CPE301 – SPRING 2019

Design Assignment 3B

Student Name: Ron Joshua Recrio

Student #: 5003825419

Student Email: recrio@unlv.nevada.edu

Primary Github address: <https://github.com/recrio/submissions>

Directory: /DesignAssignments/DA3B

Submit the following for all Labs:

1. In the document, for each task submit the modified or included code (only) with highlights and justifications of the modifications. Also, include the comments.
2. Use the previously create a Github repository with a random name (no CPE/301, Lastname, Firstname). Place all labs under the root folder ESD301/DA, sub-folder named LABXX, with one document and one video link file for each lab, place modified asm/c files named as LabXX-TYY.asm/c.
3. If multiple asm/c files or other libraries are used, create a folder LabXX-TYY and place these files inside the folder.
4. The folder should have a) Word document (see template), b) source code file(s) and other include files, c) text file with youtube video links (see template).

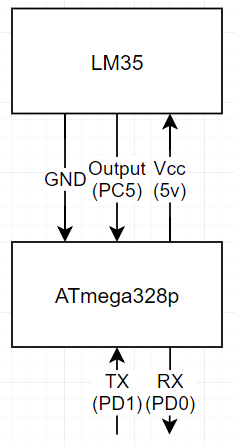
1. **COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS**

List of Components used:

ATmega328p Xplained Mini

LM35 Temperature Sensor

Block diagram with pins used in the Atmega328P



1. **INITIAL CODE OF TASK 1/A**

#define *F\_CPU* 16000000UL

#define PRESCALAR 1024

#define BAUDRATE 9600

#define BAUD\_PRESCALAR (((*F\_CPU* / (BAUDRATE \* 16UL))) - 1)

#define ONESEC (0xFFFF - ((*F\_CPU*/PRESCALAR)\*1) - 26)

#include <avr/io.h>

#include <avr/interrupt.h>

#include <stdio.h>

void USART\_init(void); // initializes USART settings

void USART\_sendChar(char ch); // sends a character

void USART\_sendString(char\* str); // sends a string

void TIMER\_init(void); // initializes timer sequence for interrupts

char num = '7'; // random number

char string[] = "Hello World!"; // basic string

float floating = 12.345; // random float value

char fl[20]; // character buffer for float value

int main(void)

{

*snprintf*(fl,sizeof(fl),"%f\r\n",floating); // converts floating value into string

USART\_init(); // initialize USART

TIMER\_init(); // initialize Timer/Interrupt

while (1) // Loop forever

{

}

}

void USART\_init( void )

{

UBRR0H = 0; // not needed

UBRR0L = BAUD\_PRESCALAR; // Baud Prescaler

UCSR0C = \_BV(UCSZ01) | \_BV(UCSZ00); /\* 8-bit data \*/

UCSR0B = \_BV(RXEN0) | \_BV(TXEN0); /\* Enable RX and TX \*/

}

void USART\_sendChar(char ch) {

while (!(UCSR0A & (1<<UDRE0))); // while data reg is not empty: hold

UDR0 = ch; // place character into reg

}

void USART\_sendString(char\* str) {

while ((\*str != '\0')) { // while not the end of the string

while (!(UCSR0A & (1<<UDRE0))); // while data reg is not empty: hold

UDR0 = \*str; //take in character to reg

str++; // next character

}

}

void TIMER\_init(void) {

TCNT1 = ONESEC; // ONESEC is the number to count up to 0xFFFF for 1 sec delay

TIMSK1 |= (1 << TOIE0); // Enable Interrupt for Timer1

sei(); // Enable Global Interrupt

TCCR1B |= (1<<CS12)|(1<<CS10); // Start timer 1 and set prescaler to 1024

}

ISR (TIMER1\_OVF\_vect) {

USART\_sendChar(num); // send char '7'

USART\_sendString(string); // send "Hello World!"

USART\_sendString(fl); // send 12.345

TCNT1 = ONESEC; // set the timer back

}

1. **DEVELOPED MODIFIED CODE OF TASK 1/B from TASK 1/A**

#define *F\_CPU* 16000000UL

#define PRESCALAR 1024

#define BAUDRATE 9600

#define BAUD\_PRESCALAR (((*F\_CPU* / (BAUDRATE \* 16UL))) - 1)

#define ONESEC (0xFFFF - ((*F\_CPU*/PRESCALAR)\*1) - 60)

#include <avr/io.h>

#include <avr/interrupt.h>

#include <stdio.h>

void USART\_init(void); // initializes USART settings

void USART\_sendChar(char ch); // sends a character

void USART\_sendString(char\* str); // sends a string

void TIMER\_init(void); // initializes timer sequence for interrupts

void ADC\_init(void); // initializes ADC settings

volatile int adc\_temp;

int main(void)

{

USART\_init(); // initialize USART

TIMER\_init(); // initialize Timer/Interrupt

ADC\_init();

while (1) // Loop forever

{

}

}

ISR (TIMER1\_OVF\_vect) {

TCNT1 = ONESEC; // set the timer back

ADCSRA |= (1<<ADSC); // start conversion

while ((ADCSRA&(1<<ADIF))==0){} // Wait for conversion

ADCSRA |= (1<<ADIF); // Clear Interrupt Flag

adc\_temp = ADCL; // take in lower bits first

adc\_temp = adc\_temp | (ADCH<<8); // take in upper bits

char temp[20]; // buffer

*snprintf*(temp,sizeof(temp),"%d\r\n",adc\_temp); // print to the buffer

USART\_sendString(temp); // send the temp out

}

void USART\_init( void )

{

UBRR0H = 0; // not needed

UBRR0L = BAUD\_PRESCALAR; // Baud Prescaler

UCSR0C = \_BV(UCSZ01) | \_BV(UCSZ00); /\* 8-bit data \*/

UCSR0B = \_BV(RXEN0) | \_BV(TXEN0); /\* Enable RX and TX \*/

}

void USART\_sendChar(char ch) {

while (!(UCSR0A & (1<<UDRE0))); // while data reg is not empty: hold

UDR0 = ch; // place character into reg

}

void USART\_sendString(char\* str) {

while ((\*str != '\0')) { // while not the end of the string

while (!(UCSR0A & (1<<UDRE0))); // while data reg is not empty: hold

USART\_sendChar(\*str); //take in character to reg

str++; // next character

}

}

void TIMER\_init(void) {

TCNT1 = ONESEC; // ONESEC is the number to count up to 0xFFFF for 1 sec delay

TIMSK1 |= (1 << TOIE0); // Enable Interrupt for Timer1

sei(); // Enable Global Interrupt

TCCR1B |= (1<<CS12)|(1<<CS10); // Start timer 1 and set prescaler to 1024

}

void ADC\_init(void) {

ADMUX |= (0<<REFS1)| // Reference Select

(1<<REFS0)| // Selected AVcc

(0<<ADLAR)| // Left Adjust Result OFF

(1<<MUX2) | // Analog Channel Select

(0<<MUX1) | // 1 0 1

(1<<MUX0) ; // Channel 5 or PC5

ADCSRA |= (1<<ADEN)| // Enable ADC

(0<<ADSC)| // Do not start conversion

(0<<ADATE)|// Auto Trigger Disabled

(0<<ADIF)| // Interrupt Flag Cleared

(0<<ADIE)| // Interrupt Disabled

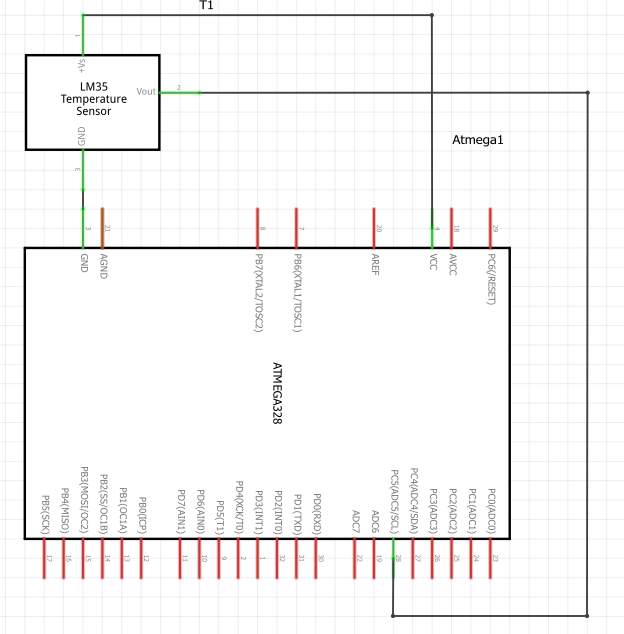
(1<<ADPS2)| // ADC Prescaler Select

(0<<ADPS1)| // Set to

(1<<ADPS0); // 32

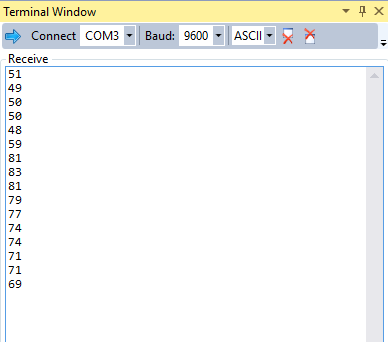
}

1. **SCHEMATICS**

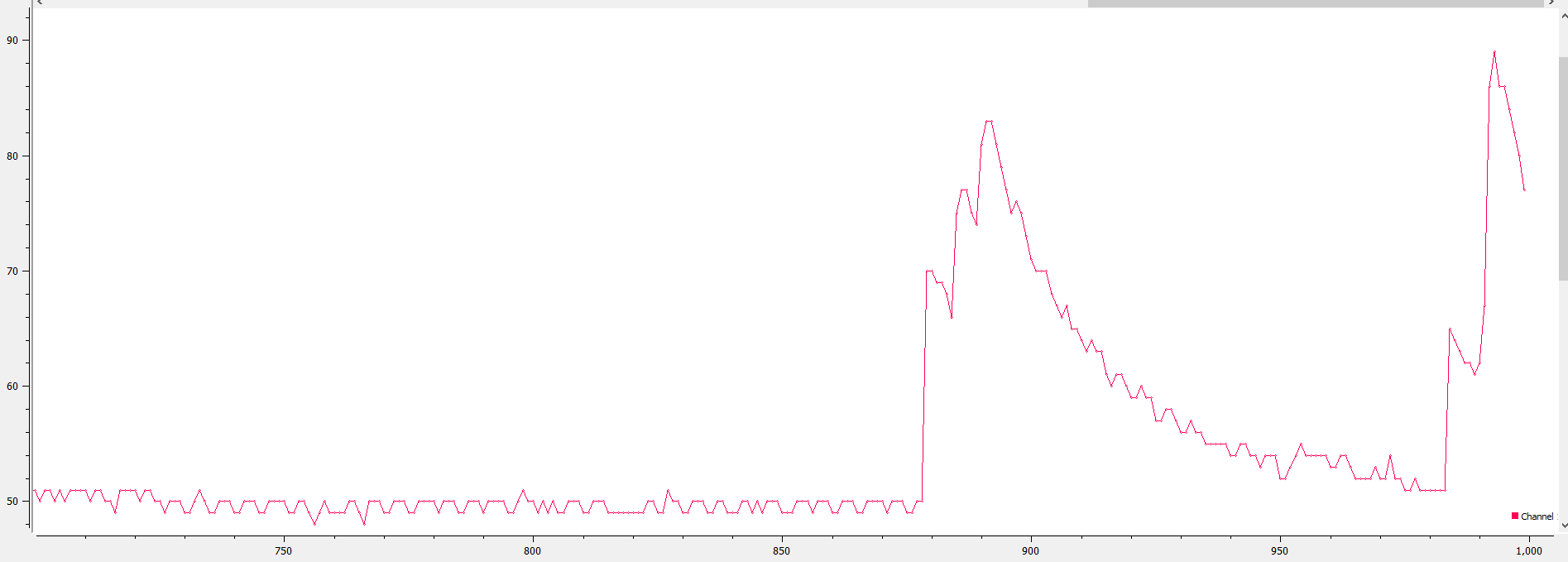


1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**

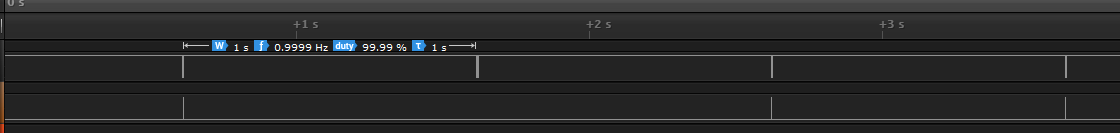
Terminal Output:

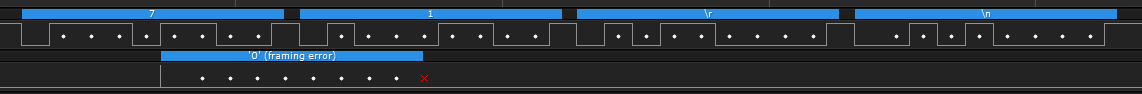


Serial Plot:

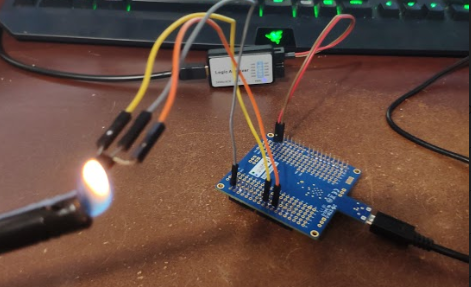


Logic Analyzer:





1. **SCREENSHOT OF EACH DEMO (BOARD SETUP)**



1. **VIDEO LINKS OF EACH DEMO**

<https://youtu.be/9m0bMNAhUso>

1. **GITHUB LINK OF THIS DA**

<https://github.com/recrio/submissions/tree/master/DesignAssignments/DA3B>

**Student Academic Misconduct Policy**

<http://studentconduct.unlv.edu/misconduct/policy.html>

“This assignment submission is my own, original work”.

Ron Joshua Recrio