CPE 301 Microprocessor System Design Lab

LAB #EC02 (Extra Credit Lab) Fall 2013

This extra credit Lab must be completed and demonstrated to Andy before 3pm on November 27 in order to receive the extra credit toward your course grade.

***Objective*:**

To learn how to program the ATmega using interrupts.

***Procedure*:**

9.8 Connect a controllable signal generator source to port pin D4.**Warning: you should calibrate the signal before connecting it to the Arduino**. Do so by connecting the signal generator to an oscilloscope and verify you have a 0-to-5V source. Be sure to note the frequency. Use the program created in problem 9.7 (HW 12). Compare your output with that determined by the oscilloscope. Note: don't forget the pin change interrupt occurs on both rising and falling edges, so you will need to account for that in your output to the user.

***Notes*:**

1. **ALL programs written for this course need to be documented in a complete manner which will make sense to you if you need to go back and review the program a year or two from now. You need to include your NAME and a revision number on ALL programs as well.**

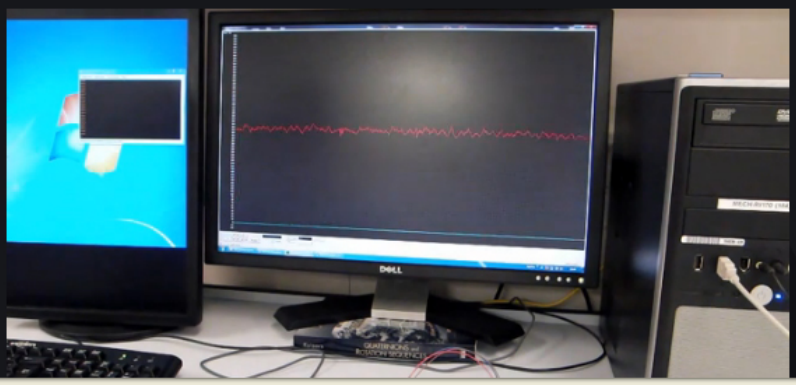
Serial oscilloscope

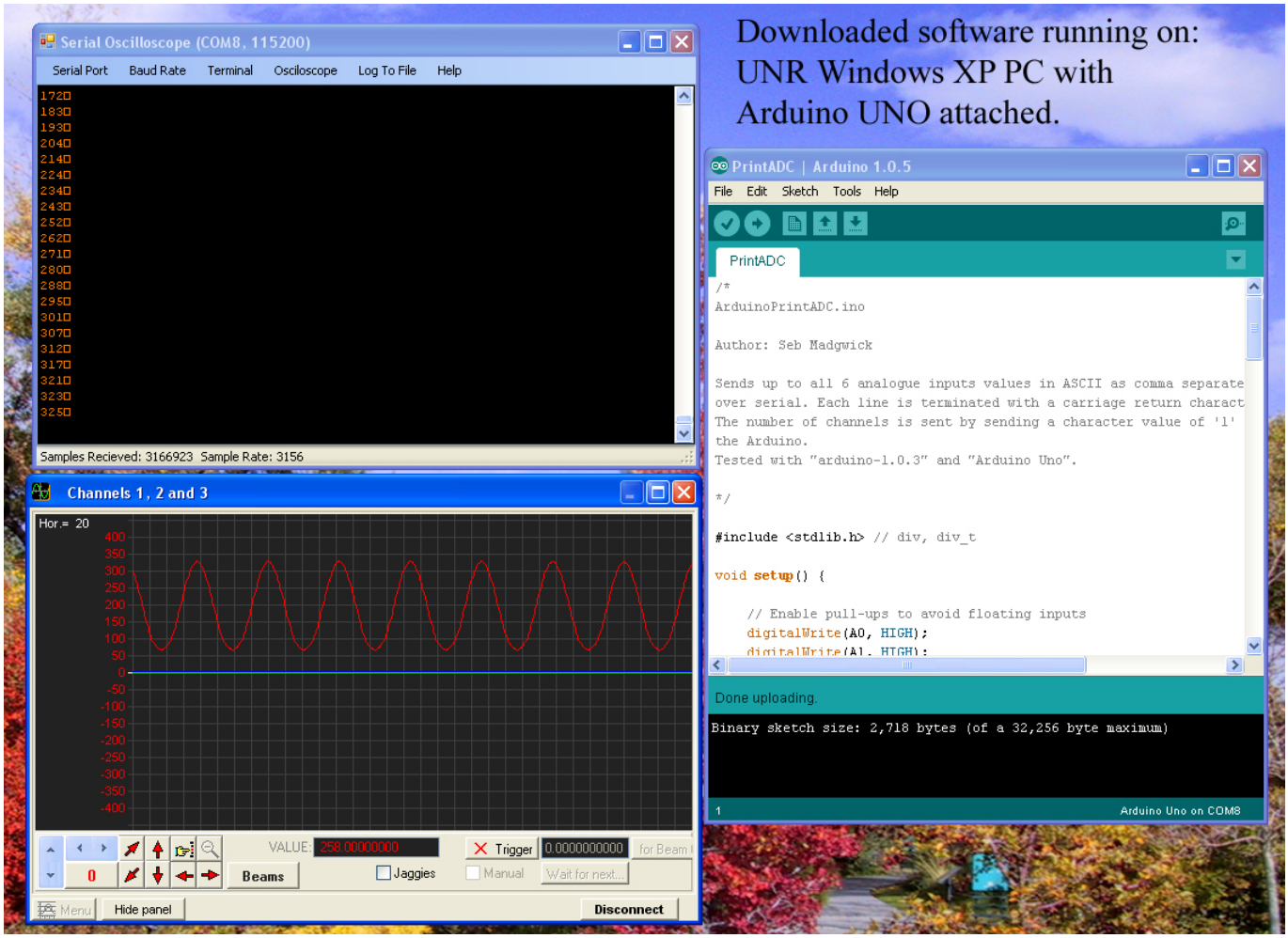
Posted on April 6th, 2013 by Seb Madgwick

Printing data to a serial terminal is a useful debugging tool when developing embedded systems but often the data of interest is a continuous stream of numbers that is of little use displayed as text. For example, this might be data from a sensor or real-time statistics such as bandwidth performance. Serial Oscilloscope is a Windows application that plots comma-separated variables within any incoming serial steam as channels on a real-time oscilloscope. The application also functions as a basic serial terminal, received bytes are printed to the terminal and typed characters are transmitted. The project uses Michael Bernstein's oscilloscope library to plot up to 9 channels on 3 different oscilloscope with view and trigger menus.

Serial Oscilloscope is compatible with any serial stream containing comma-separated values terminated by a new-line character (`"\r"`). For example, `"11,22,33\r"` will be interpreted as values 11, 22 and 33 for channels 1, 2 and 3 respectively. The serial stream can also include non numerical characters which will be ignored. For example, `"a=0.5,blue,x=3.14,t1t2t3,8\r\n"` will be interpreted as values 0.5, 3.14, 123 and 8 for channels 1, 2, 3 and 4 respectively.

The open-source resources also include an Arduino sketch to send analogue input values over serial. In the video I use the Arduino and Serial Oscilloscope to plot data from an IR distance sensor, a triple-axis accelerometer and a microphone.




Downloaded software running on:
UNR Windows XP PC with
Arduino UNO attached.
Serial Oscilloscope (COM8, 115200)
Serial Port
Baud Rate
Terminal
Osciloscope
Log To File
Help
Samples Recieved: 3166923 Sample Rate: 3156
Channels 1, 2 and 3
Hor.= 20
VALUE:
Trigger
Beams
Jaggies
Manual
Wait for next...
Menu
Hide panel
Disconnect
PrintADC | Arduino 1.0.5
File
Edit
Sketch
Tools
Help
PrintADC
/*
ArduinoPrintADC.ino
Author: Seb Madgwick
Sends up to all 6 analogue inputs values in ASCII as comma separate
over serial. Each line is terminated with a carriage return charact
The number of channels is sent by sending a character value of '1'
the Arduino.
Tested with "arduino-1.0.3" and "Arduino Uno".
*/
#include <stdlib.h> // div, div_t
void setup() {
// Enable pull-ups to avoid floating inputs
digitalWrite(A0, HIGH);
Done uploading.
Binary sketch size: 2,718 bytes (of a 32,256 byte maximum)
Arduino Uno on COM8