CPE301 – SPRING 2019

Design Assignment 5

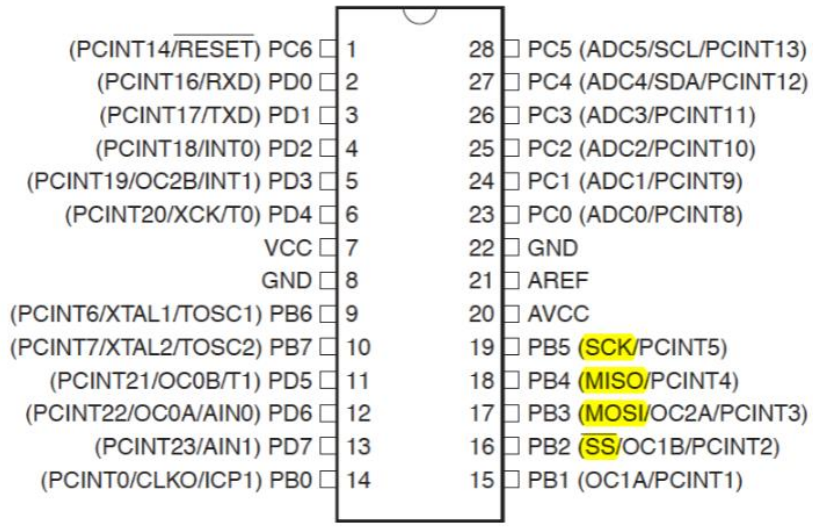
Student Name: Tyler Gardenhire
Student #: 8000450294
Student Email: gardenhi@unlv.nevada.edu
Primary Github address: gardenhi@unlv.nevada.edu
Directory: https://github.com/tylergardenhire/submission_projects.git

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

Atmel Studio 7 w/ AVR assembly and simulator, Atmega328p board.



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

Task 1 C code:

```
//set clock frequency
#define F_CPU 16000000UL

#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#include <stdbool.h>
#include <stdio.h>
#include <string.h>

//     Set up UART for printf();
#define BAUD 9600
#include "inc\STDIO_UART.c"

//     Include nRF24L01+ library
#include "inc\nrf24l01.c"
#include "inc\nrf24l01-mnemonics.h"
#include "inc\spi.c"
void print_config(void);

//     Used in IRQ ISR
volatile bool message_received = false;
volatile bool status = false;

void ADC_init (void);
volatile unsigned char ADCtemp[5];
volatile uint8_t ADCvalue;

int main(void)
{
       //set message to send
       char tx_message[32];                     //define string array
       strcpy(tx_message,"It's working!");      //copy string into array

       uart_init(); //initialize UART
       ADC_init(); //initialize ADC
       nrf24_init(); //initialize nRF24L01+
       print_config();
       nrf24_start_listening(); //listen to incoming messages
       strcpy(tx_message,"Initializing Chat Room..."); //copy string into array
       nrf24_send_message(tx_message);

       while (1)
       {
              if (message_received)
              {
                     //print message
                     message_received = false;
                     printf("Received Temperature: %s\n\n",nrf24_read_message());

                     //     Send message as response
                     _delay_ms(500);
                     status = nrf24_send_message(ADCtemp);
```

```
                        if (status == true) printf("Temperature Successfully Sent\n\n");
                }
        }
}

//interrupt on IRQ pin
ISR(INT0_vect)
{
        message_received = true;
}

//interrupt for Temperature Sensor
ISR(ADC_vect)
{
        volatile unsigned int j=0;
        char temp[5];

        ADCvalue = (ADCH << 1);             //shifts the value left to one place
        itoa(ADCvalue, temp, 10);    //converts integers to string

        while (j<5)                                  //transfers the temp string from itoa()
to ADCtemp
        {
                ADCtemp[j] = temp[j];
                j++;
        }
}

//prints configuration
void print_config(void)
{
        uint8_t data;
        printf("Startup successful\n\n nRF24L01+ configured as:\n");
        printf("-----------------------------------------\n");

        nrf24_read(CONFIG,&data,1);
        printf("CONFIG              0x%x\n",data);
        nrf24_read(EN_AA,&data,1);
        printf("EN_AA               0x%x\n",data);
        nrf24_read(EN_RXADDR,&data,1);
        printf("EN_RXADDR           0x%x\n",data);
        nrf24_read(SETUP_RETR,&data,1);
        printf("SETUP_RETR          0x%x\n",data);
        nrf24_read(RF_CH,&data,1);
        printf("RF_CH               0x%x\n",data);
        nrf24_read(RF_SETUP,&data,1);
        printf("RF_SETUP            0x%x\n",data);
        nrf24_read(STATUS,&data,1);
        printf("STATUS              0x%x\n",data);
        nrf24_read(FEATURE,&data,1);
        printf("FEATURE             0x%x\n",data);

        printf("-----------------------------------------\n\n");
}

void ADC_init (void)
{
        //ADC Multiplexer Selection Register
```

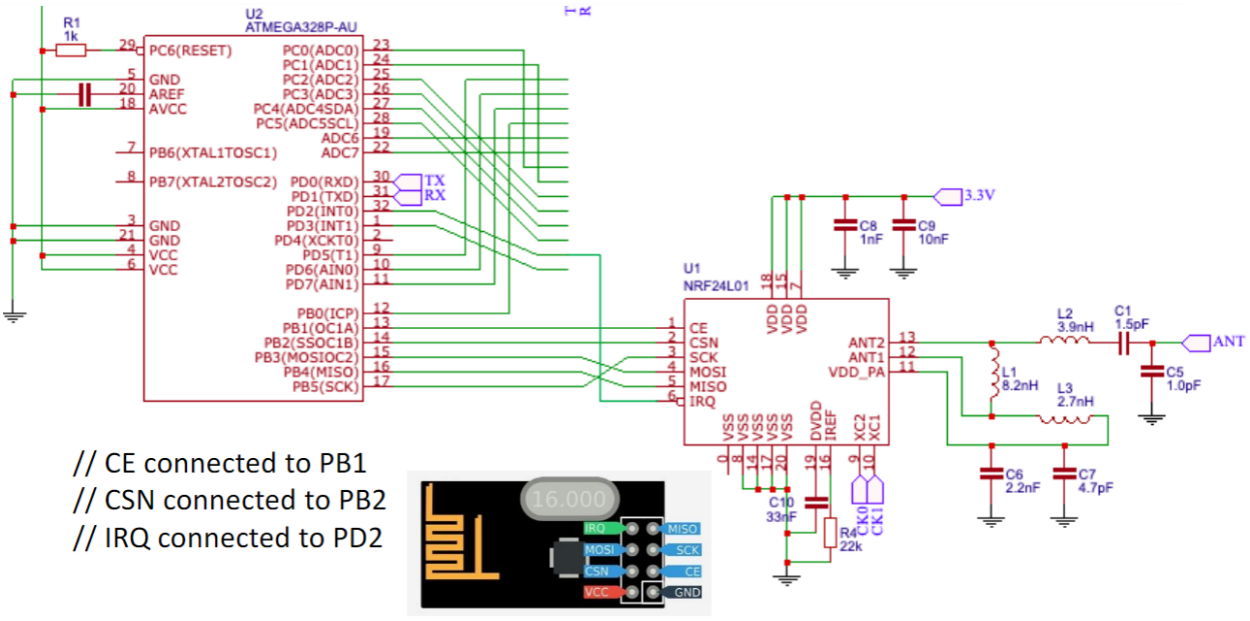
```
        ADMUX = (1 << REFS0) |              //voltage reference during conversion, "AVcc
with external capacitor at AREF pin"
        (1 << ADLAR);                       //left adjust ADC conversion result in ADC Data
Register

        //ADC Control and Status Register A
        ADCSRA = (1 << ADEN) |              // ADC enable
        (1 << ADSC) |         //ADC Start Conversion
        (1 << ADATE) |                //ADC Auto Trigger enable
        (1 << ADIE) |         //ADC Interrupt enable
        (1 << ADPS2) |
        (1 << ADPS1) |
        (1 << ADPS0);         //ADPS2:0 = 111 = 128 prescaler
}
```

3. DEVELOPED MODIFIED CODE OF TASK 2/A from TASK 1/A

N/A

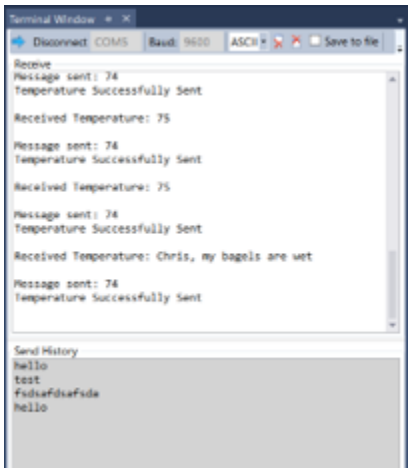
4. SCHEMATICS



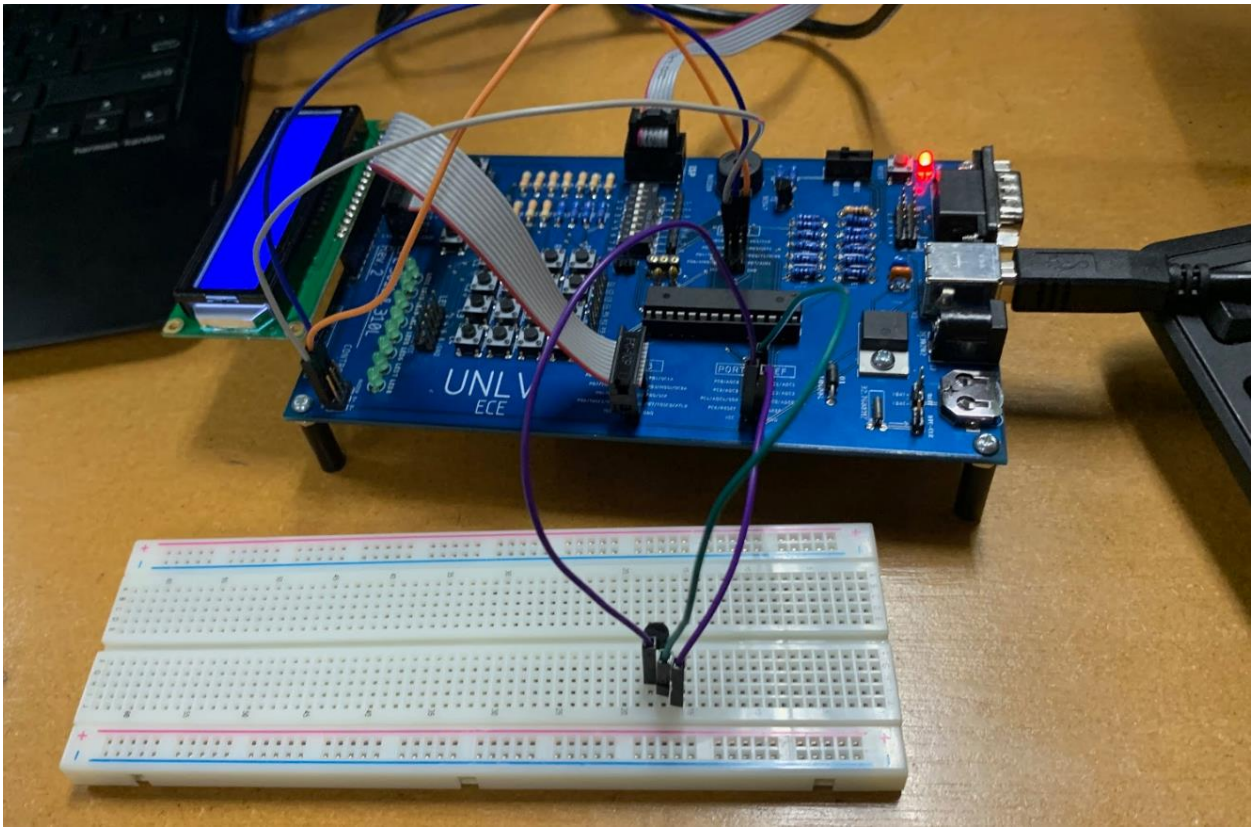
// CE connected to PB1
// CSN connected to PB2
// IRQ connected to PD2

N/A

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



6. SCREENSHOT OF EACH DEMO (BOARD SETUP)



7. VIDEO LINKS OF EACH DEMO

N/A

8. GITHUB LINK OF THIS DA

https://github.com/tylergardenhire/submission_projects.git

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

"*This assignment submission is my own, original work*".
TYLER GARDENHIRE