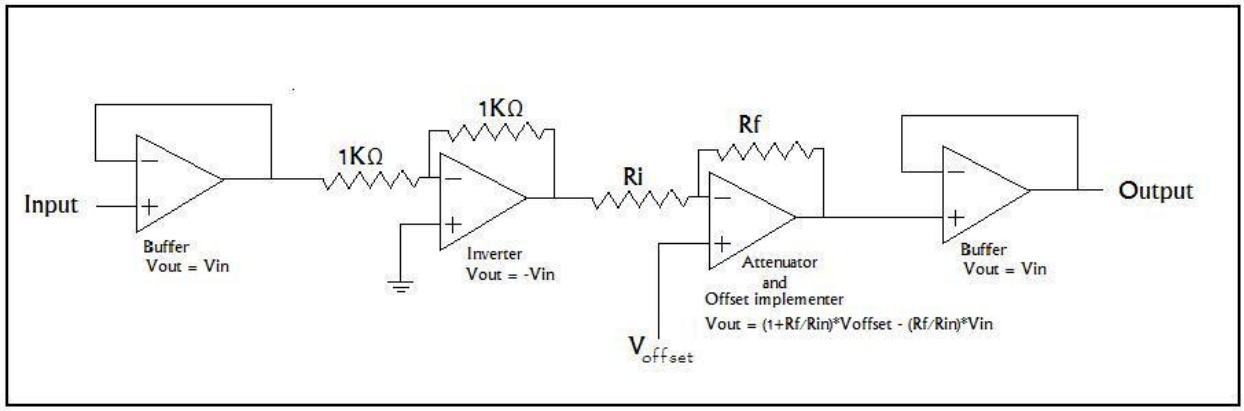

Figure 1: Op-amp circuitry the last Op-Amp has a V_{cc} & V_{EE} are 5 and 0 respectively for the others is -12 & +12.

The formula that results from the analysis of the above circuit is:
Output = $(1+R_f/R_{in})*V_{offset} + (R_f/R_{in})*Input$

III. SOFTWARE IMPLEMENTATION

The Compiler & Language

Custom made GUI is designed using C++ language. We used Bloodshed Dev-C++ compiler which is open source software.

Graphics

For implementing graphics we downloaded some library and header file compatible with Dev-C++ and had some easy to use and familiar functions. The graph is drawn by joining two dots to make a line.

Interfacing with Serial RS232 port

The application transmits and receives the data using the API's present in "windows.h".

IV. DETAILED EMPLEMENTATION

On getting "Start" message from computer, MCU follow the following cycle continuously - it signals ADC to get digital sample of current analog voltage, after completion of conversion it stores the sample to a buffer and polls if any byte from *comp* is received. UART *device* by means of interrupt calls clears that array by sending the sample to *comp*. MCU bypasses first two steps of the cycle if "stop" byte is received.

The application, when executed, (Custom made Graphical User Interphase) automatically detects and reads the serial (or virtual serial) port which is connected to the hardware *device*. After that, some default settings are sent to *µC*. Then it follows the following continuous cycle: send "start" to *µC* -> stores as much number of data as required to plot one screen full graph -> sends "stop" -> plots the graph and listen for any user commands. At larger time per division setting, the rate of transmission and plotting the graph exceeds the rate at which data is sampled so real time plot is possible. In this case the application starts plotting the graph while it stores the data.

User friendly interface

The Screen is occupied by two windows the upper one is the control panel which has some buttons for controlling graph parameter and the lower one emulates the CRO screen i.e. displays the graph. It has the capacity of plotting a two channel graph.

To begin with software provide the user with the means of changing voltage and time scale - by either using the control panel (clicking on appropriate area) or using the command sequence # v - <desired volt per div in mV> - ENTER # The time scale can also be similarly changed, the command sequence # T - <desired time per div in ms> - ENTER # .

The user can pass from single to dual channel mode by using numeric keys '1' & '2' respectively. X-Y and (normal) V-T can be toggled by using the key 'x'.

It has buttons Pause, Resume and Exit. Pause stops the plotting so that you can analyze it better and also save it at a bitmap! (You can always take snapshot by using the key "*prnt scrn*"). Pause-Resume can also be toggled using the space bar.

It also incorporates pointers (say 1 & 2) for convenient measurement, these can be moved around using right-click and left-click of the mouse. Each pointer is a set having two lines parallel to the X and Y axis. The upper right corner of the control-panel constantly displays the difference |X2-X1| as dx and |Y1-Y2| as dy.

Figure 2: The illustration shows the GUI application. Some periodic signal has been plotted. Two different sitting of volt and time per division has been shown for same signal.

Other Details

- Sampling rate of ADC of μC is to be adjusted in accordance with time per division user set in application i.e. higher the time per division lower is the sampling rate. Whenever user changes it, the optimum time gap between two samples - for that setting - is sent to μC. μC sets appropriate delays and clock to ADC.

- To ensure that *comp* and μC are synchronized wrt time *comp* can request for acknowledgement, in return μC send the acknowledgement.

- It might be possible that at higher sampling rate as compared to data transmission rate, due to limited memory of μC, the buffer fills up completely. In that case μC notifies the problem to *comp*.

- Whenever user toggle from single channel to dual channel or vise versa, notification is send to μC. Because in case of dual channel source of analog signal is to be changed alternatively.

V. CONCLUSION

The project can be said to be complete. The signals of order kHz can be plotted effectively. The device also features dual channel plot and X-Y Plot. Although at higher sampling rate the ADC looses its 8bit precision. If a signal of high frequency is plotted at "larger time per div", due to low sampling rate, periodic graphs get mapped to low frequency and the experimenter can be deceived. The Op-amp circuitry needs power source of range -10 to +10 V, as a result we need to use an external power source to supply V_{CC} and V_{EE}. Moreover this circuitry adds $0.4V_{PP}$ AC noise to the given signal which is reflected on the plot. A more efficient algorithm can be used to make sampling rate as high as 100,000 samples per second. Other future work may include feature like digital communication tapping.

Though some features of an actual oscilloscope have been compromised upon but the essence and usefulness have been conserved, making this an ideal and portable *device*. All you need is a computer, the *device* and a copy of our custom-made GUI based software. It can also be made compatible with USB ports by using commercially available RS232-USB controllers.

ACKNOWLEDGEMENT

We are thankful to E-Club coordinators *–Arpit Mathur, Geetak Gupta* and *Paritosh Karnatak* for their support and suggestions and especially to *Ashish Dembla* for intelligent guidance which made our Oscilloscope more realistic.

REFERENCES

[1] The graphics library compatible with Dev-C++ compiler can be downloaded form http://www.uniqueness-template.com/devcpp/#step2

[2] Details of interfacing with Serial RS232 (windows.h) port can be found at http://msdn.microsoft.com/en-us/library/ms810467.aspx

Key Notes

μC: Microcontroller

Comp: *Comp* anywhere refers to Computer/PC/Laptop

ADC: Analog to Digital Converter

UART: Universal Asynchronous Receiver Transmitter

GUI: Graphical User Interface

V_{PP}: Voltage Peak-to-Peak

Device: *Device* anywhere in the document refers to the "*Hardware*" (the IC circuit)