CPE301 – SPRING 2019

Design Assignment 3B

Student Name: Rocky Yasuaki Gonzalez
Student #: 5003229733
Student Email: gonzar14@unlv.nevada.edu
Primary Github address: https://github.com/rockyg1995/ihswppdar.git
Directory: C:\Users\rocky\Documents\CpE 301+L - Embedded Systems Design\CpE 301\Repository\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

Atmega328PB Xplained Mini
Micro USB Cable (Power Supply)
x2 100nF Capacitors
10uH Inductor
Wire Connectors
LM35

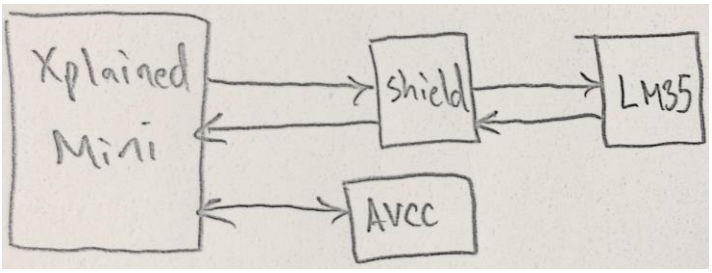

Figure 1 – Block Diagram

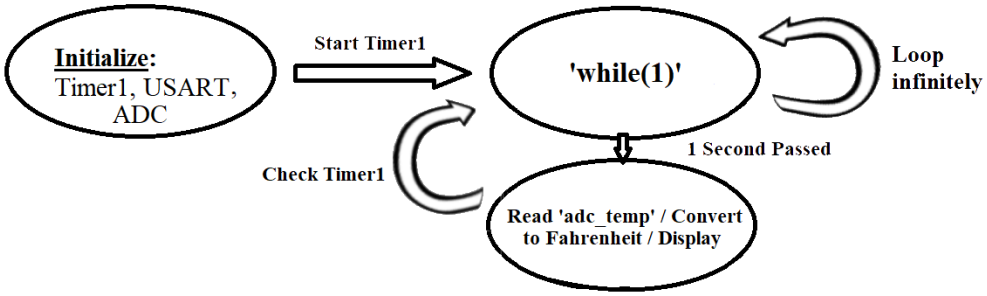

Figure 2 – Flow Chart for Coding Algorithm

2. INITIAL/DEVELOPED CODE OF TASK

```
/*
 * DA3B.c
 *
 * Created: 3/29/2019 6:58:23 PM
 * Author: rocky
 */

#define F_CPU 16000000UL           // Frequency of Xplained Mini (16MHz)
#include <avr/io.h>                // Standard AVR Library
#include <stdio.h>                 // AVR library containing printf functions
#include <avr/interrupt.h>         // AVR library containing interrupt functions
#include <util/delay.h>            // AVR library containing _delay_ms() function

#define BAUDRATE 9600              // Baudrate in Bits per second (bps)
#define BAUD_PRESCALLER ((F_CPU / (BAUDRATE * 16UL)) - 1)     // Baudrate Prescaler
#define MAX_1s 15624               // 1s delay: OCR1A = (16MHz*1000ms/prescaler) -1

//Declaration of our functions
void Timer1_init(void);            // Function to initialize Timer1
void USART_init(void);             // Function to initialize USART
```

```
void adc_init(void);              // Function to initialize Analog to Digital Converter

unsigned char USART_receive(void); // Function to receive Serial data from UDR0
void USART_send(unsigned char data);// Function to send individual char data into UDR0
void USART_putstring(char* StringPtr);// Function to break string into chars and send
void read_adc(void);              // Function to read temperature received from ADC

volatile float adc_temp;          // Stores ADC Value representing Temperature
char outs[20];                    // 'outs[]' used to store integer and float values into
                                  // array of chars size 20

int main(void) {


	TIMSK1 |= (1 << OCIE1A);    // Set Interrupt on Compare Match
	sei();                      // Enable Global Interrupts
	Timer1_init();              // Call the Timer1 initialization code
	USART_init();               // Call the USART initialization code
	adc_init();                 // Call the ADC initialization code
	USART_putstring("Connected!\r\n"); // Pass 'Connected!' to function to send chars
	_delay_ms(125);             // Wait a bit
	float adc_tempf;            // to store ADC Fahrenheit Temperature

	while (1) {                 // Infinite loop
		if (TCNT1 == OCR1A) {       // Display a String when Timer1 Matches OCR1A
			read_adc();         // Read value of ADC Temperature
			adc_tempf = (ADCH << 8) + ADCL;// Stores Temperature as float
			adc_tempf = (9/5)*adc_tempf + 32;  // Converts Celsius to Fahrenheit
			snprintf(outs,sizeof(outs),"%3f\r\n", adc_tempf);// Stores adc_temp
			USART_putstring(outs);      // Pass 'outs' to function to send chars
		}
	}
	return 0;
}
//-------------------------------------------------------------------------
void Timer1_init(void) {          // Function to Initialize Timer1 properties
	OCR1A = MAX_1s;             // Let the OCR1A be the Max Value used for 1s delay
	TCCR1A = (0<<COM1A1)|(0<<COM1A0);  // Normal Operation, Disconnect OC1A 'COM1A1:0'
	TCCR1A = (0<<COM1B1)|(0<<COM1B0);  // Normal Operation, Disconnect OC1B 'COM1B1:0'
	TCCR1A = (0<<WGM11)|(0<<WGM10);    // Set Timer1 to CTC Mode 'WGM1:0'
	TCCR1B = (0<<WGM13)|(1<<WGM12);    // Set Timer1 to CTC Mode 'WGM3:2'
	TCCR1B = (1<<CS12)|(0<<CS11)|(1<<CS10);   // Set Clock Select for Prescaler '1024'
}

ISR(TIMER1_COMPA_vect) {    // Function to Truncate Timer1 to delay properly for 1 second
	TCNT1 = 0;          // Restart Timer1 from the Beginning
	return;             // Resume code from where interrupt left off
}
//-------------------------------------------------------------------------
void USART_init(void) {                 // Function to Initialize USART properties
	UBRR0H = (uint8_t)(BAUD_PRESCALLER >> 8); // Store Upper Baudrate into UBRR0H
	UBRR0L = (uint8_t)(BAUD_PRESCALLER);      // Store Lower Baudrate into UBRR0L
	UCSR0B = (1 << RXEN0) | (1 << TXEN0);     // Enable Receiver and Enable Transmitter
	UCSR0C = (3 << UCSZ00);                   // Set UCSZ02:1 as 8-bit character data
}

unsigned char USART_receive(void) {       // Function to receive ASCII value from UDR0
	while (!(UCSR0A & (1 << RXC0)));  // Keep Checking until RXC0 is 'High' to break
```

```
	return UDR0;                        // Return received serial into unsigned char
}

void USART_send(unsigned char data) {     // Function to transmit ASCII value into UDR0
	while (!(UCSR0A & (1 << UDRE0)));  // Keep Checking until  UDRE0 'High' to break
	UDR0 = data;                        // Store unsigned char serial data into UDR0
}

void USART_putstring(char* StringPtr) {   // Function to break string, then USART_send()
	while (*StringPtr != 0x00) {        // Loop until String Completed (null/0-bits)
		USART_send(*StringPtr);     // Send unsigned char pointed by string pointer
		StringPtr++;                // Increment pointer to next char array location
	}
}
//---------------------------------------------------------------------
void adc_init(void) {
	ADMUX = (0<<REFS1)|(1<<REFS0)|     // Reference Selection, AVcc Ext cap at AREF
	(0<<ADLAR)|                        // ADC Left Adjust Result
	(0<<MUX3)|(1<<MUX2)|(0<<MUX1)|(1<<MUX0);// Analog Channel Selection 'ADC5/PC5'

	ADCSRA = (1<<ADEN)|                // ADC Enable
	(0<<ADSC)|                         // ADC Start Conversion
	(0<<ADATE)|                        // ADC Auto Trigger Enable
	(0<<ADIF)|                         // ADC Interrupt Flag
	(0<<ADIE)|                         // ADC Interrupt Enable
	(1<<ADPS2)|(0<<ADPS1)|(1<<ADPS0);  // ADC Prescaler Select Bits '32'
}

void read_adc(void) {
	unsigned char i = 4;               // Set 'i' for iterations
	adc_temp = 0;                      // set float 'adc_temp'
	while (i--) {                      // Decrement 'i' until 4 samples take
		ADCSRA |= (1<<ADSC);       // If ADSC is high (ADC Start Conversion)...
		while (ADCSRA & (1<<ADSC)); // Start the ADC Conversion
		adc_temp += ADC;           // Store the analog value on of current adc_temp
		_delay_ms(50);             // delay 50ms for sampling
	}
	adc_temp = (adc_temp/4);           // Average of 4 samples taken into adc_temp
}
//---------------------------------------------------------------------
```

3. SCHEMATICS

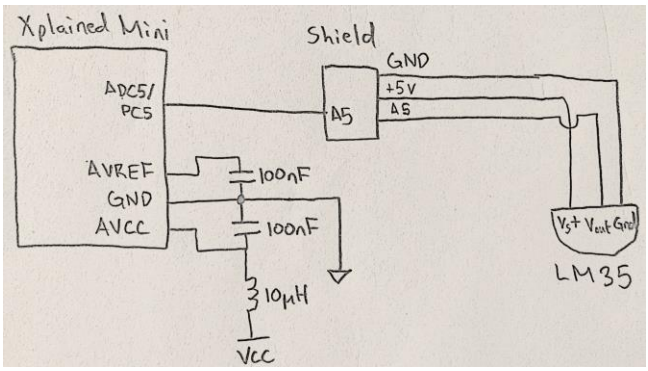

Figure 3 – Schematic of LM35 and Vcc Filter connected to Xplained Mini

4. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)

Since the Atmega328P/PB Xplained Mini has a built in AVR RX/TX module, we can send our serial data into a terminal which is able to display the temperature reading of ADC when connected to a temperature sensor (LM35) sending analog data into ADC channel 5. The output is portrayed in *Figure 4*:

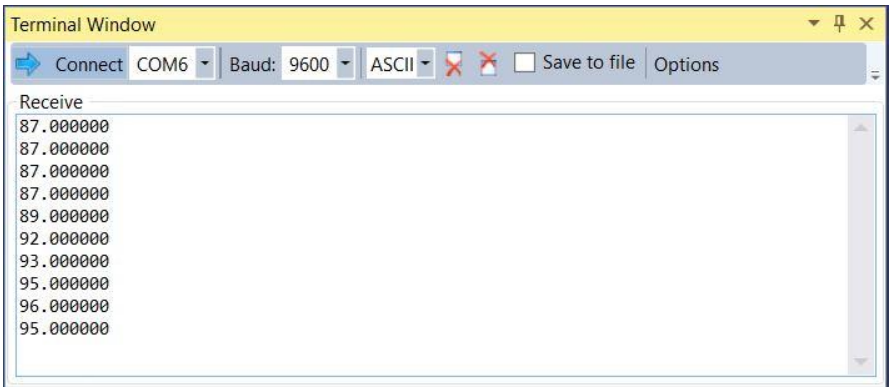

Figure 4 – Output Terminal of Serial Data

5. SCREENSHOT OF EACH DEMO (BOARD SETUP)

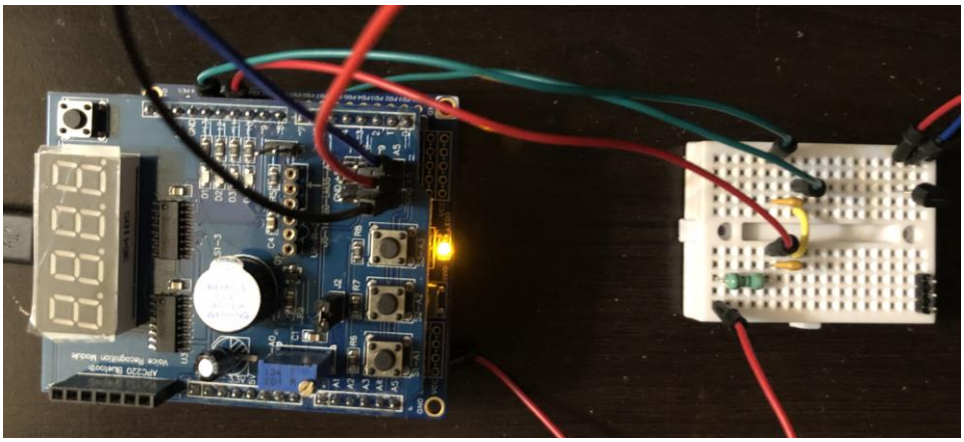

Figure 5 – Xplained Mini connected to Vcc Filter and LM35

6. VIDEO LINKS OF EACH DEMO

N/A (Not required for assignment)

7. GITHUB LINK OF THIS DA

https://github.com/rockyg1995/ihswppdar/tree/master/DesignAssignments/DA3B

Student Academic Misconduct Policy
http://studentconduct.unlv.edu/misconduct/policy.html

"*This assignment submission is my own, original work*".
Rocky Gonzalez