Patrick Austin

CPE 301 – 1104

Assignment # 9

11/14/2016

Assignment description:

In this lab we studied a provided program whose purpose was to reflect ADC output to a graph on the PC via the UART. We adapted the provided Arduino code to work without using any Arduino library functions, and we set up a circuit that used a photosensitive element to test that the program was working correctly. We could change the amount of light the element received and see this change reflected in terms of the ADC voltage output graph on the PC running the corresponding program.

Problems encountered:

We had difficulty with this lab. Call it “one of those days”. But we stuck with it and were able to get correctly working code and circuit results that matched expectations. Difficulties mostly involved the need to carefully initialize and use the ADC via the different status register bits, as well as basic understanding of what the provided code we were trying to manipulate was accomplishing. Some difficulty was conceptual, as we had to understand that the program was taking ADC readings off of multiple pins, and we were implementing this feature, even though we were effectively only taking one pin’s worth of readings multiple times, redundantly, just to prove the feature was working. We went over the allotted time for the lab but did end up with a result we were happy with.

Lessons learned:

Lessons learned in this lab largely related to a slow crawl through the ADC status registers for the correct bits, reasoning out what needed to change from the previous homework involving the ADC and what could stay the same. In the end I feel I got a better sense of ‘best practices’ involving ADC initialization than I did in the homework, which was good. Also got some experience with the photosensitive element, which I had never used in a circuit before. Also got some review on UART usage, which was also a bit temperamental but came out well in the end.

Description of completed lab:

Here is the specified code:

//Patrick Austin

//CPE 301 Lab 9

//Revision Number 1

//Revision date: 11/14/2016

//Adapted from ArduinoPrintADC.ino by Seb Madgwick, as specified in lab prompt

#include <stdlib.h> // div, div\_t

//hardware pointer declarations

volatile unsigned char \* myPortF = (unsigned char \*)0x31;

volatile unsigned char \* myDDRF = (unsigned char \*)0x30;

volatile unsigned char \* myPinF = (unsigned char \*)0x2F;

volatile unsigned char \*myUCSR0A = (unsigned char \*)0x00C0;

volatile unsigned char \*myUCSR0B = (unsigned char \*)0x00C1;

volatile unsigned char \*myUCSR0C = (unsigned char \*)0x00C2;

volatile unsigned int \*myUBRR0 = (unsigned int \*) 0x00C4;

volatile unsigned char \*myUDR0 = (unsigned char \*)0x00C6;

volatile unsigned char \*myADCSRA = (unsigned char \*)0x7A;

volatile unsigned char \*myADCSRB = (unsigned char \*)0x7B;

volatile unsigned char \*myADMUX = (unsigned char \*)0x7C;

volatile unsigned char \*myDIDR0 = (unsigned char \*)0x7E;

volatile unsigned int \*myADCDR = (unsigned int \*) 0x78;

void setup()

{

\*myDDRF = 0; // Enable port F as input with pull-ups

\*myPortF = 0xFF;

unsigned long FCPU = 16000000; //initialize UART

\*myUCSR0A = 0x22;

\*myUCSR0B = 0x18;

\*myUCSR0C = 0x06;

\*myUBRR0 = 16; //baud rate preload value for 115200

\*myADCSRB = 0x40; //initialize ADC

\*myADCSRA |= 0x80; //0b1000 0000

\*myADMUX |= 0xC0;

\*myDIDR0 |= 0xFF; //or 0x20??

}

void loop()

{

static int numChans = 2; //assume 2 inputs

// Print ADC results for active channels

PrintInt(pollADC(0));

if(numChans > 1) {

U0putchar(',');

PrintInt(pollADC(1));

}

U0putchar('\r'); // print new line

}

// Fast int to ASCII conversion - code by Seb Madgwick

void PrintInt(int i) {

static const char asciiDigits[10] = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9' };

div\_t n;

int print = 0;

if(i < 0) {

U0putchar('-');

i = -i;

}

if(i >= 10000) {

n = div(i, 10000);

U0putchar(asciiDigits[n.quot]);

i = n.rem;

print = 1;

}

if(i >= 1000 || print) {

n = div(i, 1000);

U0putchar(asciiDigits[n.quot]);

i = n.rem;

print = 1;

}

if(i >= 100 || print) {

n = div(i, 100);

U0putchar(asciiDigits[n.quot]);

i = n.rem;

print = 1;

}

if(i >= 10 || print) {

n = div(i, 10);

U0putchar(asciiDigits[n.quot]);

i = n.rem;

}

U0putchar(asciiDigits[i]);

}

unsigned char U0kbhit() //kbhit, getchar, putchar as used in previous labs & hw

{

return (\*myUCSR0A & 0x80);

}

unsigned char U0getchar()

{

return (\*myUDR0);

}

void U0putchar(unsigned char U0pdata)

{

while ( (\*myUCSR0A & 0x20) == 0 )

{}

\*myUDR0 = U0pdata;

}

unsigned int pollADC(int pin) //return 10 bit ADC conversion value from ADC data register

{

unsigned int result;

if (pin == 0) //read from specified pin on port F- modify ADMUX value appropriately

{

\*myADMUX &= 0xE0; //0b 1111 1000

}

else

{

\*myADMUX &= 0xE0; //0b 1111 1000

\*myADMUX |= 0x01; //0b 0000 0001

}

\*myADCSRA |= 0x40; //start conversion

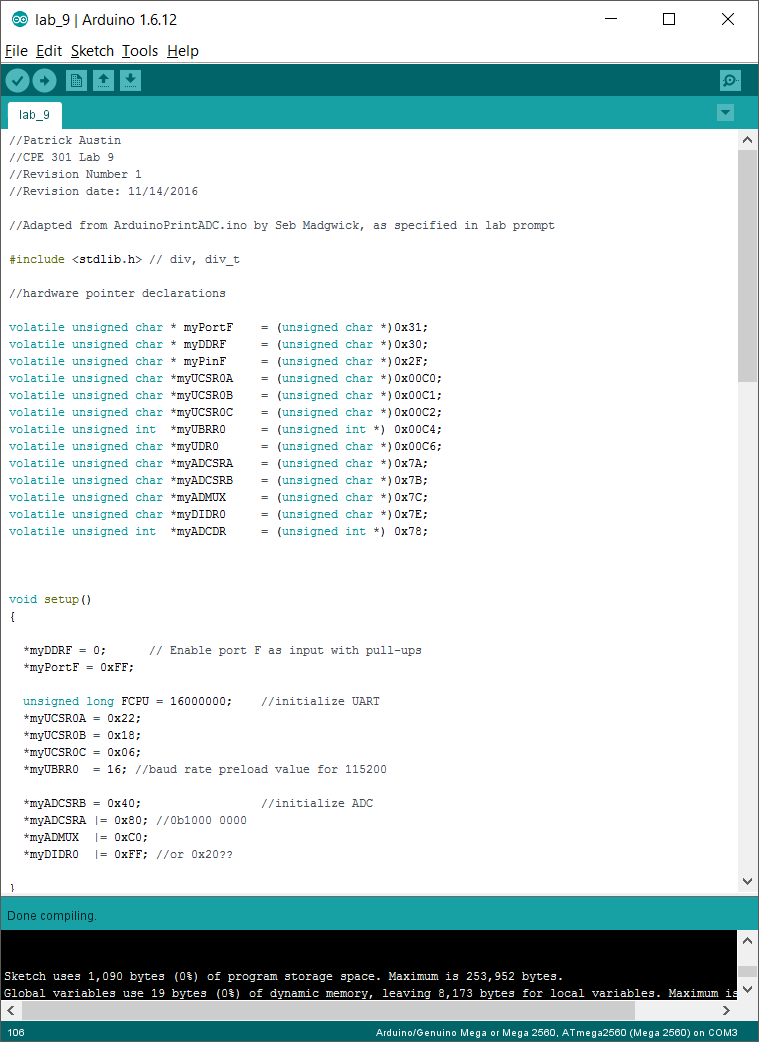
while((\*myADCSRA & 0x40) != 0); //wait for conversion to complete

result = \*myADCDR; //take result, return it

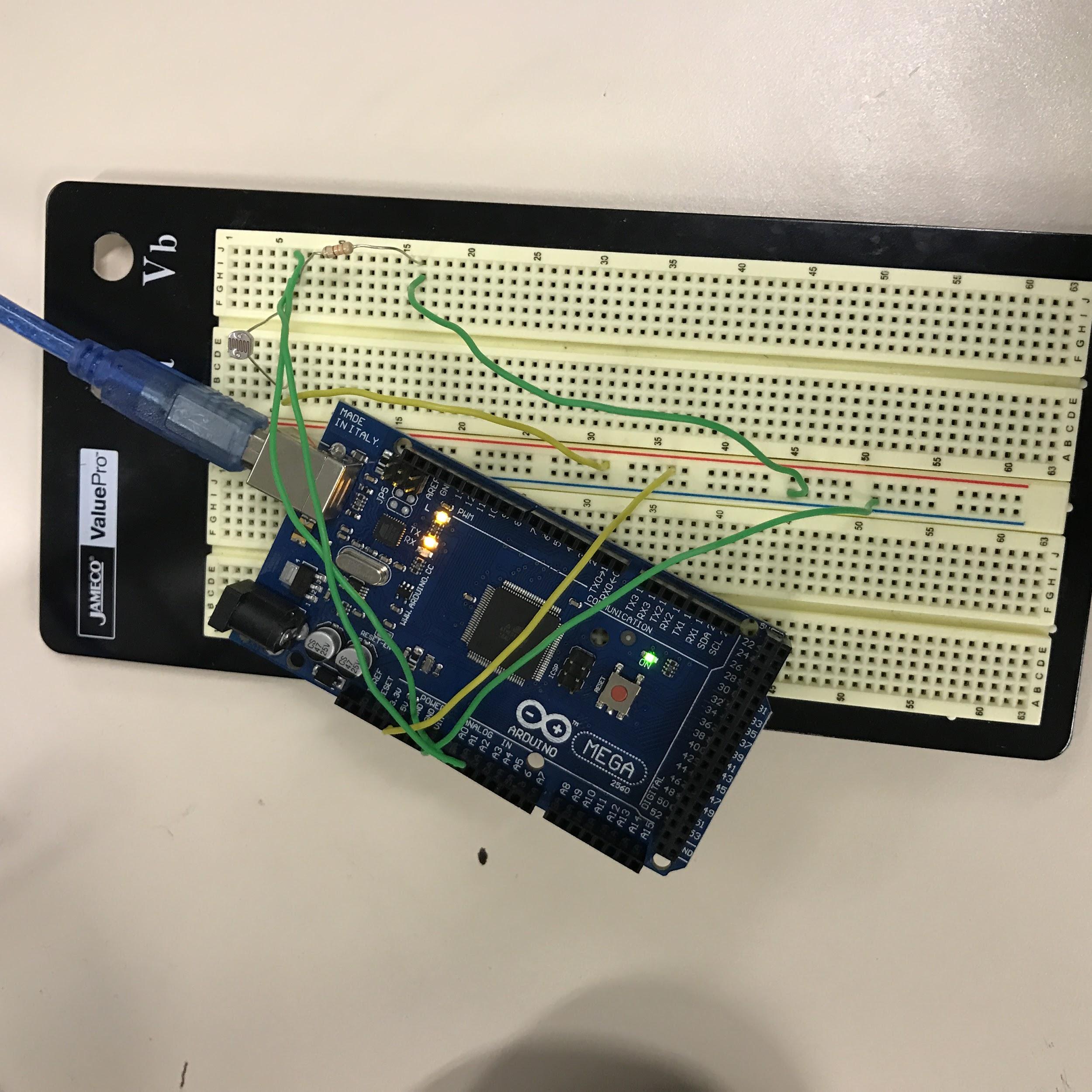
return result;

}

Here is a screenshot of the program compiling successfully:



Here is a photo of the completed circuit:



Here is a photo of the output to the PC:

