CPE301 – SPRING 2019

Design Assignment 5

Student Name: Brian West
Student #: 5003032874
Student Email: westb2@unlv.nevada.edu
Primary Github address: https://github.com/westbrian2/Spring2019
Directory: https://github.com/westbrian2/Spring2019/tree/master/DesignAssignments

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.

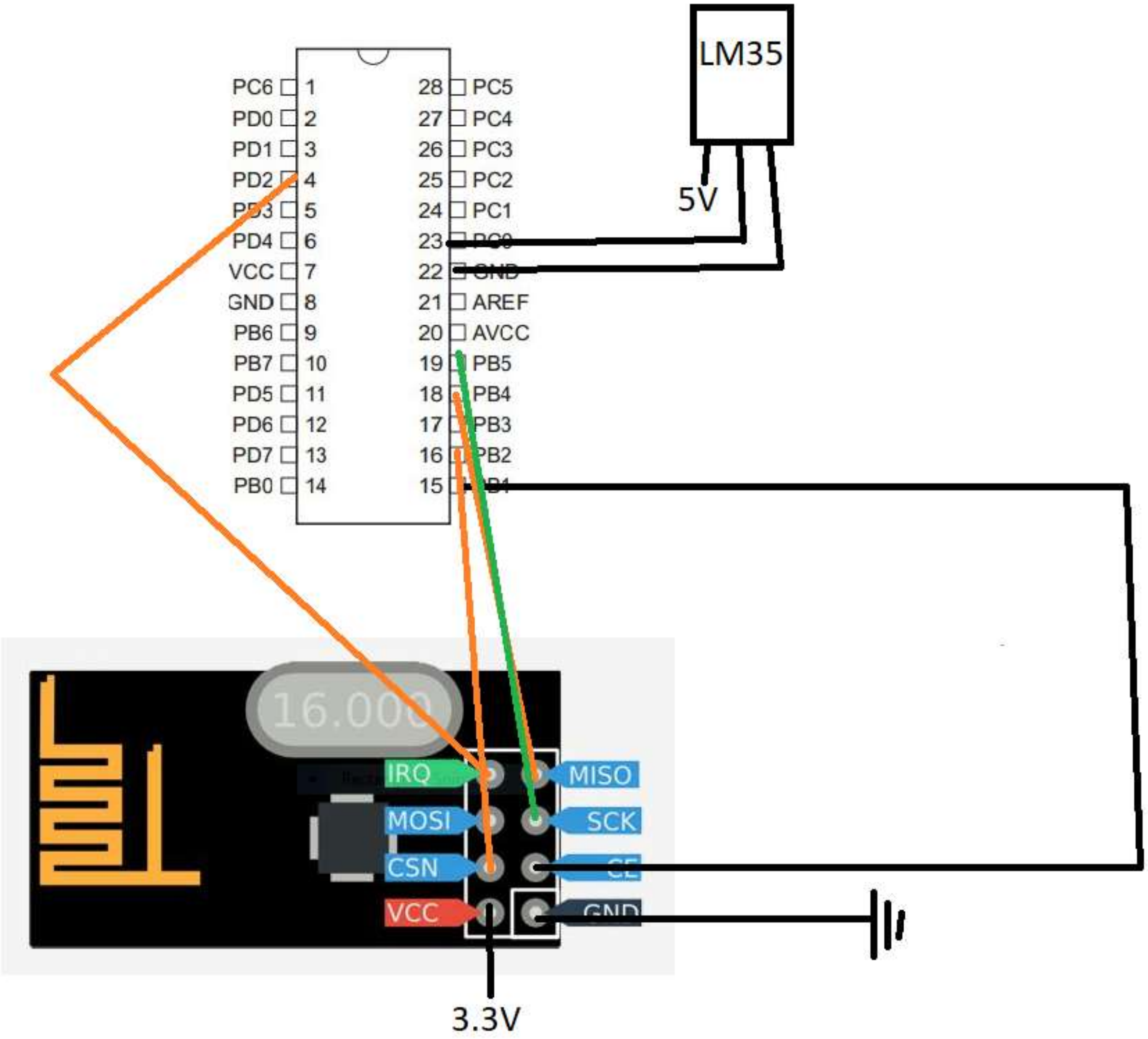
2. Use the previously create a Github repository with a random name (no CPE/301, Lastname, Firstname). Place all labs under the root folder ESD301/DA, sub-folder named LABXX, with one document and one video link file for each lab, place modified asm/c files named as LabXX-TYY.asm/c.

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

Xplained Mini
Nrf24L01
LM35



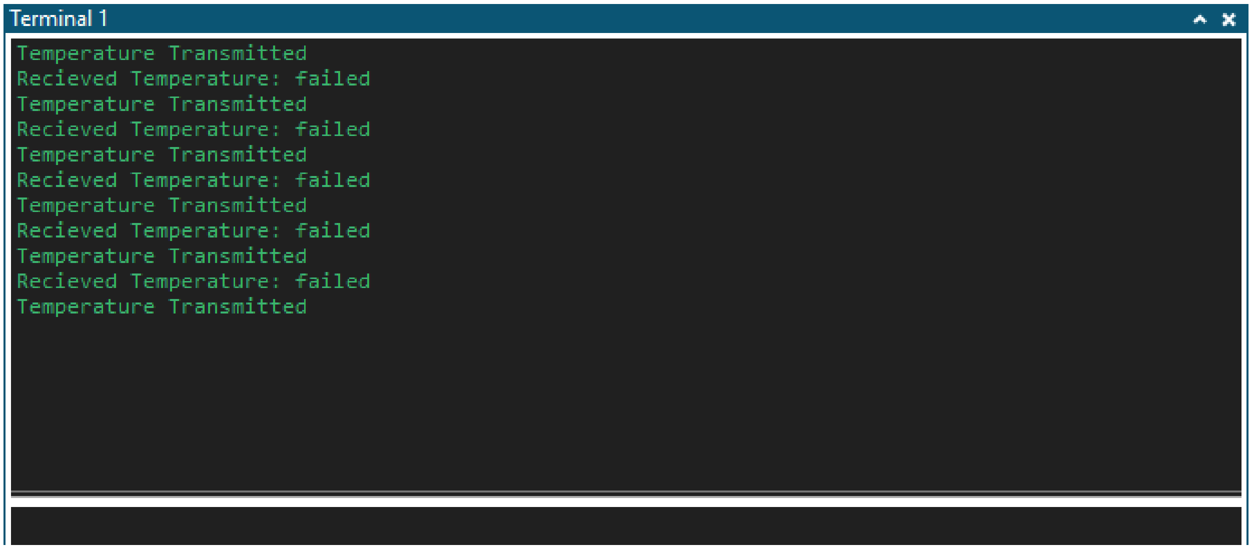
2. INITIAL/MODIFIED/DEVELOPED CODE OF TASK 1/A

```
#define F_CPU 16000000UL
#define UBRR_9600 103 //Baud rate for 16MHz
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#include <stdbool.h>
#include <stdio.h>
```

```
#include <string.h>

#include "nrf24l01.h"
#include "nrf24l01-mnemonics.h"
#include "spi.h"
//void print_config(void);

volatile bool message_received = false;
volatile bool status = false;

void read_adc(void);
void adc_init(void);

void USART_init( unsigned int ubrr ); //Sets up usart for use
void USART_tx_string( char *data ); //function that outputs data (usart)
volatile unsigned int adc_temp; //holds temp value
char output[32];
int main(){
        char tx_message[32];

        USART_init(UBRR_9600);
        adc_init();

        nrf24_init();
        nrf24_start_listening();

        while(1){
                read_adc();
                snprintf(output,sizeof(output),"%3d\r\n",adc_temp);
                strcpy(tx_message,output);
                nrf24_send_message(tx_message);
                _delay_ms(1000);
                sei();
                        if(message_received){
                                message_received=false;
                                snprintf(output,sizeof(output),"Recieved Temperature:
%s\n",nrf24_read_message());
                                USART_tx_string(output);
                                _delay_ms(500);
                                status=nrf24_send_message(tx_message);
                                if(status==true)
                                        USART_tx_string("Temperature Transmitted\n");
                }
        }
}

void USART_init(unsigned int ubrr){
        UBRR0H=(unsigned char)(ubrr>>8); //Setting up
        UBRR0L=(unsigned char)(ubrr);
        UCSR0B=(1<<TXEN0)|(1<<RXEN0);//Enabling reciever, transmitter, and rx interrupt
        UCSR0C=(1<<UCSZ01)|(1<<UCSZ00); //async  8 n 1
}
```

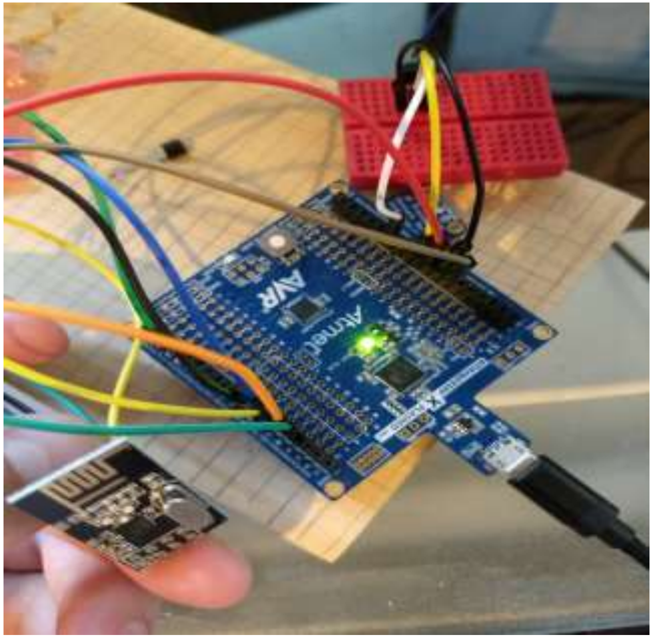
```
void USART_tx_string(char *data){ //sends string
        while((*data!= ’\0’)){
                while(!(UCSR0A&(1<<UDRE0)));
                UDR0=*data;
                data++; //gets next part of data
        }
}
void adc_init(void)
{
        /** Setup and enable ADC **/
        ADMUX = (0<<REFS1)| // Reference Selection Bits
        (1<<REFS0)| // AVcc - external cap at AREF
        (0<<ADLAR)| // ADC Left Adjust Result
        (0<<MUX2)| // setting input to PC0
        (0<<MUX1)|
        (0<<MUX0);
        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)| // ADC Prescaler Select Bits
        (0<<ADPS1)|
        (1<<ADPS0); // Select Channel
}
void read_adc(void) {
        unsigned char i =4; //to get 4 samples
        adc_temp = 0;
        while (i--) {
                ADCSRA |= (1<<ADSC); //start conversion
                while(ADCSRA & (1<<ADSC)); //waiting for coversion to finish
                adc_temp+= ADC;

                _delay_ms(50);
        }
        adc_temp = (adc_temp / 4)-20; // Average a few samples and adjusts for slight offset
}
//Interrupt on IRQ pin
ISR(INT0_vect)
{
        message_received = true;
}
```

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



```
Temperature Transmitted
Recieved Temperature: failed
Temperature Transmitted
Recieved Temperature: failed
Temperature Transmitted
Recieved Temperature: failed
Temperature Transmitted
Recieved Temperature: failed
Temperature Transmitted
Recieved Temperature: failed
Temperature Transmitted
```

4. SCREENSHOT OF EACH DEMO (BOARD SETUP)



5. GITHUB LINK OF THIS DA

https://github.com/westbrian2/Spring2019/tree/master/DesignAssignments/DA5_submission

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

"This assignment submission is my own, original work".
Brian West