UNIVERSITY OF NEVADA LAS VEGAS – DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING



Design Assignment 6

CPE 301 Fall 2016

Luis Ruiz

**PART 0**

The assignment took me about 1 hours to do.

1. **PART A: Description**

My design consists of a C program that reads in a value from a potentiometer and outputs on to the screen using UART. The program takes advantage of Analog to Digital converter to read a voltage off a potentiometer that is creating a voltage divider with a resistor. The value being read is consistently fluctuating from 0 – 5 V max, depending on the restive state of the potentiometer. The value of the resistor is 330 ohms as for the potentiometer, it has a max value of 2000 ohms. The output will be via PUTTY using UART using asynchronous communication. Note a delay of 1 second is placed for more readability, since it is a string is consistently being outputted.

**PART B: Code**

**Header File ioe.h**

#include <avr/io.h>

#ifndef IOE\_H

#define IOE\_H

#ifndef FREQ

#define FREQ 16000000UL

#endif

/\*SETTINGS\*/

#define BAUDRATE 9600 //BAUDRATE (9600 is default)

#define BUFF 256

#define ASYNCH\_NORM (FREQ/16/BAUDRATE - 1)

#define ASYNCH\_DUB (FREQ/8/BAUDRATE - 1)

#define SYNC\_MASTER (FREQ/2/BAUDRATE - 1)

/\*MACROS USEFUL TO DISABLE AND ENABLE\*/

#define TX\_START() UCSR0B |= \_BV(TXEN0) // Enable TX

#define TX\_STOP() UCSR0B &= ~\_BV(TXEN0) // Disable TX

#define RX\_START() UCSR0B |= \_BV(RXEN0) // Enable RX

#define RX\_STOP() UCSR0B &= ~\_BV(RXEN0) // Disable RX

#define COMM\_START() TX\_START(); RX\_START() // Enable communicationsF

#define COMM\_STOP() TX\_STOP();RX\_STOP() // Disable Communication

//Frame Size to be Transmitted

#define CHAR6() UCSR0C |= \_BV(UCSZ00)

#define CHAR7() UCSR0C |= \_BV(UCSZ01)

#define CHAR8() UCSR0C |= \_BV(UCSZ01)|\_BV(UCSZ00)

#define CHAR9() UCSR0B |= \_BV(UCSZ02);UCSR0C |= \_BV(UCSZ01)|\_BV(UCSZ00)

/\* Interrupt macros; Remember to set the GIE bit in SREG before using (see datasheet) \*/

#define RX\_INTEN() UCSR0B |= \_BV(RXCIE0) // Enable interrupt on RX complete

#define RX\_INTDIS() UCSR0B &= ~\_BV(RXCIE0) // Disable RX interrupt

#define TX\_INTEN() UCSR0B |= \_BV(TXCIE0) // Enable interrupt on TX complete

#define TX\_INTDIS() UCSR0B &= ~\_BV(TXCIE0) // Disable TX interrupt

/\*Stop Bit\*/

#define STOPBIT\_1() UCSR0C &= ~(1<<USBS0)

#define STOPBIT\_2() UCSR0C |= (1<<USBS0)

/\*Parity Mode\*/

#define DisParity() UCSR0C &= ~(1<<UPM01);UCSR0C &= ~(1<<UPM00)

#define EvenParity() UCSR0C |= (1<<UPM01)

#define OddParity() UCSR0C |= (1<<UPM01)|(1<<UPM00)

/\*MODE\*/

#define ASYNCH\_MODE() UCSR0C &= ~(1<<UMSEL01);UCSR0C &= ~(1<<UMSEL00)

#define SYNCH\_MODE() UCSR0C |= (1<<UMSEL00)

#define MASTER\_MODE() UCSR0C |= (1<<UMSEL01)|(1<<UMSEL00)

/\*Enable Interrupts\*/

#define recInterrupt UCSR0B |= (1 << RXCIE0)

/\*FUNCTION DECLERATION\*/

/\*

\* Procedure to initialize USART0 asynchronous with enabled RX/TX, 8 bit data,

\* no parity, and 1 stop bit.

\*/

void usart0\_init\_ ();

// Return a char from the serial buffer

/\* Use this function if the RX interrupt is not enabled.

\* Returns 0 on empty buffer

\*/

unsigned char getChar\_(void);

//Transmits a byte

/\*

\* Use this function if the TX interrupt is not enabled.

\* Blocks the serial port while TX completes

\*/

void putChar\_(unsigned char data);

/\*A string print called printm that uses a

char array and your putchar clone to transmit

strings\*/

void printm(unsigned char \*str);

/\*uses an uninitialized char array and your getchar clone to

construct a string for your ATmega328P \*/

//const unsigned char\* scanm(void);

#endif /\*IOE\_H\*/

**Main.c**

/\*

\* Author : Luis

\*/

#include <avr/io.h>

#include "ioe.h"

#include <util/delay.h>

#include <stdio.h>

#define *F\_CPU* 16000000UL

/\*

\* Procedure to initialize USART0 asynchronous with enabled RX/TX, 8 bit data,

\* no parity, and 1 stop bit.

\*/

void usart0\_init\_ ()

{

// To set baud rate

UBRR0H = ((ASYNCH\_NORM) >> 8); //top nibble

UBRR0L = (*uint8\_t*) ((ASYNCH\_NORM)) ; //lower byte

COMM\_START(); // enable transmit/receive

// asynchronous, 8N1, disable parity, 1 stop bit

ASYNCH\_MODE();

DisParity();

STOPBIT\_1();

CHAR8();

}

// Return a char from the serial buffer

/\* Use this function if the RX interrupt is not enabled.

\* Returns 0 on empty buffer

\*/

unsigned char getChar\_(void)

{

//Check if something was received and then

//return the item

while(!(UCSR0A & \_BV(RXC0)));

return (unsigned char) UDR0;

}

//Transmits a byte

/\*

\* Use this function if the TX interrupt is not enabled.

\* Blocks the serial port while TX completes

\*/

void putChar\_(unsigned char data)

{

//What until the buffer is empty

while(!(UCSR0A & \_BV(UDRE0)));

UDR0 = (unsigned char)data;

}

/\*A string print called printm that uses a

char array and your putchar clone to transmit

strings\*/

void printm(unsigned char \*str)

{

//While it's not NULL

while(\*str != '\0')

{

putChar\_(\*str);

++str;

}

}

/\*uses an uninitialized char array and your getchar clone to

construct a string for your ATmega328P \*/

const unsigned char\* scanm(void)

{

//Allocate buffer of size 256

//and a ptr to it

unsigned char buff[BUFF] = {0};

unsigned char\* ptr;

ptr = buff;

while((\*ptr = getChar\_())){

if(\*ptr == '\n'|| \*ptr == '\0' )break;

putChar\_(ptr);

++ptr;

}

return buff;

}

void ADC0init()

{

DDRC &= ~(0<<DDC0); // SET PC.0 as an input

ADCSRA = 0x87; // Enable ADC and CLK/128

ADMUX = (1<<REFS0); // VCC reference, ADC0 single ended input

}

double Convert(double val)

{

return (((val\*5.0)/1024.0));

}

void main()

{

char str[30];

double val;

unsigned int prev;

usart0\_init\_();

ADC0init();

while(1)

{

//Start conversion

ADCSRA |= (1 << ADSC);

//wait conversion to finish

while((ADCSRA & (1 << ADIF)) == 0);

//Get value that was converted

if(!((unsigned int)ADC == prev)){

prev = ADC;

val = Convert(prev);

*dtostrf*(val,1,3,str);

printm("Voltage Read: ");

printm(str);

putChar\_('\n');

putChar\_('\r');

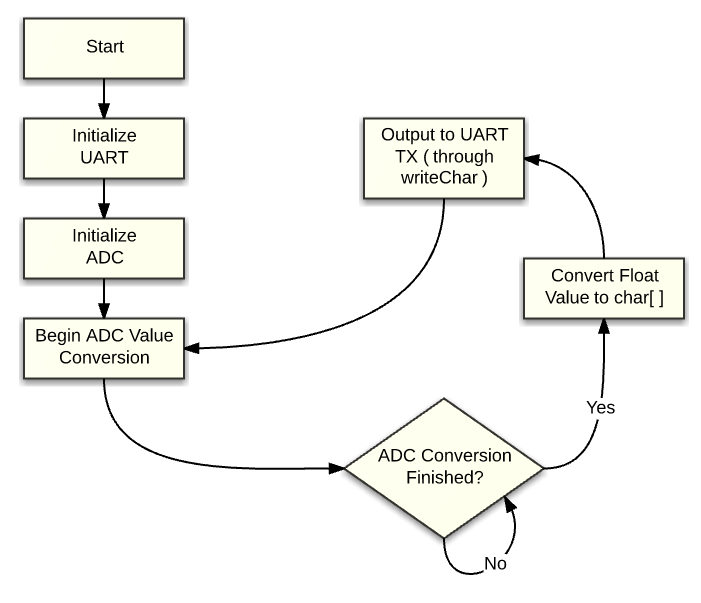
*\_delay\_ms*(1000);

}

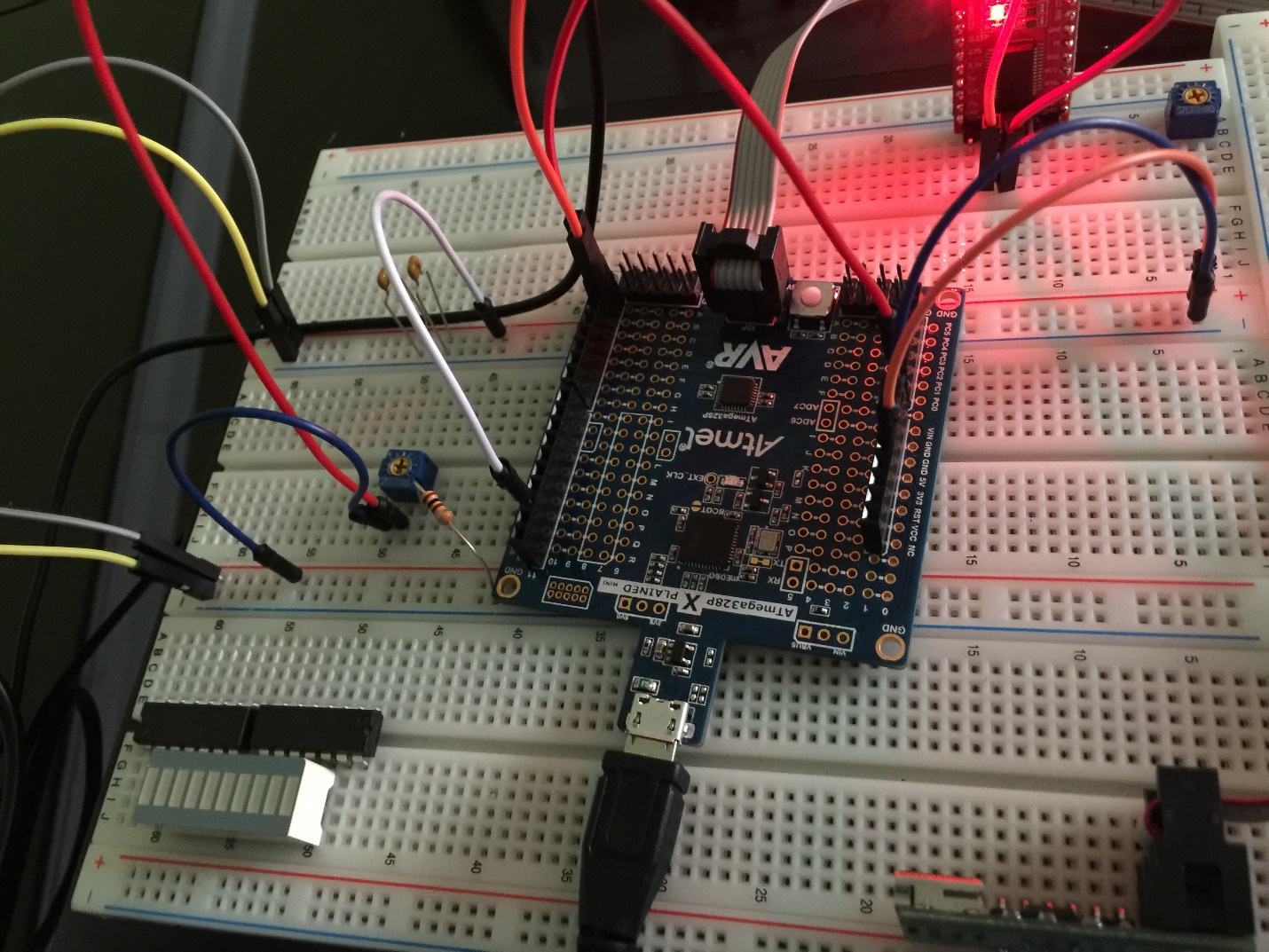
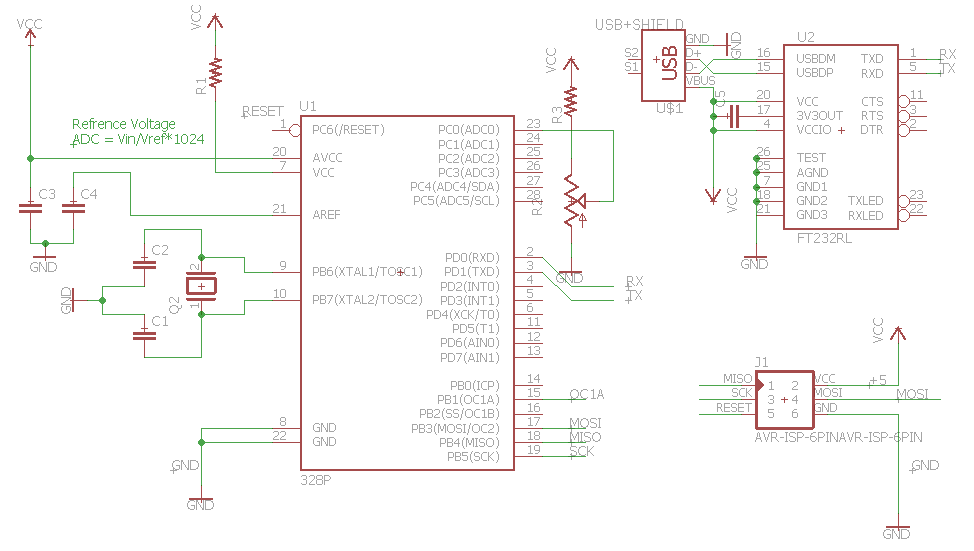
}

}

**PART C:Flow Chart**



**PART D: Schematic**



**PART E: Video**URL Video of Design Assignment 6: <https://youtu.be/Q0ZNh5p4kQE>