Voltage Plotter Using Programmable System on Chip

Milind Sonawane, Abhay Tambe, Sachin Gengaje

Abstract—During testing, just knowing the amplitude of the signal is not always sufficient. An oscilloscope can be used but is too bulky to be carried around on site. This paper explains implementation of an interfacing card to use computer as a voltage plotter and designing of software that would enable just the same. The interfacing card is developed using PSoC. Input port can be selected in case there are multiple inputs connected to the computer. The application on the computer is developed using Visual Studio

Index Terms—ADC,Measuring voltage of a signal, PC Based Oscilloscope, PSoC, Serial Port, System on Chip,Virtual Instrumentation.

1 Introduction

THEbest way to understand any signal is to haveit's visual representation. The handheld multi-meters are widely used for getting the voltage. The AC mode will provide us peak to peak voltage. Let us assume that there were some noise added in the signal near the zero crossing. Using a multi-meter, it would be impossible to know that. This is where seeing the signal is indispensable. It is not uncommon to find an oscilloscope being used for testing and verification of electronic equipment. For testing telecom devices, it would be very tedious to carry around something as bulky. Especially, when the only requirement is to view the signal and voltage. This is where the idea of a voltage plotter comes from. One can see the signal and get its the voltage. This is similar to the primitive oscilloscopes available back in 1930's, where only the signal could be plotted on a circular CRT screen with 6" diameter. To be portable and satisfy all the criteria within the given budget is a monumental task. If a GLCD were to be used with a microcontroller [1], the approximate cost would come around INR 1000. Affordable, low battery consumption but poor resolution. If a LCD screen is to be used like the ones in cell phones, a good battery would be required too, taking up the total cost to over INR 2500. So instead, a computer is used as a display. To give the input, an interfacing card is developed. As every engineer owns a laptop, the only cost incurred is of the interfacing card. For the card, Programmable System on Chip or simply PSoC [2] developed by Cypress Semiconductors is used. It is programmed using PSoC Creator [3]. Of the number of modules available on chip, ADC [4] and UART [5] are required for this project. For developing the application, there are a number of options available. One may consider using the online tools as explained in [6] [7]. This would however require the user to have network access.

So instead, visual studio is used. The output of the project developed using this IDE is an executable file, which the user simply needs to run. There are other offline tools available for developing the application, such as MATLAB. The downside is that one must install MATLAB to run the application. Such implementations have been done before in [8] [9]. The estimated cost of the project comes down to INR 800. The job of the PSoC is to obtain the data, convert is and transmit it to the PC. The transmission is done using UART. The application on the PC can scale and plot the signal besides fulfilling the primary requirement of detecting the voltage value. Handheld portable oscilloscopes have been implemented before, as explained in [10].Plotters have applications in audio synthesizers and mixers.

2 Implementation

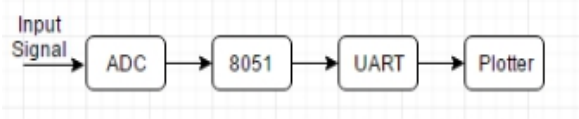

Fig. 1. Block Diagram of the System: PSoC Side

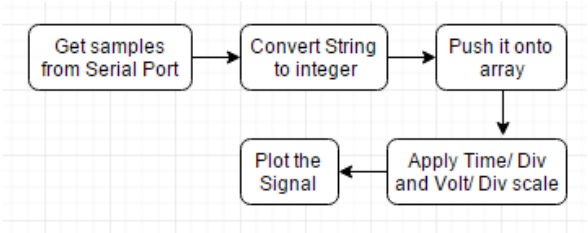

Fig. 2. Block Diagram of the System: Plotter

Any AC signal or DC signal can be applied to at the input. The voltage range for the input is selected between 0 to 2 volts. Outside the range, the value will wraparound. So the AC signal has to be level shifted before being applied. Once the sig-

- *Mr. Milind Sonawaneis currently pursuing M Tech in VLSI in Bharti Vidyapeeth Deemed University, India, PH-+917276829869. E-mail: milinds.747@gmail.com*
- *Dr. Sachin Gengaje is currently Head of Department in Walchand Institute of Technology, Solapur, India, PH-+919028874000. E-mail: gosachin22@gmail.com*
- *Mr. Abhay Tambe is the Managing Director of Reanu Microelectronics, Pune, India, PH-+919423009474. E-mail abhay@reanumicroelectronics.com*

nal is applied, the ADC samples it at the set sampling rate within the schematic. The converted signal is then transmitted to the PC via UART. The transmission speed of UART too can be changed as per requirement. Not more than one sample value is transmitted at a time. The data is transmitted in the form "$72;" This format is selected for compatibility with the serial plotter software available [13], which will later help in verifying the correctness of the designed software. Once done, the program obtains the next value and transmits it. This is done continuously. The working of the plotter can be seen in Figure 2 and Figure 4. The user interface is seen in Figure The sample value obtained is in the form of a string, so it is converted into integer after trimming out the '$' and ';' Then the value is pushed on to the array and plotted after applying the time/ div and volt/ div scale.

2.1Hardware

PSoC, developed by cypress semiconductors is used to transmit the digitized input signal to the computer via UART. It is an extremely versatile and powerful computing platform. The modules used are listed and explained below.

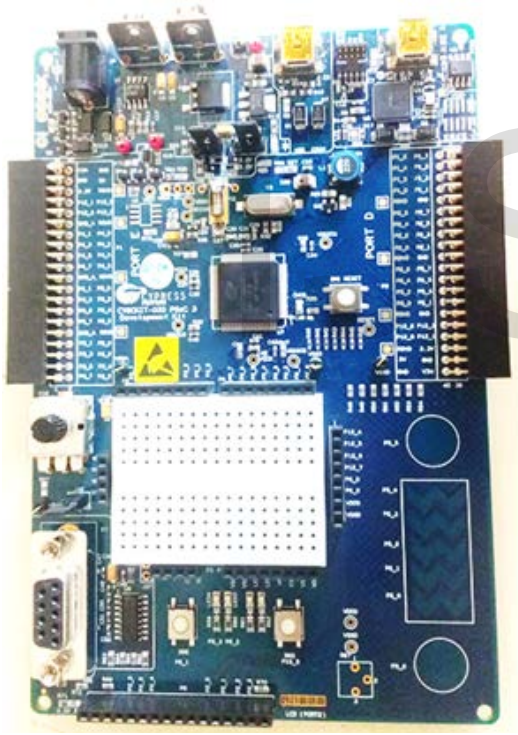

Fig. 3. Block Diagram of the System: PSoC Side

2.1.1 ADC

Delta sigma ADC is available in CY8C386. The user can select desired sampling rate, resolution, conversion mode, interrupts, internal or external clock, input range, reference voltage and number of inputs among others. Within the PSoC family, two main variants of ADC are available: SAR ADC and Delta Sigma ADC. The hardware used has Delta Sigma ADC. For continuous conversion, continuous mode is selected. The baud rate available for transmission is by the lower end, so 8 bit resolution is used. Internal clock is used for higher conversion speed and interrupts for ADC are disabled. The input range is between 0 Volts to 2.05 Volts.

2.1.2 8051

The 8051 used in PSoC is different from the one available in market. The processor executes using a three stage pipeline at clocking speed of 62 MHz. additionally, it has DMA (Direct Memory Access) and NVIC (Nested Interrupt Vector Control). All this can be programmed in the PSoC Creator. Being a programmable system, every single module available on board can be configured as per the application.

2.1.3 UART

As UART is the most simple and basic way of transmitting the data from one device to another. A full duplex communication channel can be established using UART. For this application, half duplex mode is sufficient. The baud rate selected is 38400 bauds per second. The ADC sample is preceded and succeeded by a special character and an end line '\n' is used. This helps to separate the samples being received.

3 Algorithm

3.1 PSoC Side

The PSoC is programmed utilizing different API (Application Programming Interface). This way, the designer has can spend more time finding the most optimum solution as opposed to learning how to program the controller. Initially, the ADC and UART are initialized. These blocks are already configured in the schematic, as explained in theprevious subsections. Next, ADC conversion is started. For this, the SOC (Start of Conversion) goes high. We wait till the EOC (End of Conversion) goes high. Either an interrupt can be used or the pin can be polled. Once the EOC goes high, conversion is completed. The result is converted into ASCII (American Standard Code for Information Interchange) and transmitted through UART encapsulated between one special character at the start and end each to indicate one frame. When we obtain the result, the EOC is automatically reset by the API. This operation is performed as long as there is power being supplied to the controller.

3.2 User Interface

The user interface uses two threads and multiple events to perform the tasks of plotting the wave, calculating and displaying the amplitude and frequency. The data from the UART is automatically obtained as a string every time one frame is detected. The first thread is used obtains the data from the serial port and stores it in an integer array. Then the values in the main array, the one containing all the samples is shifted and the new values are inserted. The voltage and frequency parameters are obtained by this thread. To display the waveform, second thread is created. This thread performs many tasks, besides displaying the waveform. It displays the grid lines, checks what all parameters are to be displayed, the incoming values are to be plotted or the stored waveform is to be drawn on the grid, applying the scale, applying enough delay so that the wave appears to be moving without being

blurred. Both the threads access the same array to store and display the values. Adding delay also avoids contentions between these two threads ensuring that the application runs smoothly.

4 PLOTTER

4.1 Initializing the Application

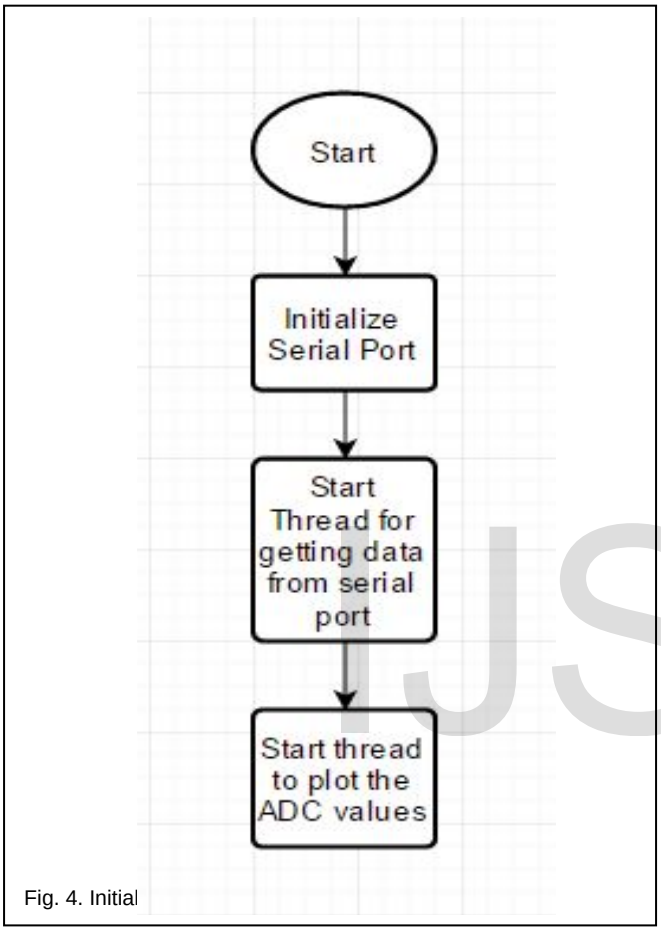

Fig. 4. Initial

The initialization of application is done in 3 steps

1. Initializing serial port
2. Getting data from serial port
3. Displaying the received data

The serial port is set to match the settings on the transmitter side.For getting the data and displaying it multithreading is employed. These threads are synchronized and run independently. The working ofthese Threads andinitialization of serial port is explained in detail in the following sections.

4.2 COM Port

The serial port is treated as a component in Visual Studio. All the parameters of this component have to be initialized before the data can be received. Initially an object is created for the serial port. Then the list of serial ports is obtained and displayed in a combo box. According to the selection made the name of the port is set. The remaining parameters are set as follows

1. Baud rate = 38400
2. Data bits = 8
3. Stop bits = 1 bit
4. Handshake = none
5. Parity = none
6. Read buffer size = 50 Bytes
7. Write buffer size = 50 Bytes

Once all the parameters are said we can say that the serial port has been initialized. It has been ensured that the data received

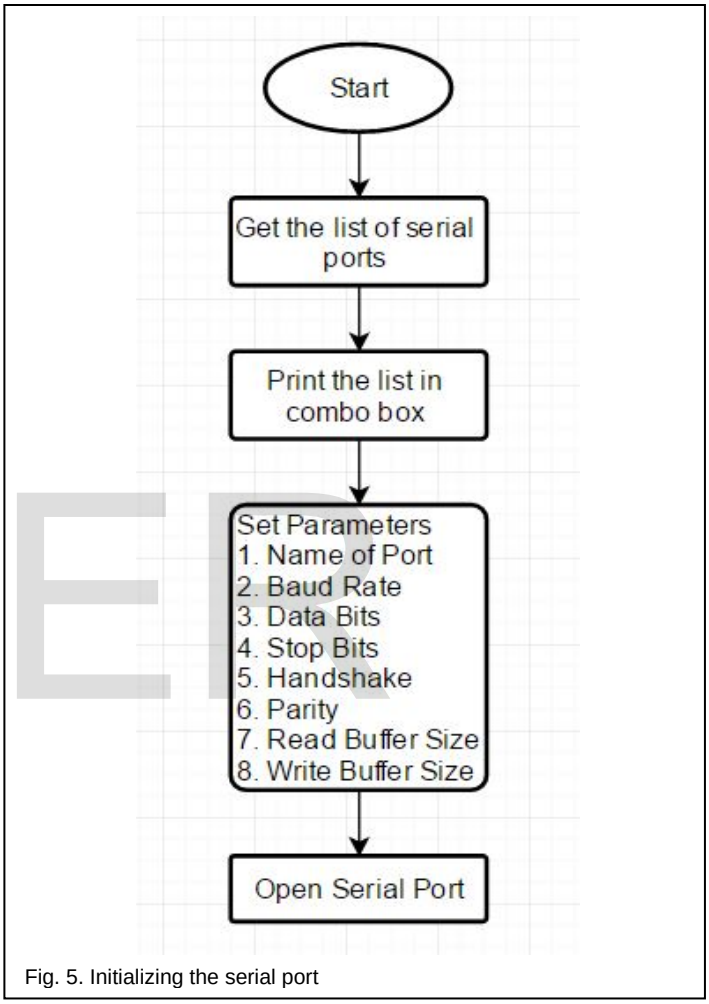

Fig. 5. Initializing the serial port

from the serial port would be in the required format. The serial port is open as long as the application is running. Once the application is closed the serial port is closed.

4.3 Display Thread

This thread is initialized and started when the application starts running. The job of this thread is to print the signal from the stored ADC values. The values are ready to be plotted. This is insured by the thread receiving data from serial port. The sample values I heard according to the scale of volte and time set by the user. The volt and time setting is calculated by using formula 1 and 2 respectively. According to this scale the points are added to the polyline element. Once done, the grid is refreshed. This includes

1. Clearing all the previous elements
2. Printing the grid lines

3. Printing the parameters checked
4. Printing the signal

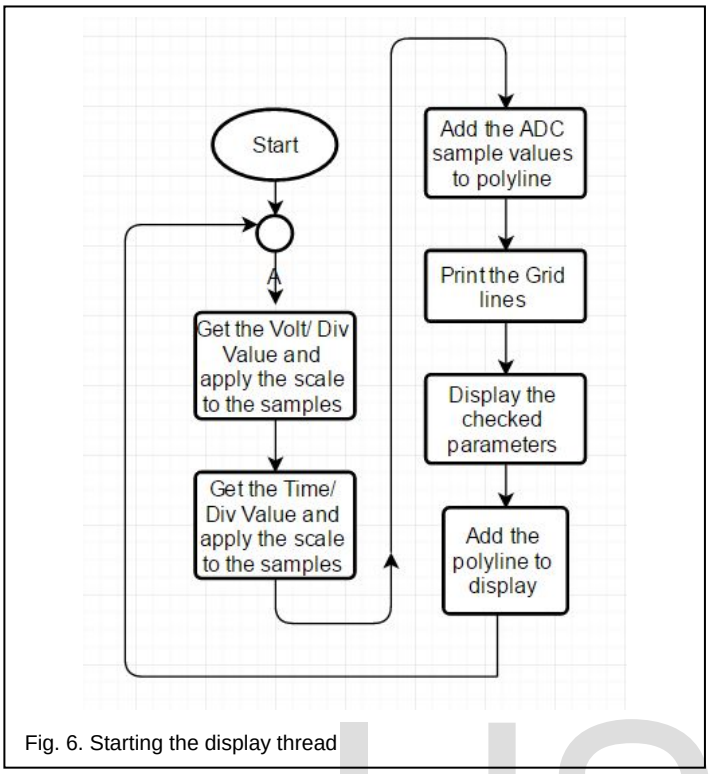

Fig. 6. Starting the display thread

To obtain a stable display, a delay of 400 milliseconds is applied between each frame. This process repeats as long as the application is running. When the application is closed the thread is safely terminated. In case the display is paused, new sample values obtained from the serial port are not displayed.

4.4 Serial Data Receiver Thread

This thread operates smoothly only after the serial port is open. The job of this thread is to receive data from the serial port. Once this thread has started it checks whether the data is in given format, which is "$123;". If it is indeed in the given format we trim the first and last character from the string. Now, note that the data received from the serial port is in the form of a string. The samples can be plotted only if they are in the form of integer. So the string is parsed into integer and stored in an array. The oldest sample is deleted and the new sample is pushed in. Now the thread can go back to monitoring the serial port and receive the next frame.

4.5 Voltage Detection

When the data is obtained from the serial port it is stored in two different arrays. This has to be done because whenever an element is deleted from an array, it is replaced by zero. Due to this we cannot detect the minimum voltage of the signal. So an alternate array is used and the values are stored in it. The values are simply overwritten instead of deleting them, because the sequence of the samples does not matter. The array storing the values to be displayed in the sequential order is different. From this alternative we obtain the maximum and the minimumvalue using the API .Max() and.Min() then we something value is converted into voltage value. This is how we obtain the maximum and minimum voltage. For obtaining the peak to peak voltage the difference of Max and min value of taken and converted into voltage value. These values are that displayed in the UI. Please note that the operation mentioned about is carried out only when the checkboxes for voltages are checked by the user.

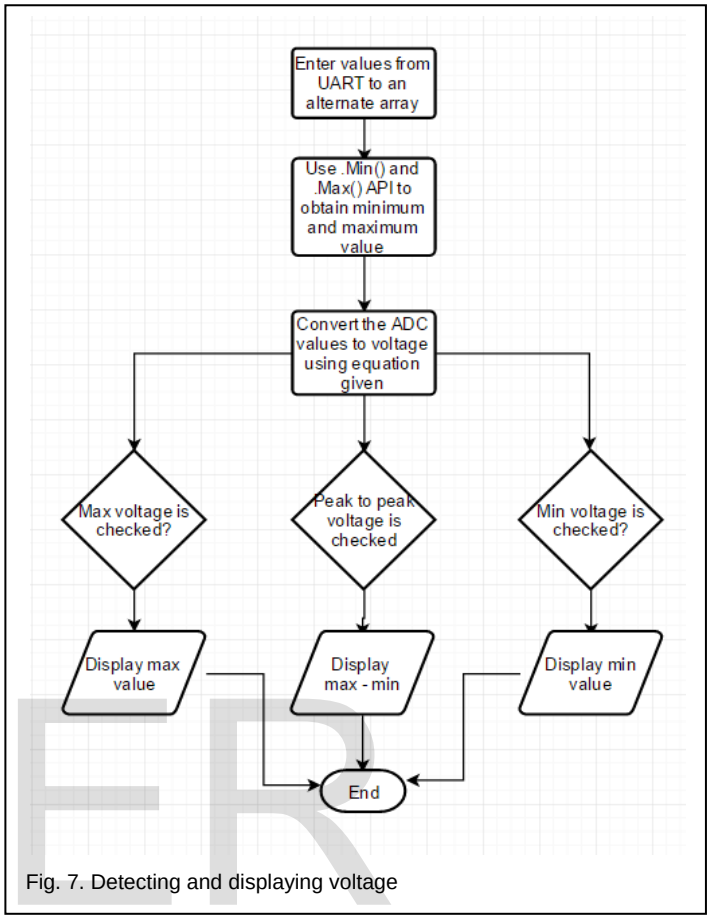

Fig. 7. Detecting and displaying voltage

4.6 Calculating Voltage

Since the values are converted and stored into an array, obtaining the maximum voltage is as simple as getting the largest from the array. Obtaining the minimum value is not as easy. To store new values, the old values have to be discarded. To do this, the array is shifted and the empty places in the array are replaces by zero automatically. So while obtaining the minimum value, the non-zero value has to be obtained.

Peak to peak value is the difference between maximum and minimum value. In case a stored signal is being displayed, the values shall be taken from the respective array. Once the integer value is obtained, the formula used for calculating voltage is given by

$$Voltage = \frac{Value\ obtained}{2^n} \times Input\ Voltage\ Range$$

Where n = Resolution of ADC; 8 bits in this case and the input voltage range is 2.048 Volts.

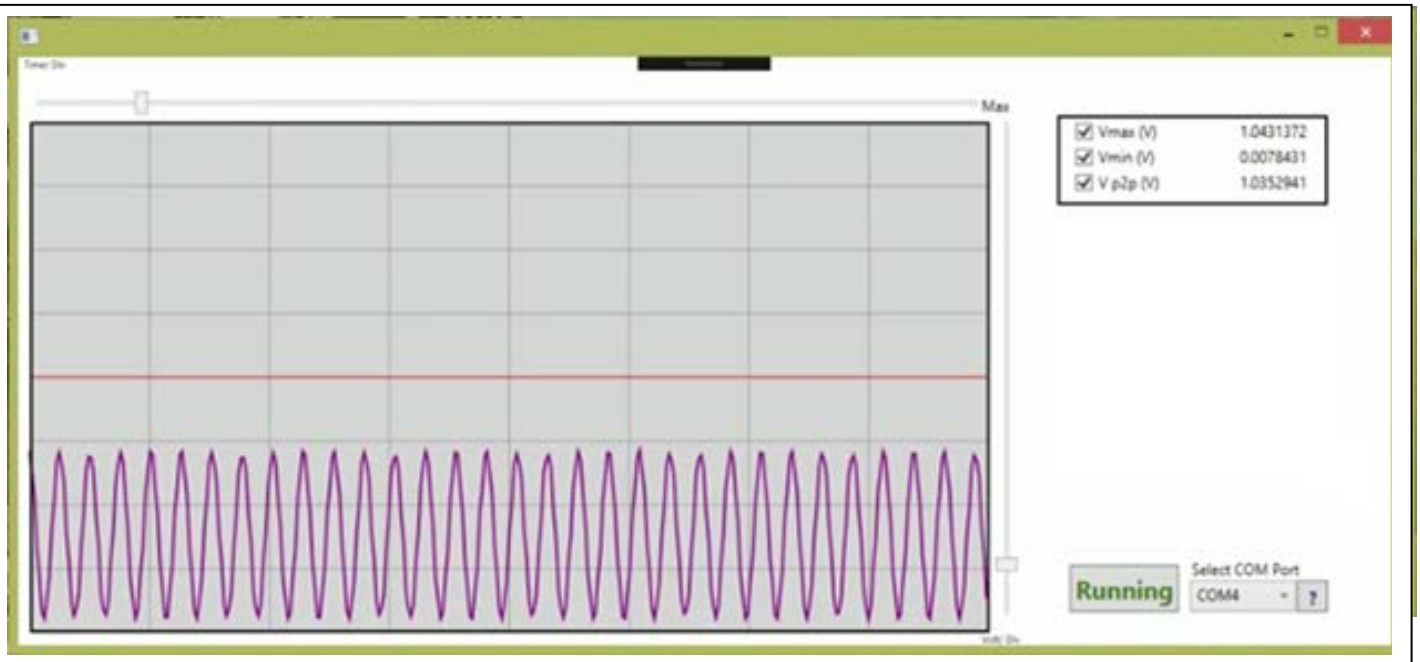

Fig. 8. Detecting and displaying voltage

4.7 Volt/ Division

Essentially, this is as simple as changing the scale of the y axis during run time. For this the ADC sample value obtained is divided by the slider value set. This is more user friendly than manually entering a scaling value. From programming point of view too this becomes simple. Since no value is entered by the user, the programmer need not verify whether the entered value is a number or not.

$$Scaled\ Voltage\ Value = \frac{ADC\ Sample\ Value}{Volt\ Slider\ Value}$$

4.8 Time/ Division

Scaling the time/div is analogous to scaling the x axis in real time. This is implemented by changing the distance between two sample points being plotted. If the scale is decreased, the points being plotted are closer so more sample points are required. Similarly, when the scale is decreased, the distance between sample points increases and fewer sample points are required. Greater the number of sample points, greater the clarity when we increase the scale.

5 Results and Discussion

The sampling rate used is 0.3 Msps and the baud rate is 38400 bps. Up to 100 Hz signal could be faithfully plotted and the parameters received were fairly accurate. The results are recorded in Table 1. Due to some issue with the RS232 IC on the kit, higher transmission rate could not be achieved. User may make his own interfacing card using any microcontroller, use this software and plot the signal. The only condition here is that the transmission rate, ADC input range and frame format are compatible. The software provides this flexibility.

TABLE 1
Applied and Observed Voltage Values

Applied Voltage (Volts)	Observed Voltage (Volts)
0	0.02
0.5	0.57
1	1.02
1.5	1.56
2	1.99
2.1	2.08

6 Conclusion and Future Scope

The readings obtained within the specified voltage range have only 3% error. The waveform seen on the software is no different from the one seen on DSO. Using the ADC samples, the frequency of the signal can be calculated. It can be done by using a combination of methods: detecting zero crossing, counting the number of samples in a cycle, averaging the value over a number of cycles.

References

[1] Sneha Padmagirwar, YuktiJadhao,

TruptiChokhandre,SwapnaDeshpande, Yogesh kale, Pranjali M. Jumle, "Graphical LCD based Digital Oscilloscope using ATmega32" International Journal of Computer and Electronics Research [Volume 2, Issue 6, December 2013]

[2] PSoC Technical Reference Manual **http://www.cypress.com/file/124706/download**

[3] PSoC Creator Manual **http://www.cypress.com/file/137441/download**

[4] ADC component data sheet **http://www.cypress.com/file/179586/download**

[5] UART component data sheet **http://www.cypress.com/file/177171/download**

[6] C. C. Ko, B. M. Chen, S. H. Chen, V. Ramakrishnan, R. Chen, S. Y. Hu and Y. A. Zhuang, "A large-scale web-based virtual oscilloscope laboratory experiment," Inst. Elect. Eng. IEE—Engineering Science Education J., vol. 9, no. 2, pp. 69–76, 2000.

[7] M. Shaheen, A. Loparo, and M. R. Buchner, "Remote laboratory experimentation," in Proc. American Control Conf., 1998, pp. 1326–1329.

[8] AurelGontean, Ioan Lie, MirceaBăbăiŃă"Oscilloscope Control with PC Roland Szabo," International Journal Of Computers And Communications

[9] Ritika, PreetiKumari, PremRanjanDubey"Designing A Pc Oscilloscope Using Freeduino," Birla Institute of Technology, Mesra

[10] Milind Sonawane,"Low Cost Portable Oscilloscopes," International Journal of Research in Advanced Engineering and Technology