DESIGNING A PC OSCILLOSCOPE USING FREEDUINO

Ritika, Preeti Kumari, Prem Ranjan Dubey

Birla Institute of Technology, Mesra, Students of Department of Electronics and Communication

Date: 22nd May, 2013

ABSTRACT

One of the frustrating things about developing and debugging electronic circuits is that we can't look inside the circuit to see what is happening. Even with a circuit laid out before us on a workbench and powered up it may seem like we're in the dark, unable to figure out why an input change or alteration in one part of the circuit isn't having the effect as expected. Sometimes it can feel like we're working with a blindfold on. In this project we use a Freeduino to capture multiple input values and pass them via the USB connection to a host computer running a program that deciphers the values and displays them on-screen. Because the Freeduino itself is not providing any particular intelligence and simply passes on any values it reads, this project is very flexible and the behaviour of the system can be changed simply by altering the software that runs on one's computer. This opens up a wide range of possibilities for using the same basic hardware to process and visualize analog data, parallel digital data, and serial digital data.

INTRODUCTION

A new type of oscilloscope is emerging that consists of a specialized signal acquisition board (which can be an external USB or Parallel port device, or an internal add-on PCI or ISA card). The hardware itself usually consists of an electrical interface providing isolation and automatic gain controls, several high-speed analog-to-digital converters and some buffer memory, or even on-board Digital_Signal_Processor (DSPs). Depending on the exact hardware configuration, the hardware could be best described as a digitizer, a data logger or as a part of a specialized automatic control system. The PC provides the display, control interface, disc storage, networking and often the electrical power for the acquisition hardware. The PCO can transfer data to the computer in two main ways — streaming, and block mode. In streaming mode the data is transferred to the PC in a continuous flow without any loss of data. Block mode utilizes the on-board memory of the PCO to collect a block of data which is then transferred to the PC after the block has been recorded. The PCO hardware then resets and records another block of data. This process happens very quickly, but the time taken will vary according to the size of the block of data and the speed at which it can be transferred. This method enables a much higher sampling speed, but in many cases the hardware will not record data whilst it is transferring the existing block, meaning that some data loss will occur.

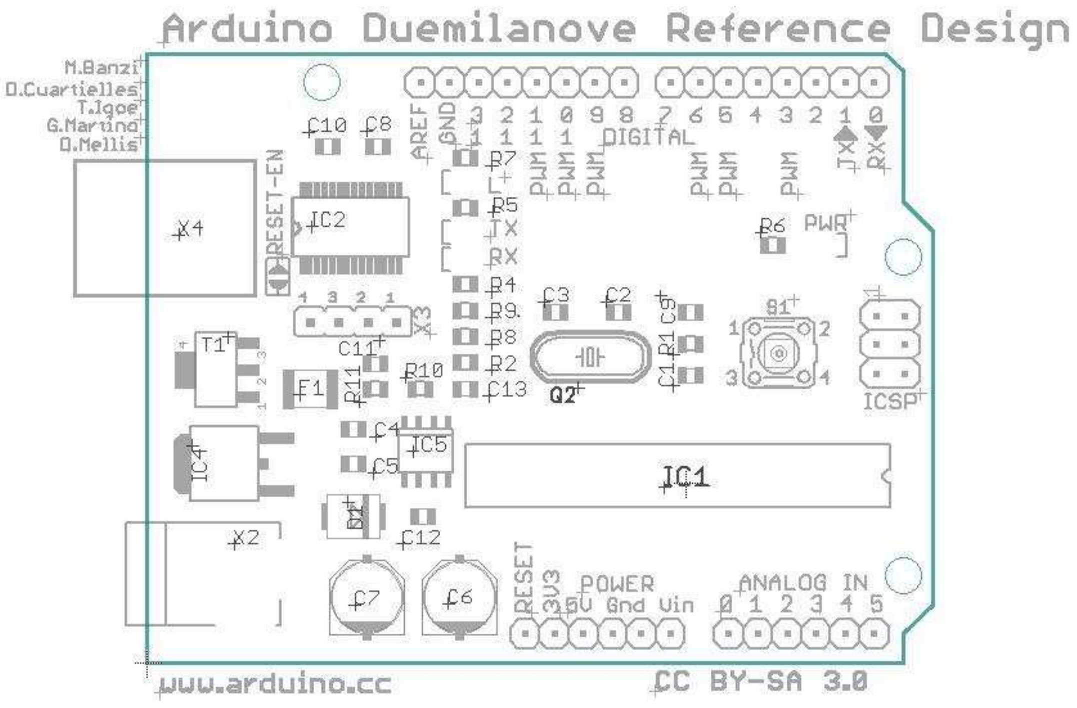
CIRCUITARY FOR PC OSCILLOSCOPE

The PC Oscilloscope has been designed using the Freeduino board. The input to be plotted is given to one of the five analog pins in the board. The software used for interfacing the board with the PC is Matlab R2012a. The compiler Arduino 1.0 has been used for uploading the codes to the board. The MAX 232 Line Driver and ATmega328 Processor has been used. The various components used in this designing has been illustrated briefly in the coming sections.

FREEDUINO BOARD

Freeduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists and anyone interested in creating interactive objects or environments.

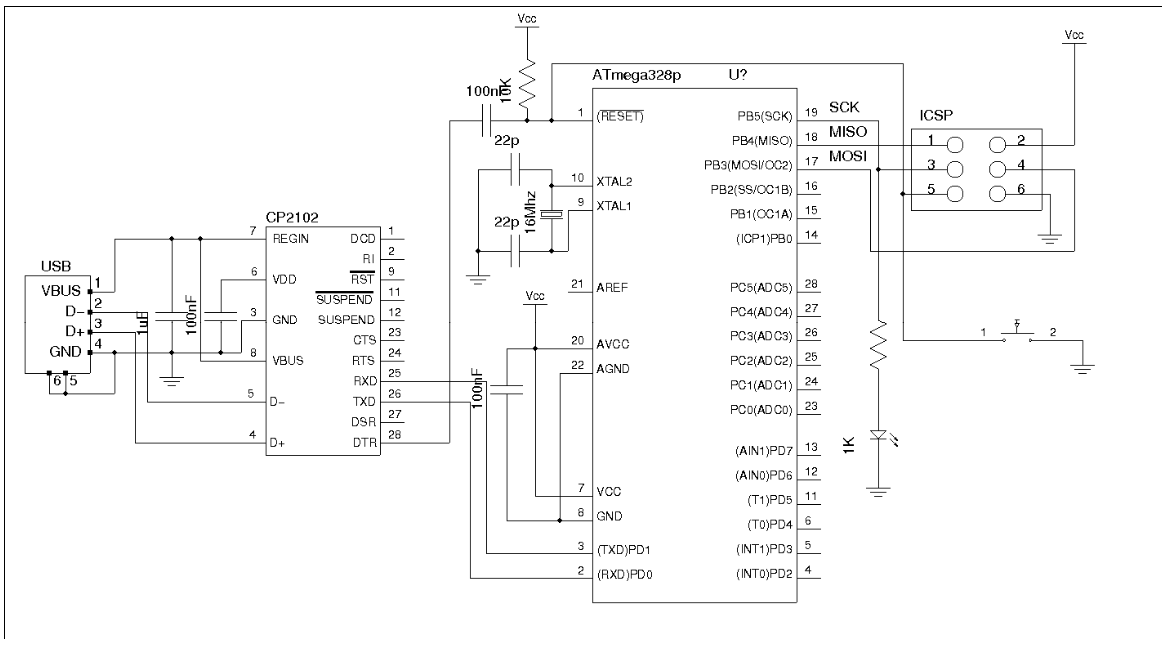
Freeduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Freeduino projects can be stand-alone or they can communicate with software running on a computer (e.g. Flash, Matlab, Processing, MaxMS).



The above schematic diagram is of the Freeduino we have used for our designing process. This Freeduino board is also known as Arduino Duemilanove. The board consists of ATmega328 as the processor and MAX 232 as the Line Driver.

The parts of the Freeduino board used are:

- Printed Circuit Board (with all the tiny bits pre-soldered on)
- Power Jack
- Power Switch
- Shunt (for above header)
- 2x3 6-Pin header (for In-circuit serial programming)
- 2 x 6-socket headers (for shield interface)
- 2 x 8-socket headers (also for shield interface)
- 28-Pin DIP Socket for Atmel Microcontroller
- tm ATmega-328 Atmel Microcontroller with Arduino bootloader
- Pushbutton reset switch



Circuit Diagram of the Arduino Duemilanove

If one wants to make the board connection by themselves then they can visit the site https://code.google.com/p/solar-monitoring-project/wiki/Freeduino.

Features

- No need of extra hardware to get the basic functionality
- Max frequency 7 KHz, enough for hobbyists
- up to 4 channels (at a lower sample rate 7/4 KHz)
- 8bit vertical resolution
- Variable Trigger voltage on Channel 0
- Can sample data for as long as you need

CIRCUIT USED

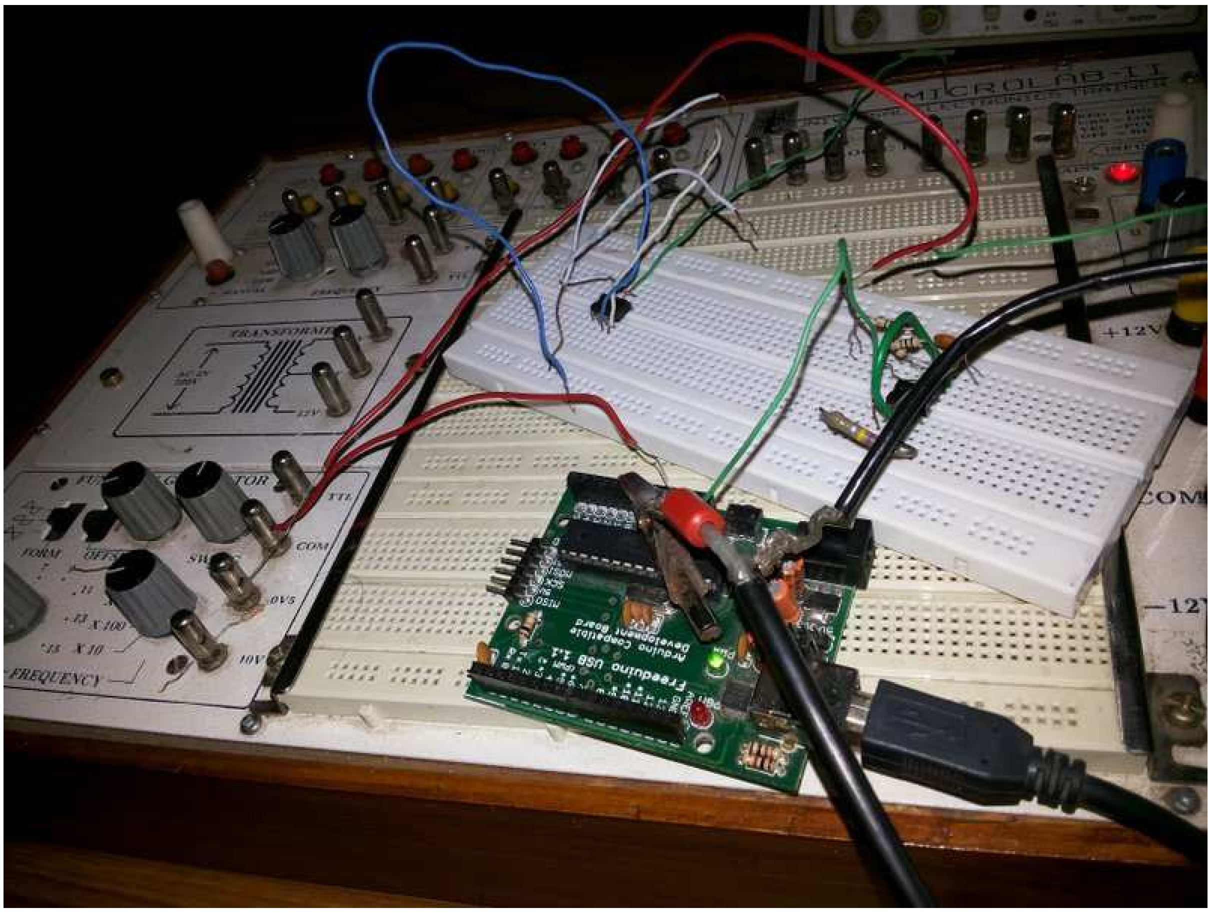
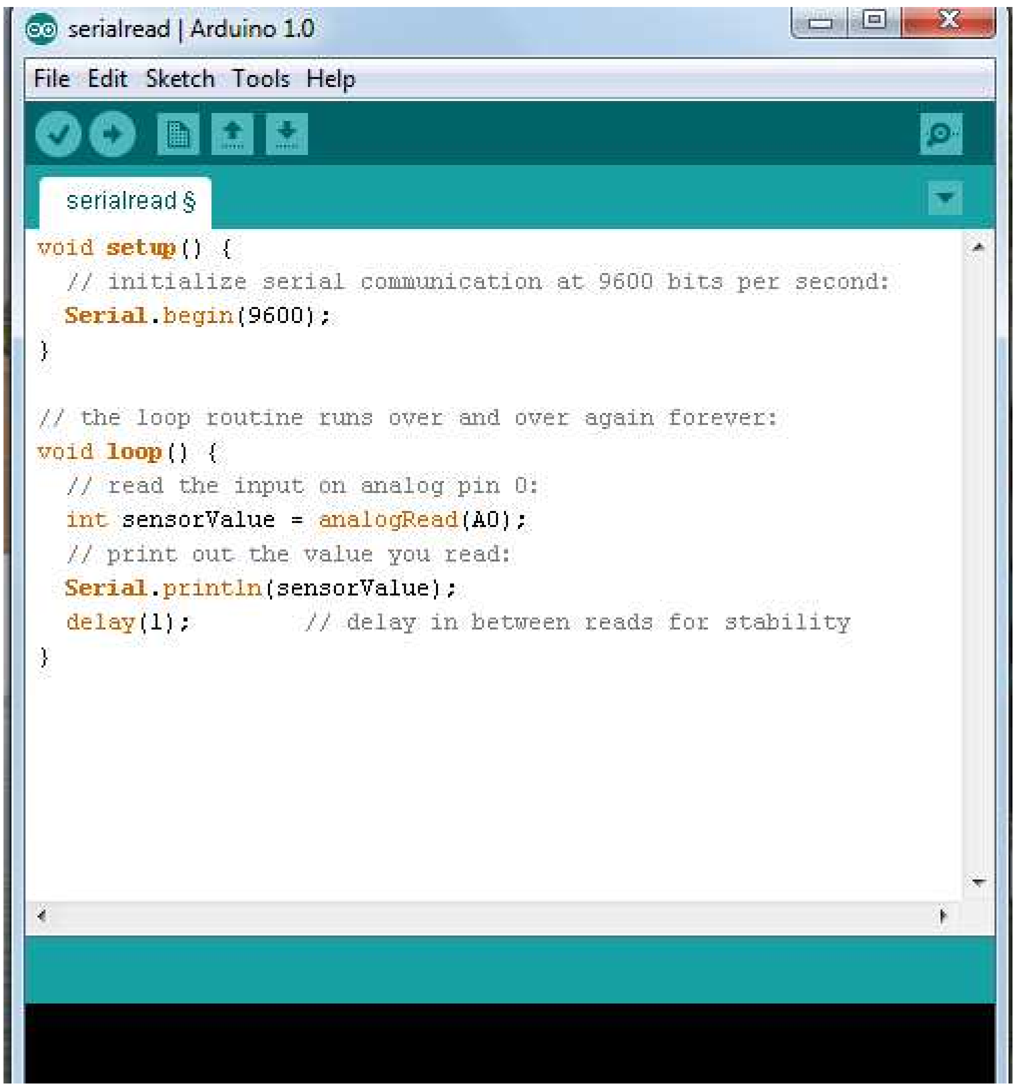


Fig: The circuit connection to the Freeduino board for obtaining the waveforms.

COMPILER USED

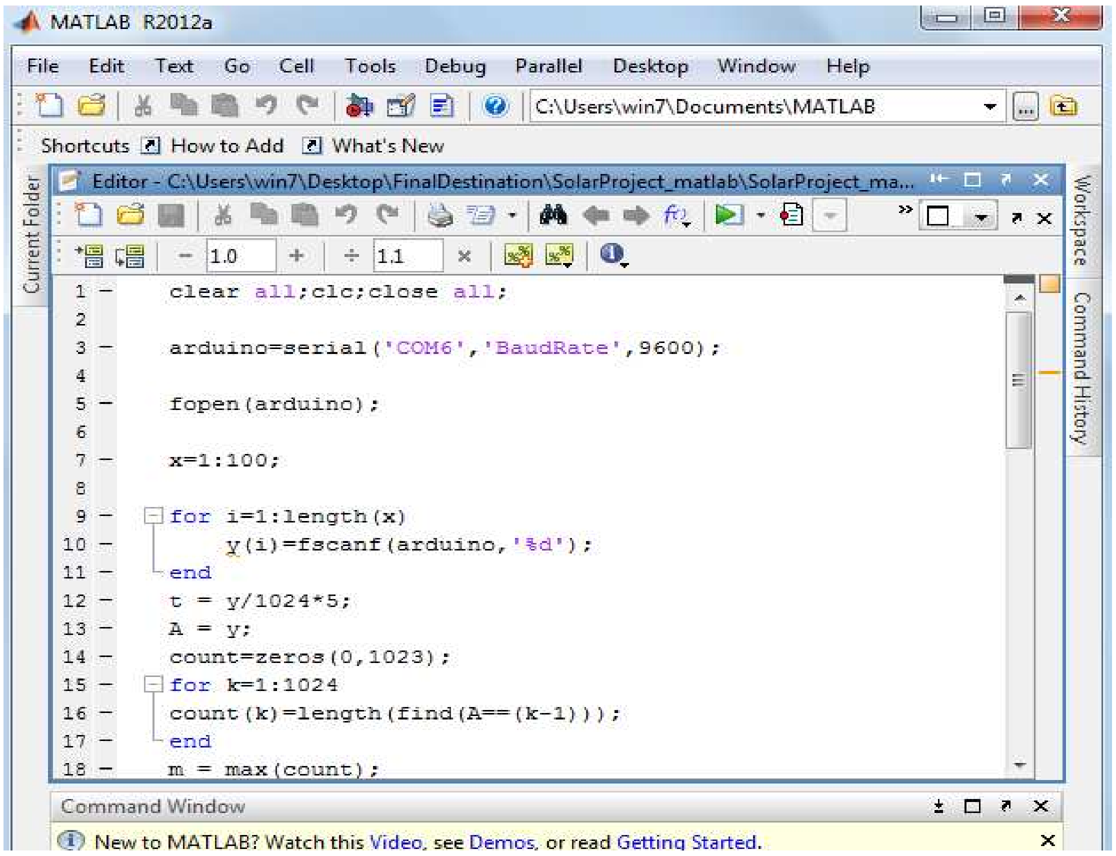
The compiler used for uploading the codes to the Freeduino board is Arduino 1.0. The Arduino1.0 compiler can be downloaded from the site http://arduino.cc/en/Main/Software.

The code used for the serial inputting of data is given below:



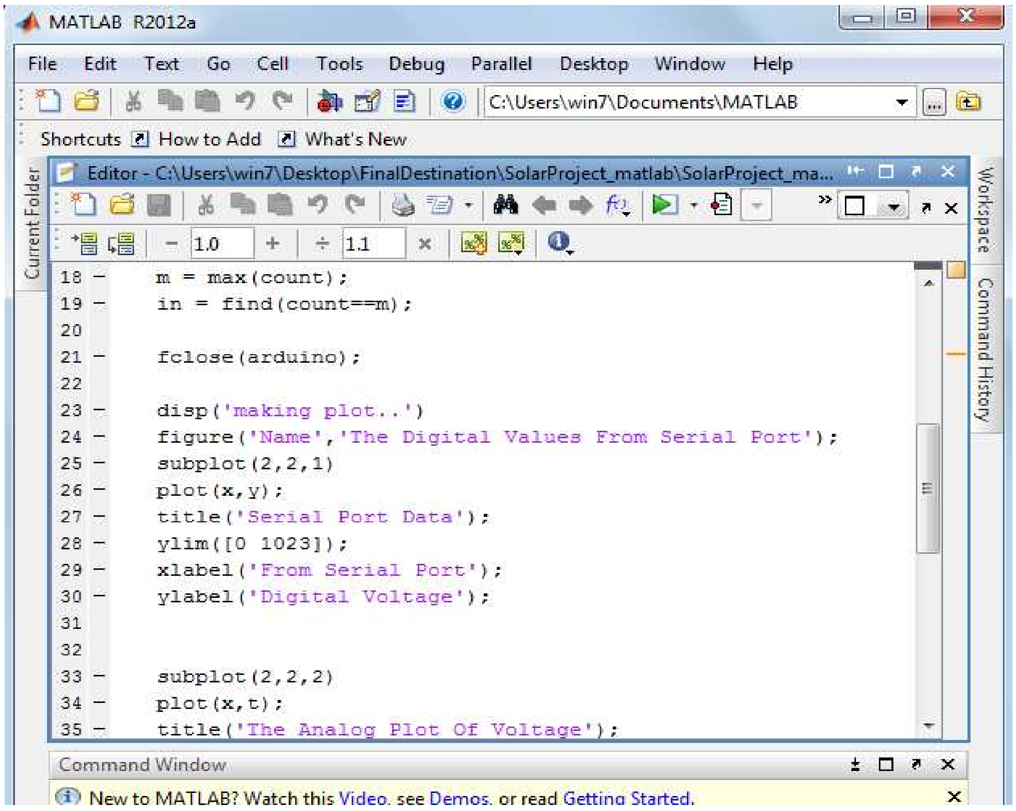
```
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // print out the value you read:
  Serial.println(sensorValue);
  delay(1);        // delay in between reads for stability
}
```

MATLAB CODE USED FOR INTERFACING


MATLAB R2012a
File Edit Text Go Cell Tools Debug Parallel Desktop Window Help
C:\Users\win7\Documents\MATLAB
Shortcuts How to Add What's New
Editor - C:\Users\win7\Desktop\FinalDestination\SolarProject_matlab\SolarProject_ma...
Current Folder
Workspace
Command History
Command Window

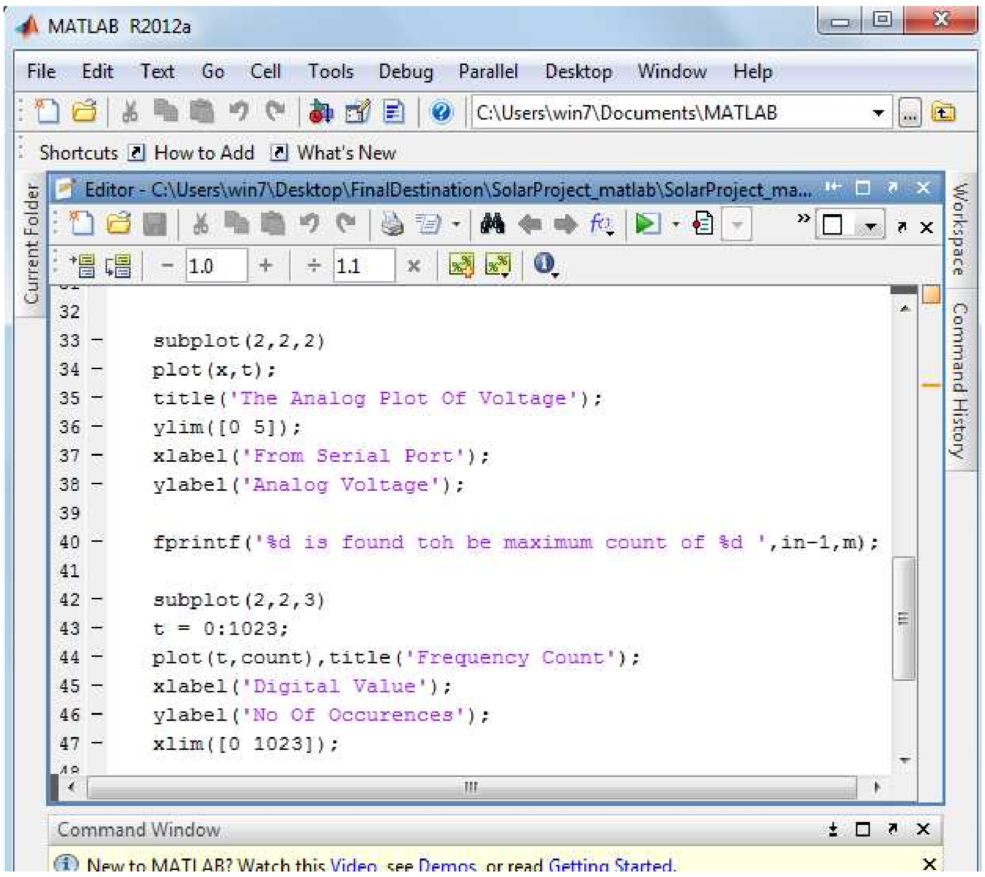
```
clear all;clc;close all;

arduino=serial('COM6','BaudRate',9600);

fopen(arduino);

x=1:100;

for i=1:length(x)
      y(i)=fscanf(arduino,'%d');
end
t = y/1024*5;
A = y;
count=zeros(0,1023);
for k=1:1024
count(k)=length(find(A==(k-1)));
end
m = max(count);
```

New to MATLAB? Watch this Video, see Demos, or read Getting Started.


MATLAB R2012a
File Edit Text Go Cell Tools Debug Parallel Desktop Window Help
C:\Users\win7\Documents\MATLAB
Shortcuts How to Add What's New
Editor - C:\Users\win7\Desktop\FinalDestination\SolarProject_matlab\SolarProject_ma...
Current Folder
Workspace
Command History
Command Window

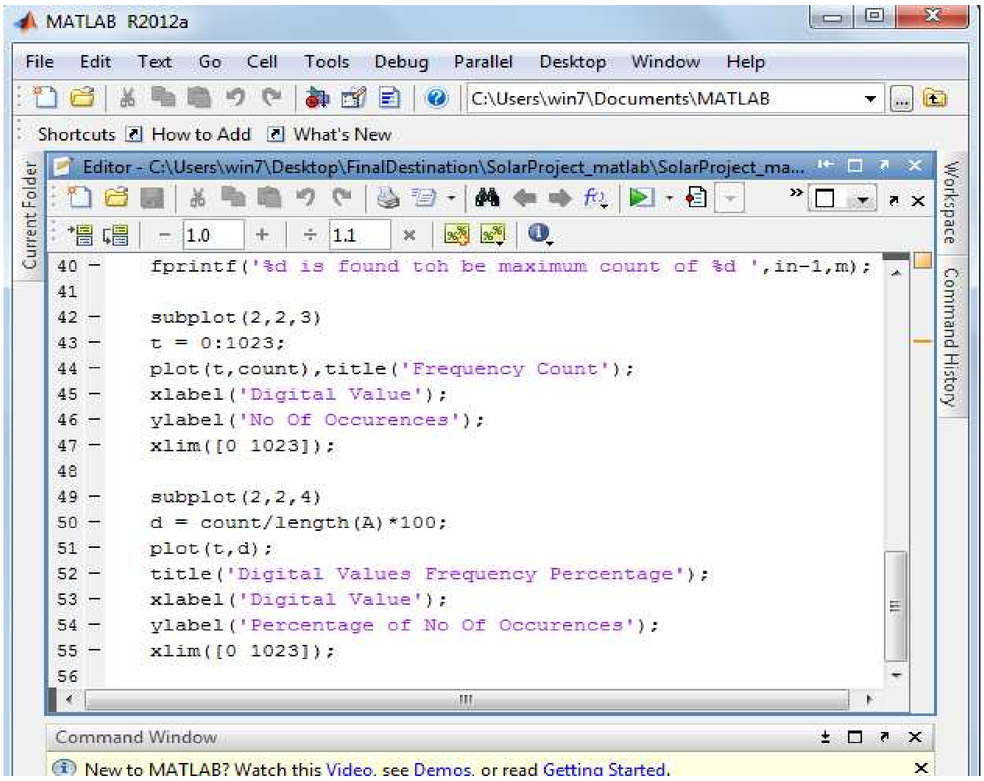
```
m = max(count);
in = find(count==m);

fclose(arduino);

disp('making plot..')
figure('Name','The Digital Values From Serial Port');
subplot(2,2,1)
plot(x,y);
title('Serial Port Data');
ylim([0 1023]);
xlabel('From Serial Port');
ylabel('Digital Voltage');


subplot(2,2,2)
plot(x,t);
title('The Analog Plot Of Voltage');
```

New to MATLAB? Watch this Video, see Demos, or read Getting Started.


MATLAB R2012a
File Edit Text Go Cell Tools Debug Parallel Desktop Window Help
C:\Users\win7\Documents\MATLAB
Shortcuts How to Add What's New
Editor - C:\Users\win7\Desktop\FinalDestination\SolarProject_matlab\SolarProject_ma...
Current Folder
Workspace
Command History
Command Window

```
subplot(2,2,2)
plot(x,t);
title('The Analog Plot Of Voltage');
ylim([0 5]);
xlabel('From Serial Port');
ylabel('Analog Voltage');

fprintf('%d is found toh be maximum count of %d ',in-1,m);

subplot(2,2,3)
t = 0:1023;
plot(t,count),title('Frequency Count');
xlabel('Digital Value');
ylabel('No Of Occurences');
xlim([0 1023]);
```

New to MATLAB? Watch this Video, see Demos, or read Getting Started.


MATLAB R2012a
File Edit Text Go Cell Tools Debug Parallel Desktop Window Help
C:\Users\win7\Documents\MATLAB
Shortcuts How to Add What's New
Editor - C:\Users\win7\Desktop\FinalDestination\SolarProject_matlab\SolarProject_ma...
Current Folder
Workspace
Command History
Command Window

```
fprintf('%d is found toh be maximum count of %d ',in-1,m);

subplot(2,2,3)
t = 0:1023;
plot(t,count),title('Frequency Count');
xlabel('Digital Value');
ylabel('No Of Occurences');
xlim([0 1023]);

subplot(2,2,4)
d = count/length(A)*100;
plot(t,d);
title('Digital Values Frequency Percentage');
xlabel('Digital Value');
ylabel('Percentage of No Of Occurences');
xlim([0 1023]);
```

New to MATLAB? Watch this Video, see Demos, or read Getting Started.

OUTPUT WAVEFORMS

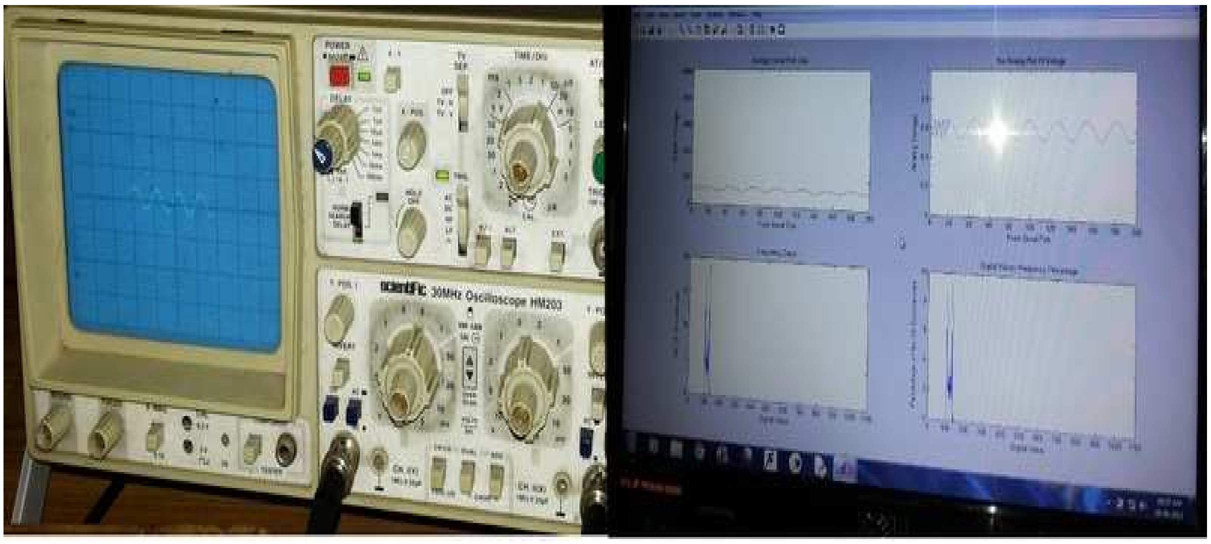


Fig: Comparison between the plots for sine wave obtained by the Oscilloscope and by our PC based Oscilloscope.

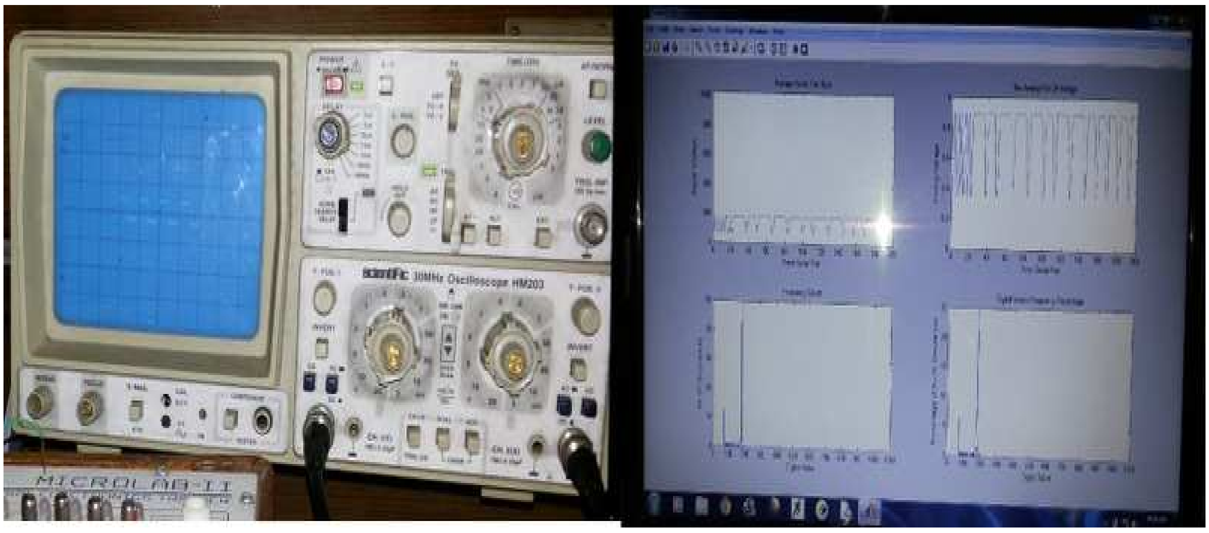


Fig: Comparison between the plots for square wave obtained by the Oscilloscope and by our PC based Oscilloscope.

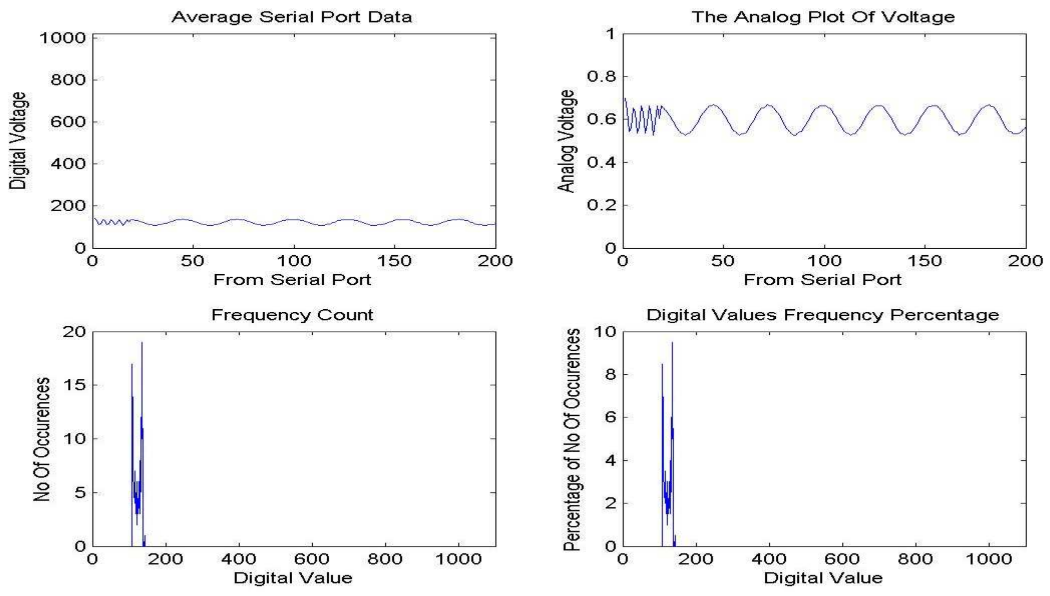


Fig: Plot for the sine wave obtained by the PC Oscilloscope.

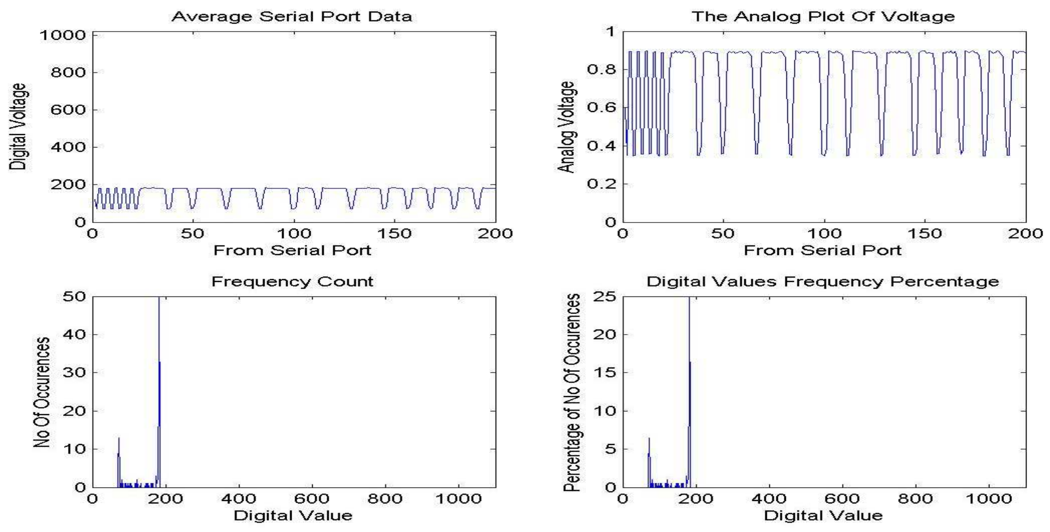


Fig : Plot obtained for the square wave by the PC Oscilloscope.

CONCLUSION

In this paper we developed and presented a PC based Oscilloscope by exploiting serial communication possibilities of MATLAB and Arduino Board. There were several waveforms that were plotted using these connections and were compared with that obtained by the Oscilloscope. The code and the connections used in this paper helps the user to design his/her own oscilloscope. This also reduces the cost which in involved in buying an Oscilloscope to large extent. Also the MATLAB codes can be extended according to the user's wish to have different functionalities like calculation of frequency, rise time, time period etc.

REFERENCES

1. Online: Solar Monitoring project, Wikipedia, building a Freeduino for building a Freeduino Board of one's own and its features and other information.
2. Online: https://code.google.com/p/xoscillo/wiki/arduino for the knowledge and operation frequency of an Arduino Board.
3. Online: http://allaboutee.com/2011/07/04/how-to-send-data-from-the-arduino-to-matlab/ for the codes of MATLAB and for serial transfer of data to the Arduino Board.
4. Online: http://arduino.cc/en/Tutorial/HomePage for learning the programming of Arduino Board.
5. Online: http://www.mathworks.in/matlabcentral/ for learning various serial port functions and commands available in the MATLAB Software.