CPE301 – SPRING 2019

Design Assignment 5

Student Name: Shaquille Regis
Student #: 2000686590
Student Email: regis@unlv.nevada.edu
Primary Github address: https://github.com/regis-shaquille/submissions-SR
Directory: https://github.com/regis-shaquille/submissions-SR/tree/master/Design%20Assignments

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).

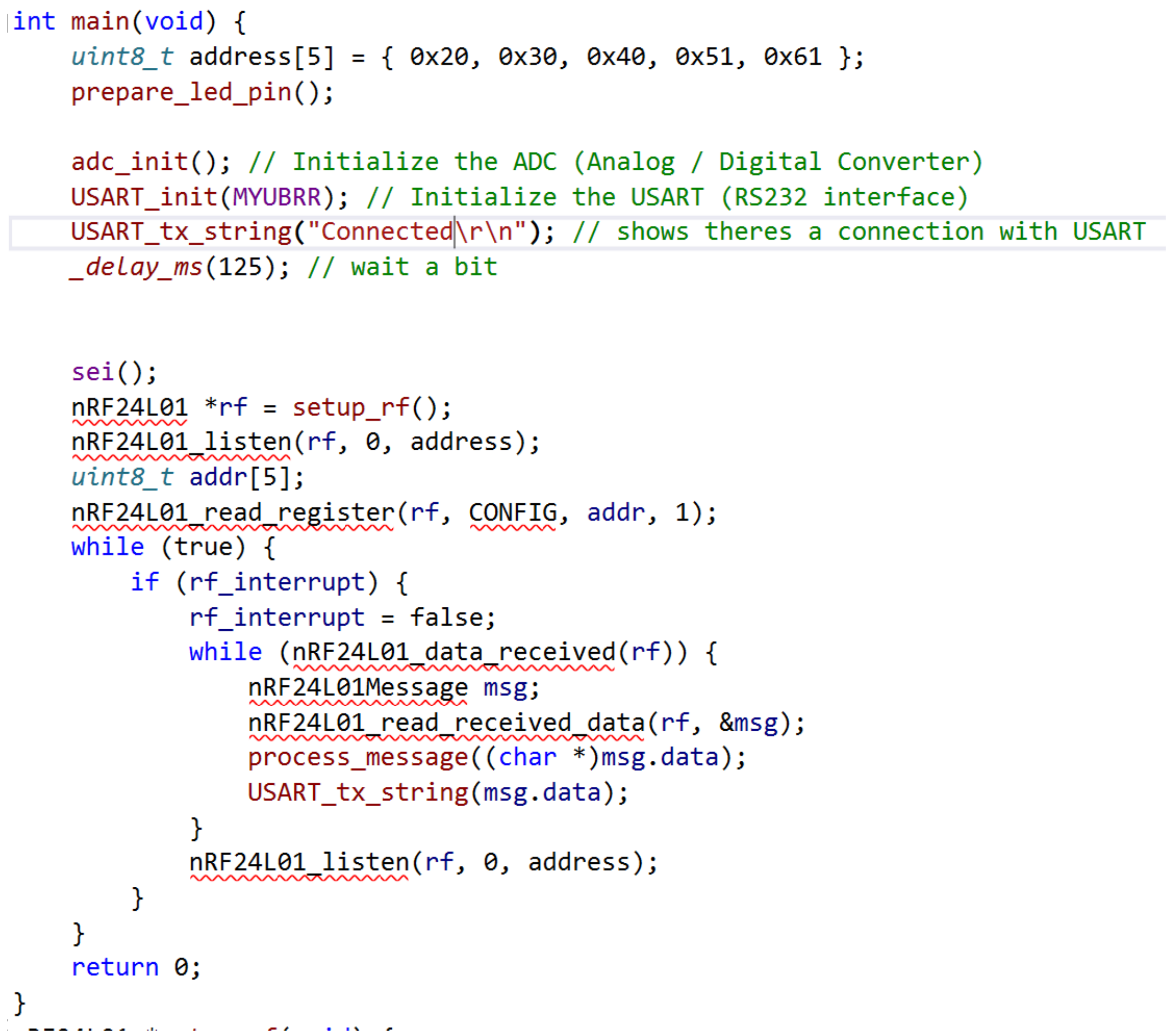
1. COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS

List of Components used
Block diagram with pins used in the Atmega328P

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

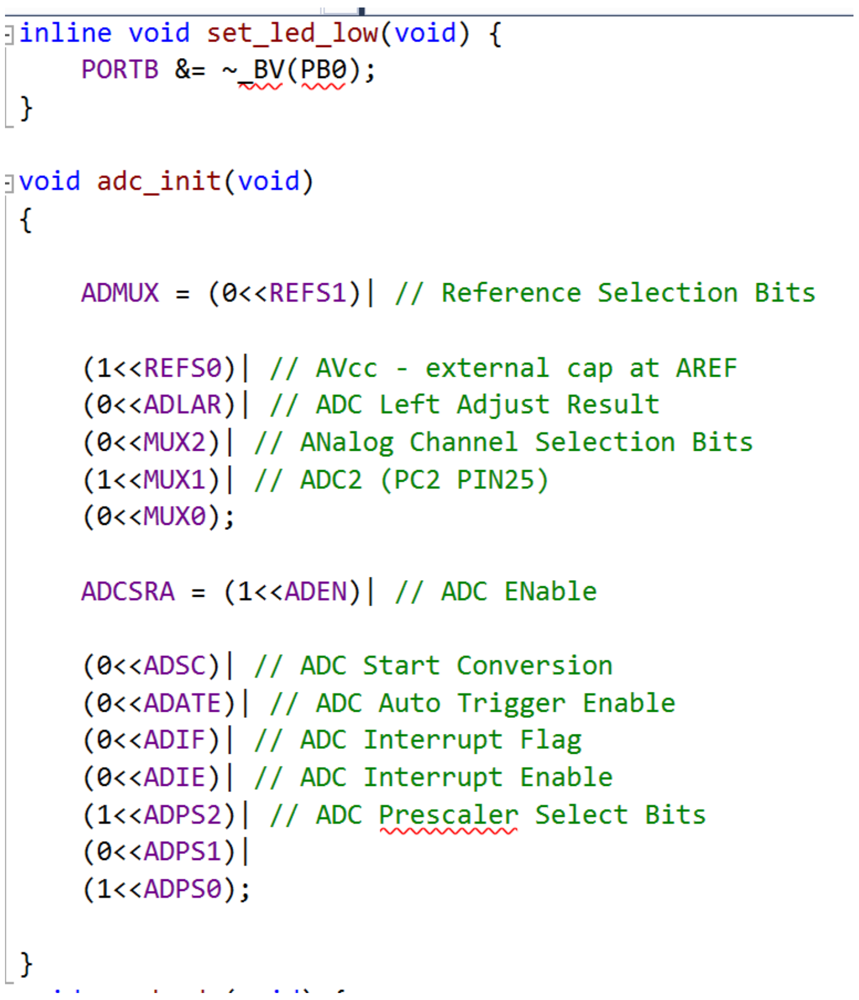
Transmit

```
#define F_CPU 16000000UL //16MHz
#define BAUD 9600 //Baud Rate
#define MYUBRR F_CPU/16/BAUD-1 //calculate Baud

#include <avr/io.h>
#include <avr/interrupt.h>
#include <stdbool.h>
#include <string.h>
#include <util/delay.h>
#include "nrf24l01.h"
#include "nrf24l01-mnemonics.h"

nRF24L01 *setup_rf(void);
void process_message(char *message);
inline void prepare_led_pin(void);
inline void set_led_high(void);

inline void set_led_low(void);
volatile bool rf_interrupt = false;


void read_adc(void);  //Read ADC
void adc_init(void);  //initialize ADC
void USART_init( unsigned int ubrr ); //initialize USART
void USART_tx_string(char *data); //Print String USART
volatile unsigned int adc_temp;
char outs[20]; //array
```

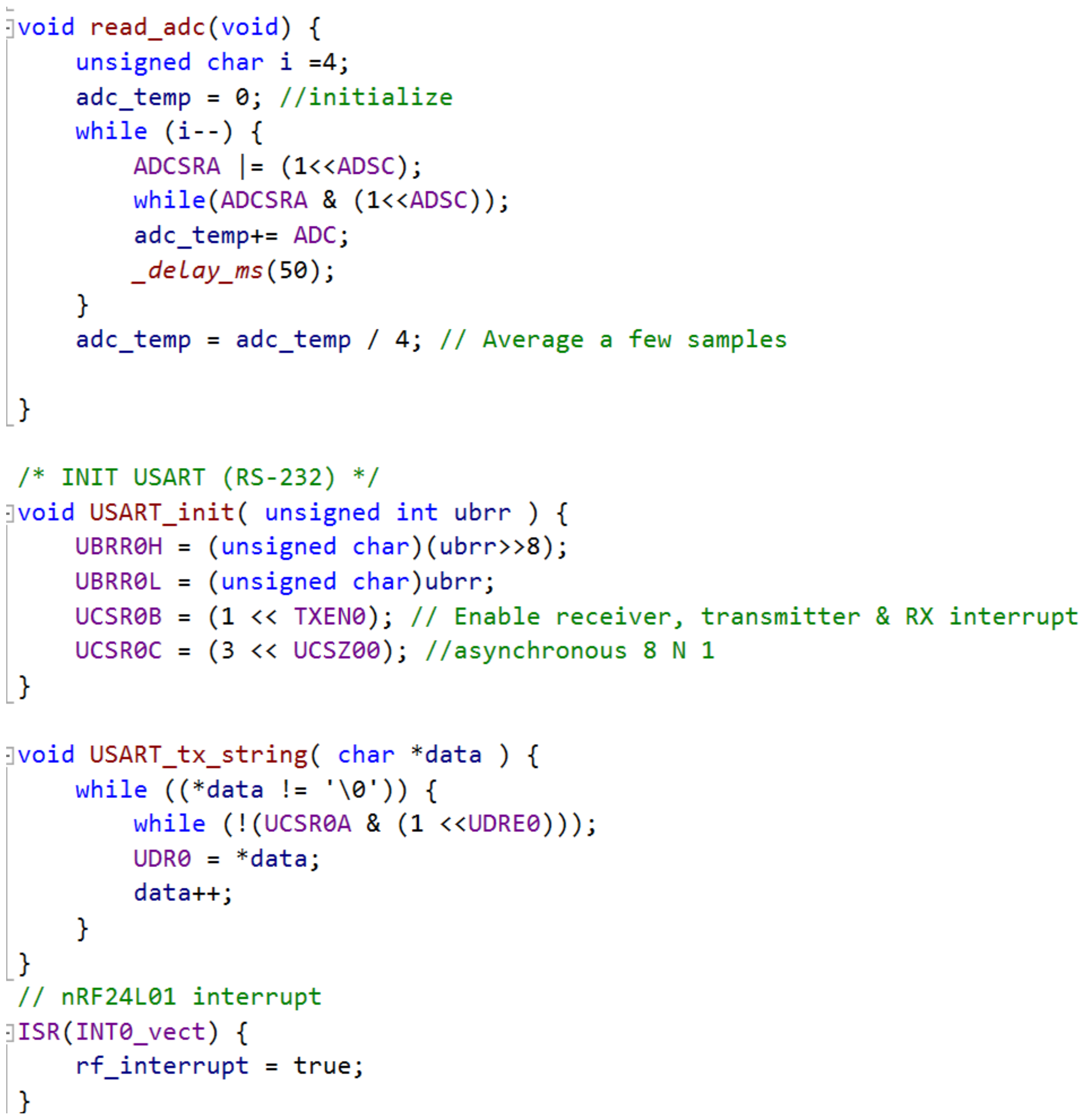


```
int main(void) {
    uint8_t address[5] = { 0x20, 0x30, 0x40, 0x51, 0x61 };
    prepare_led_pin();

    adc_init(); // Initialize the ADC (Analog / Digital Converter)
    USART_init(MYUBRR); // Initialize the USART (RS232 interface)
    USART_tx_string("Connected\r\n"); // shows theres a connection with USART
    _delay_ms(125); // wait a bit


    sei();
    nRF24L01 *rf = setup_rf();
    nRF24L01_listen(rf, 0, address);
    uint8_t addr[5];
    nRF24L01_read_register(rf, CONFIG, addr, 1);
    while (true) {
        if (rf_interrupt) {
            rf_interrupt = false;
            while (nRF24L01_data_received(rf)) {
                nRF24L01Message msg;
                nRF24L01_read_received_data(rf, &msg);
                process_message((char *)msg.data);
                USART_tx_string(msg.data);
            }
            nRF24L01_listen(rf, 0, address);
        }
    }
    return 0;
}
```

```c
nRF24L01 *setup_rf(void) {
    nRF24L01 *rf = nRF24L01_init();
    rf->ss.port = &PORTB;
    rf->ss.pin = PB2;
    rf->ce.port = &PORTB;
    rf->ce.pin = PB1;
    rf->sck.port = &PORTB;
    rf->sck.pin = PB5;
    rf->mosi.port = &PORTB;
    rf->mosi.pin = PB3;
    rf->miso.port = &PORTB;
    rf->miso.pin = PB4;
    // interrupt on falling edge of INT0 (PD2)
    EICRA |= _BV(ISC01);
    EIMSK |= _BV(INT0);
    nRF24L01_begin(rf);
    return rf;
}

void process_message(char *message) {
    if (strcmp(message, "ON") == 0)
    set_led_high();
    else if (strcmp(message, "OFF") == 0)
    set_led_low();
}
inline void prepare_led_pin(void) {
    DDRB |= _BV(PB0);
    PORTB &= ~_BV(PB0);
}
inline void set_led_high(void) {
    PORTB |= _BV(PB0);
}
```



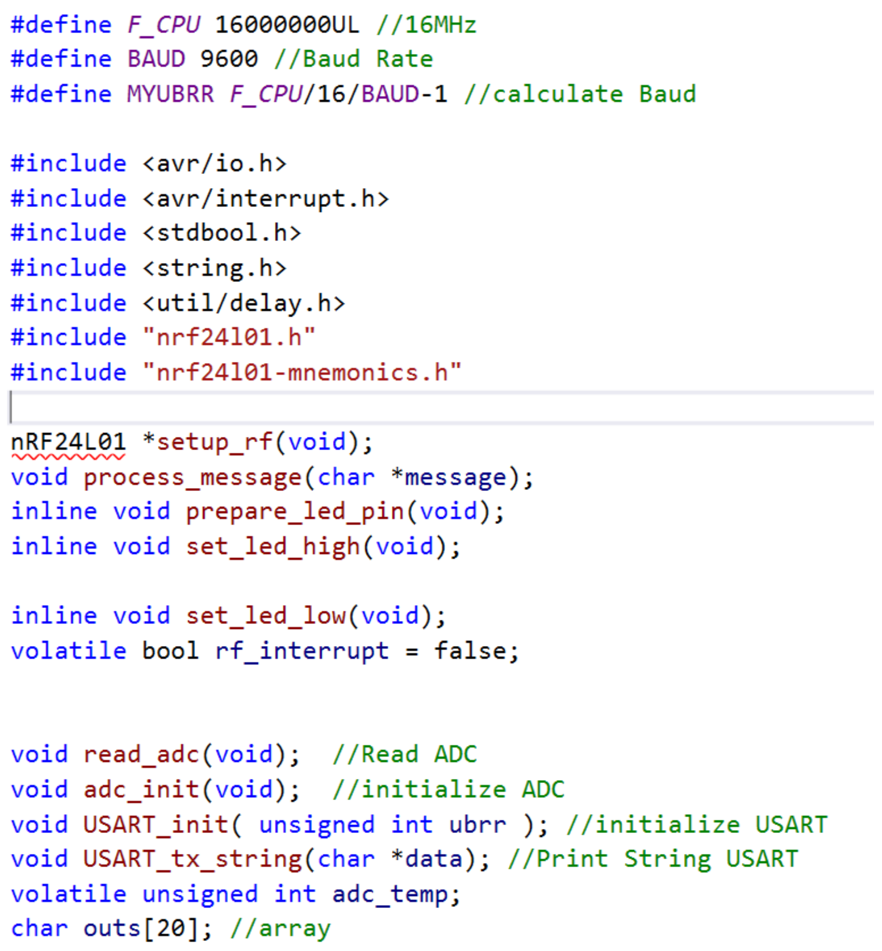
```c
inline void set_led_low(void) {
    PORTB &= ~_BV(PB0);
}

void adc_init(void)
{

    ADMUX = (0<<REFS1)| // Reference Selection Bits

    (1<<REFS0)| // AVcc - external cap at AREF
    (0<<ADLAR)| // ADC Left Adjust Result
    (0<<MUX2)| // ANalog Channel Selection Bits
    (1<<MUX1)| // ADC2 (PC2 PIN25)
    (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);

}
```



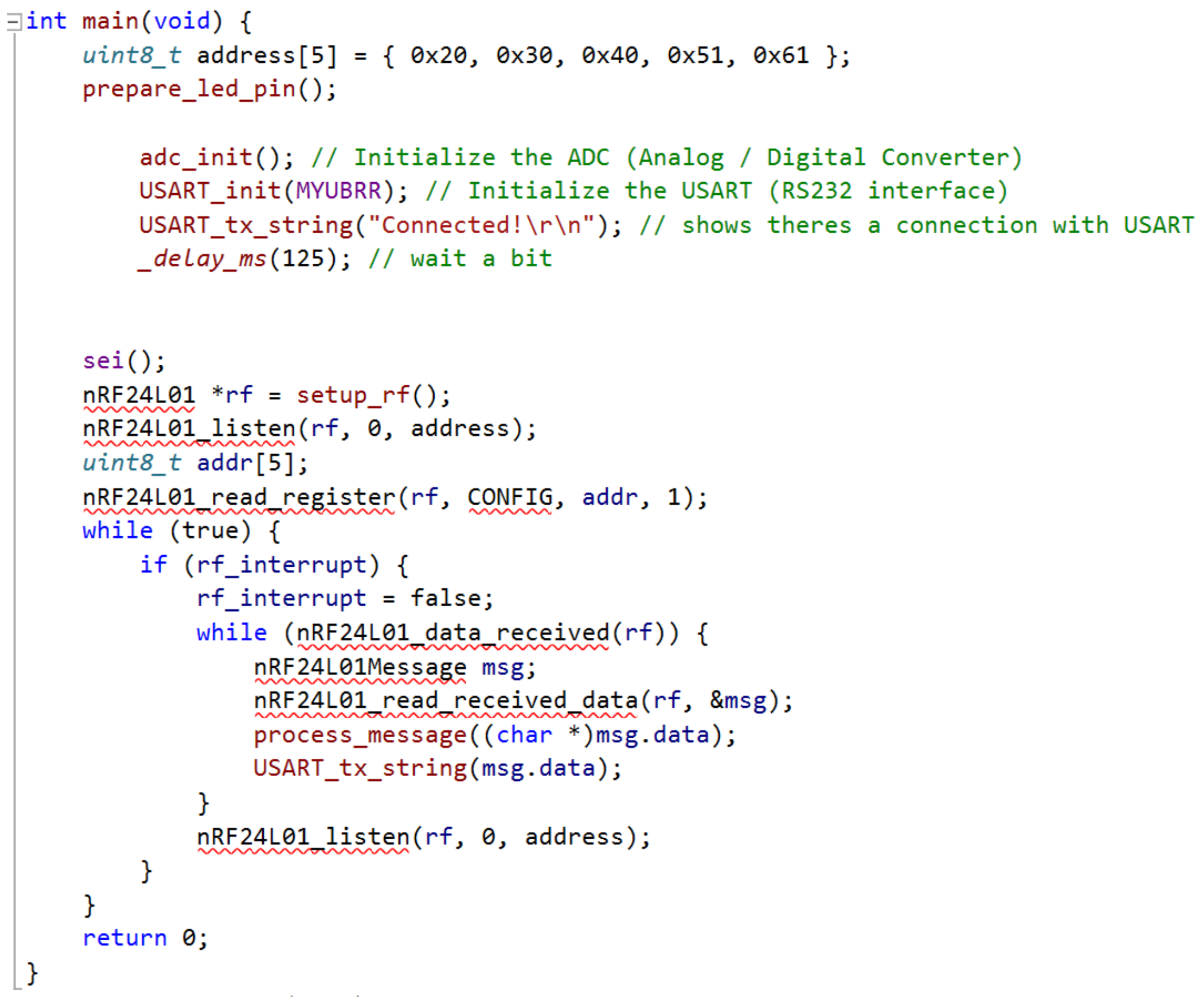
```
void read_adc(void) {
    unsigned char i =4;
    adc_temp = 0; //initialize
    while (i--) {
        ADCSRA |= (1<<ADSC);
        while(ADCSRA & (1<<ADSC));
        adc_temp+= ADC;
        _delay_ms(50);
    }
    adc_temp = adc_temp / 4; // Average a few samples

}

/* INIT USART (RS-232) */
void USART_init( unsigned int ubrr ) {
    UBRR0H = (unsigned char)(ubrr>>8);
    UBRR0L = (unsigned char)ubrr;
    UCSR0B = (1 << TXEN0); // Enable receiver, transmitter & RX interrupt
    UCSR0C = (3 << UCSZ00); //asynchronous 8 N 1
}

void USART_tx_string( char *data ) {
    while ((*data != '\0')) {
        while (!(UCSR0A & (1 <<UDRE0)));
        UDR0 = *data;
        data++;
    }
}
// nRF24L01 interrupt
ISR(INT0_vect) {
    rf_interrupt = true;
}
```

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

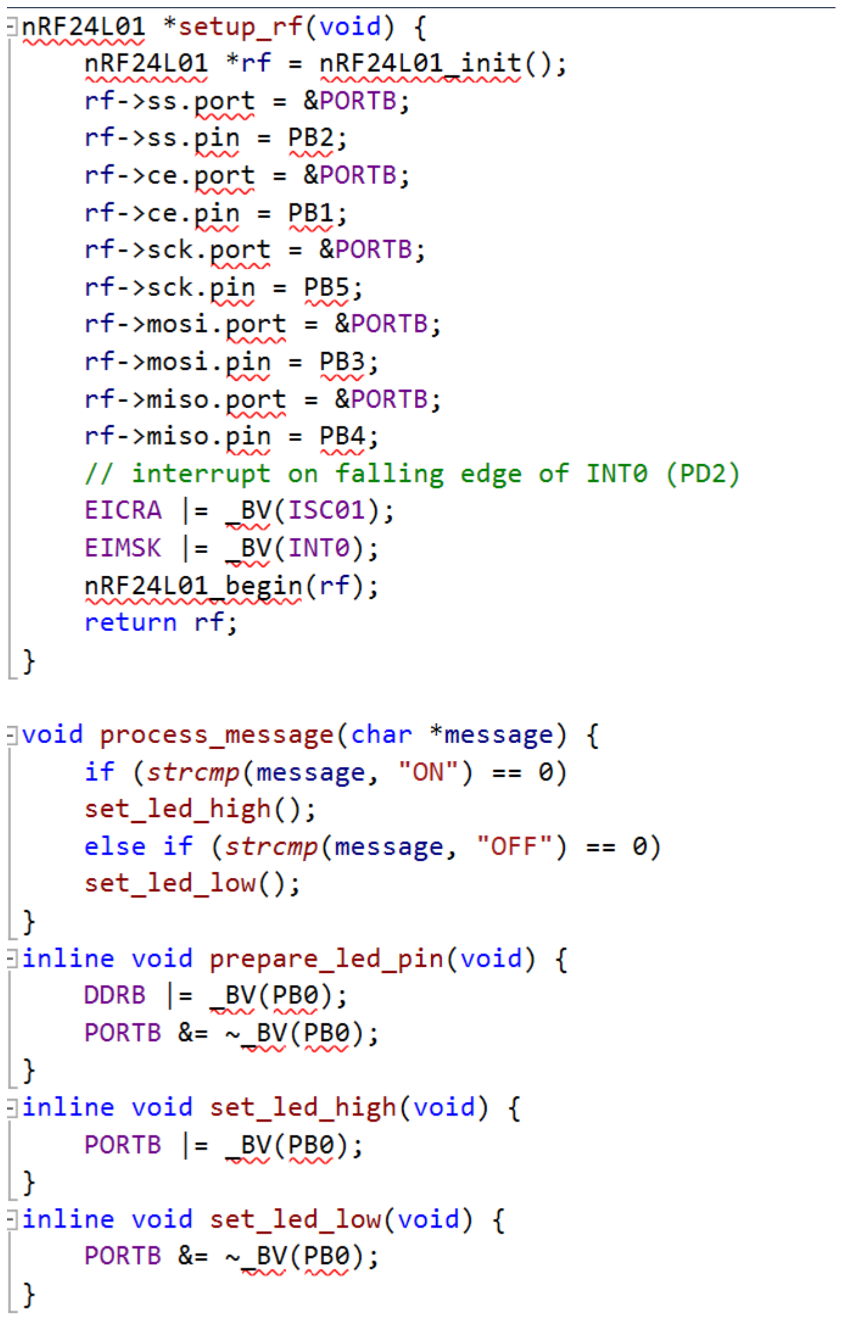
Receive



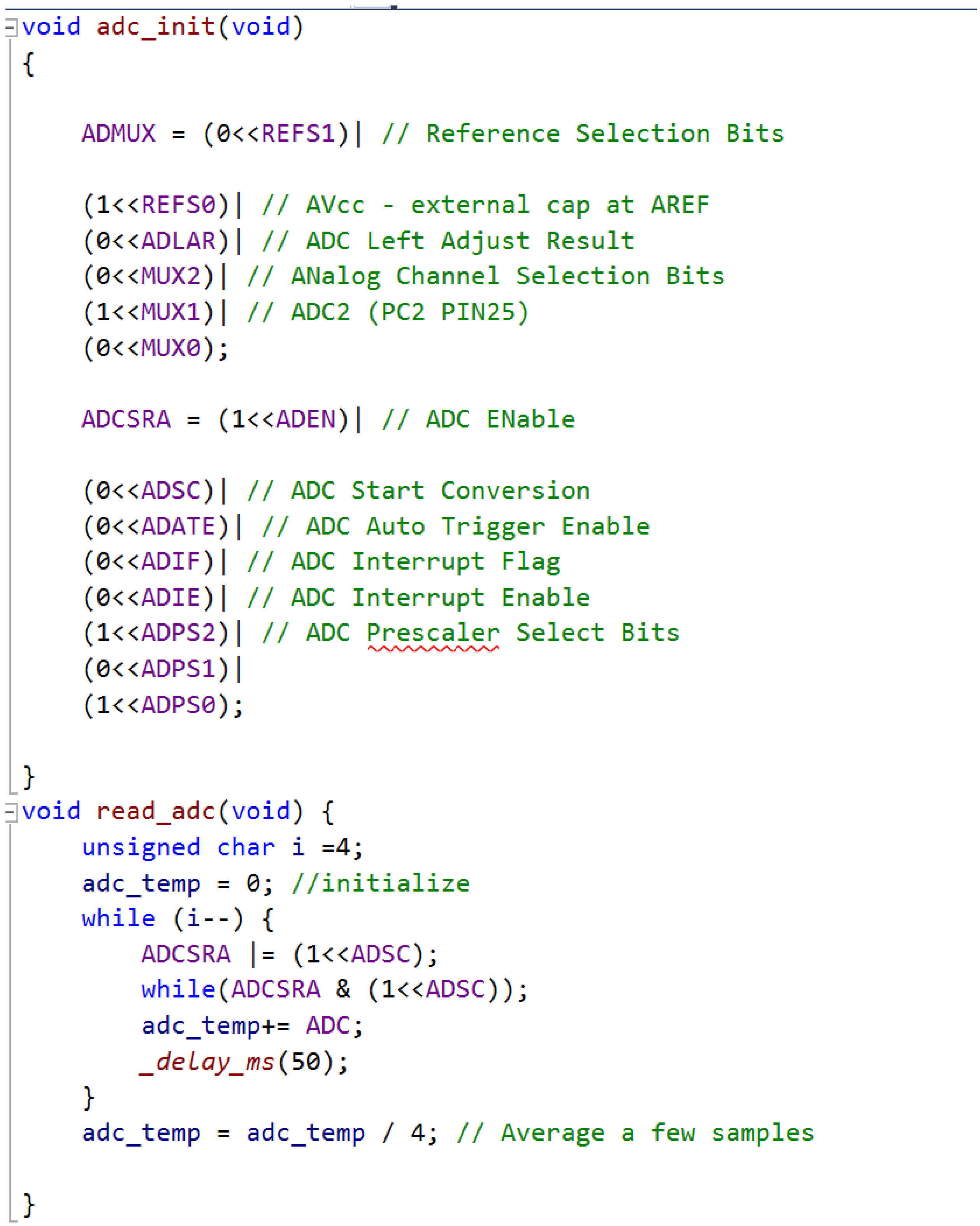
```c
#define F_CPU 16000000UL //16MHz
#define BAUD 9600 //Baud Rate
#define MYUBRR F_CPU/16/BAUD-1 //calculate Baud

#include <avr/io.h>
#include <avr/interrupt.h>
#include <stdbool.h>
#include <string.h>
#include <util/delay.h>
#include "nrf24l01.h"
#include "nrf24l01-mnemonics.h"

nRF24L01 *setup_rf(void);
void process_message(char *message);
inline void prepare_led_pin(void);
inline void set_led_high(void);

inline void set_led_low(void);
volatile bool rf_interrupt = false;


void read_adc(void);  //Read ADC
void adc_init(void);  //initialize ADC
void USART_init( unsigned int ubrr ); //initialize USART
void USART_tx_string(char *data); //Print String USART
volatile unsigned int adc_temp;
char outs[20]; //array
```



```c
int main(void) {
    uint8_t address[5] = { 0x20, 0x30, 0x40, 0x51, 0x61 };
    prepare_led_pin();

        adc_init(); // Initialize the ADC (Analog / Digital Converter)
        USART_init(MYUBRR); // Initialize the USART (RS232 interface)
        USART_tx_string("Connected!\r\n"); // shows theres a connection with USART
        _delay_ms(125); // wait a bit


    sei();
    nRF24L01 *rf = setup_rf();
    nRF24L01_listen(rf, 0, address);
    uint8_t addr[5];
    nRF24L01_read_register(rf, CONFIG, addr, 1);
    while (true) {
        if (rf_interrupt) {
            rf_interrupt = false;
            while (nRF24L01_data_received(rf)) {
                nRF24L01Message msg;
                nRF24L01_read_received_data(rf, &msg);
                process_message((char *)msg.data);
                USART_tx_string(msg.data);
            }
            nRF24L01_listen(rf, 0, address);
        }
    }
    return 0;
}
```



```c
nRF24L01 *setup_rf(void) {
    nRF24L01 *rf = nRF24L01_init();
    rf->ss.port = &PORTB;
    rf->ss.pin = PB2;
    rf->ce.port = &PORTB;
    rf->ce.pin = PB1;
    rf->sck.port = &PORTB;
    rf->sck.pin = PB5;
    rf->mosi.port = &PORTB;
    rf->mosi.pin = PB3;
    rf->miso.port = &PORTB;
    rf->miso.pin = PB4;
    // interrupt on falling edge of INT0 (PD2)
    EICRA |= _BV(ISC01);
    EIMSK |= _BV(INT0);
    nRF24L01_begin(rf);
    return rf;
}

void process_message(char *message) {
    if (strcmp(message, "ON") == 0)
    set_led_high();
    else if (strcmp(message, "OFF") == 0)
    set_led_low();
}
inline void prepare_led_pin(void) {
    DDRB |= _BV(PB0);
    PORTB &= ~_BV(PB0);
}
inline void set_led_high(void) {
    PORTB |= _BV(PB0);
}
inline void set_led_low(void) {
    PORTB &= ~_BV(PB0);
}
```

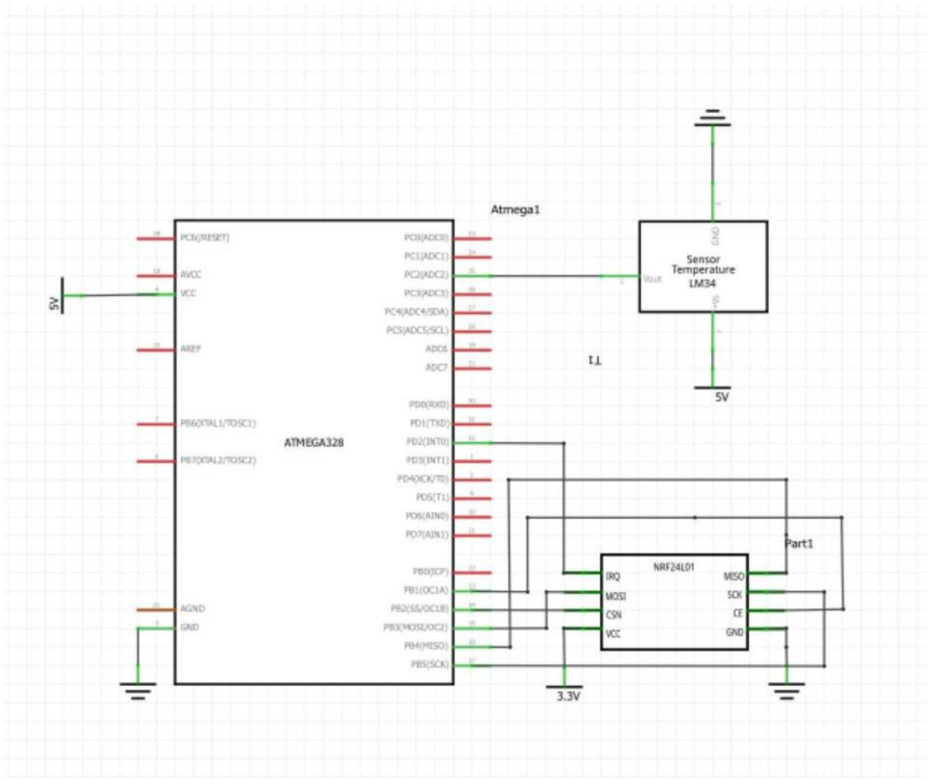


```c
void adc_init(void)
{

    ADMUX = (0<<REFS1)| // Reference Selection Bits

    (1<<REFS0)| // AVcc - external cap at AREF
    (0<<ADLAR)| // ADC Left Adjust Result
    (0<<MUX2)| // ANalog Channel Selection Bits
    (1<<MUX1)| // ADC2 (PC2 PIN25)
    (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);

}
void read_adc(void) {
    unsigned char i =4;
    adc_temp = 0; //initialize
    while (i--) {
        ADCSRA |= (1<<ADSC);
        while(ADCSRA & (1<<ADSC));
        adc_temp+= ADC;
        _delay_ms(50);
    }
    adc_temp = adc_temp / 4; // Average a few samples

}
```

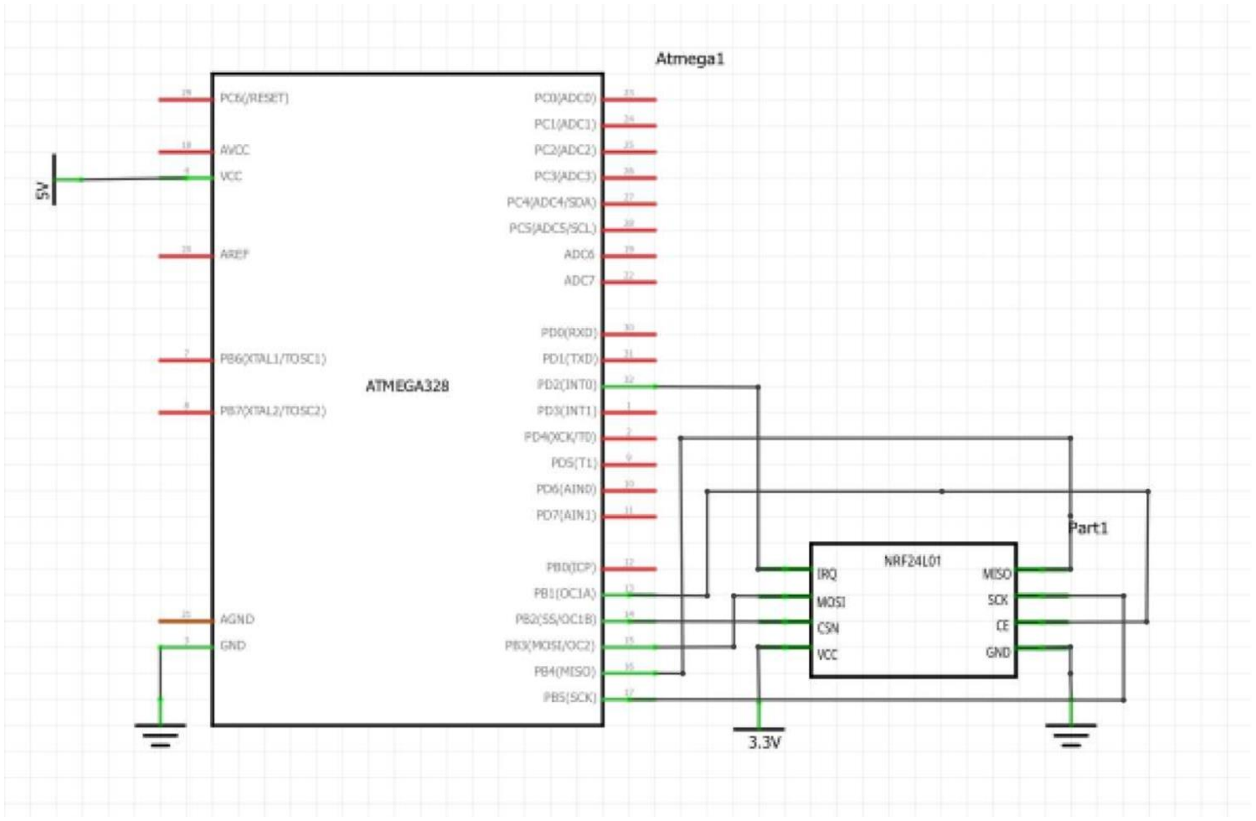
```
/* INIT USART (RS-232) */
void USART_init( unsigned int ubrr ) {
    UBRR0H = (unsigned char)(ubrr>>8);
    UBRR0L = (unsigned char)ubrr;
    UCSR0B = (1 << TXEN0); // Enable receiver, transmitter & RX interrupt
    UCSR0C = (3 << UCSZ00); //asynchronous 8 N 1
}

void USART_tx_string( char *data ) {
    while ((*data != '\0')) {
        while (!(UCSR0A & (1 <<UDRE0)));
        UDR0 = *data;
        data++;
    }
}
// nRF24L01 interrupt
ISR(INT0_vect) {
    rf_interrupt = true;
}
```

4. SCHEMATICS

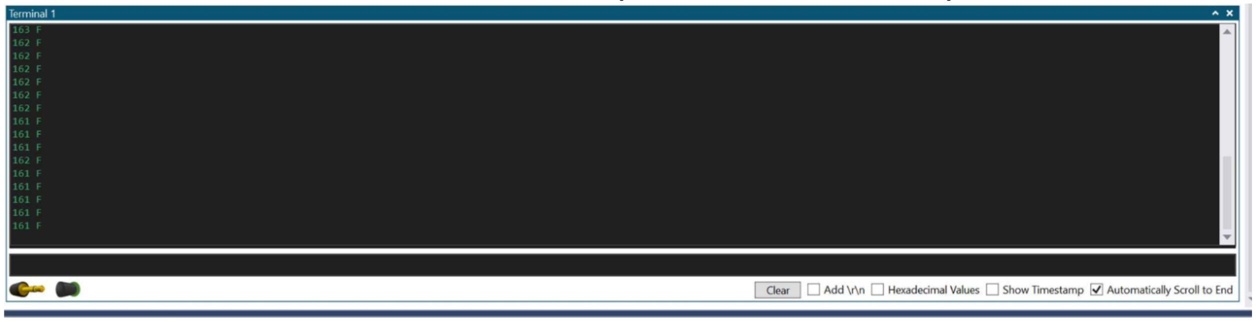
Transmit



Receive



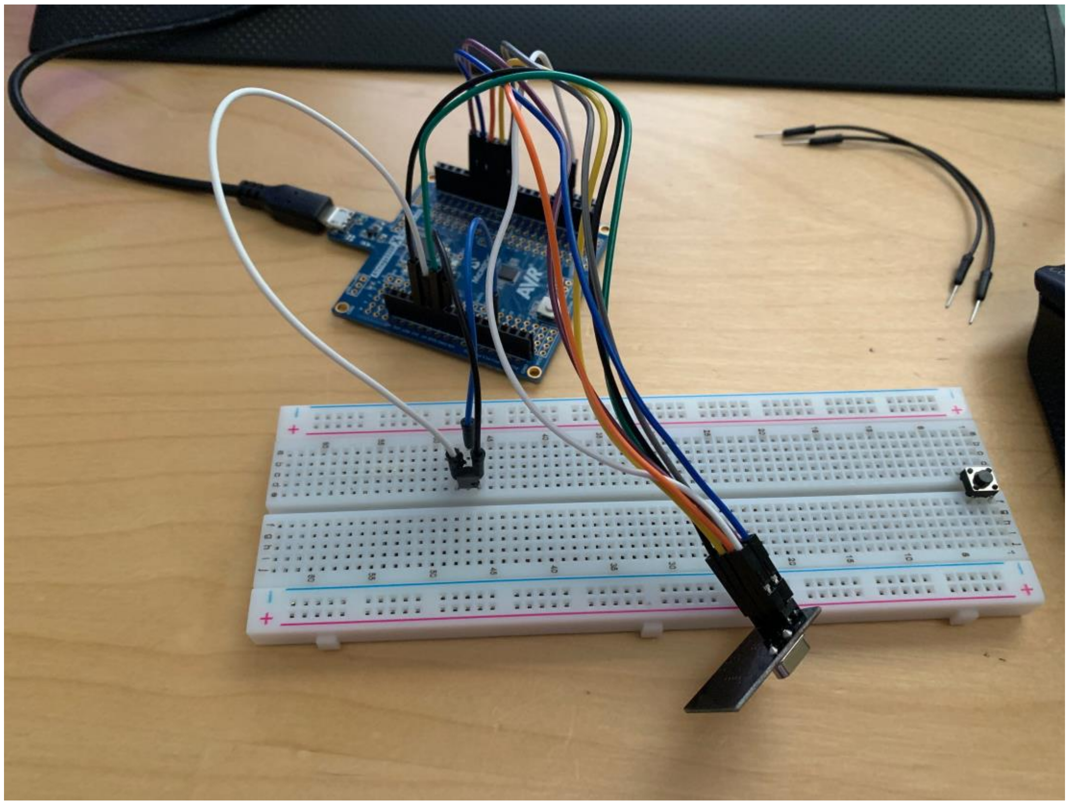
5. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)



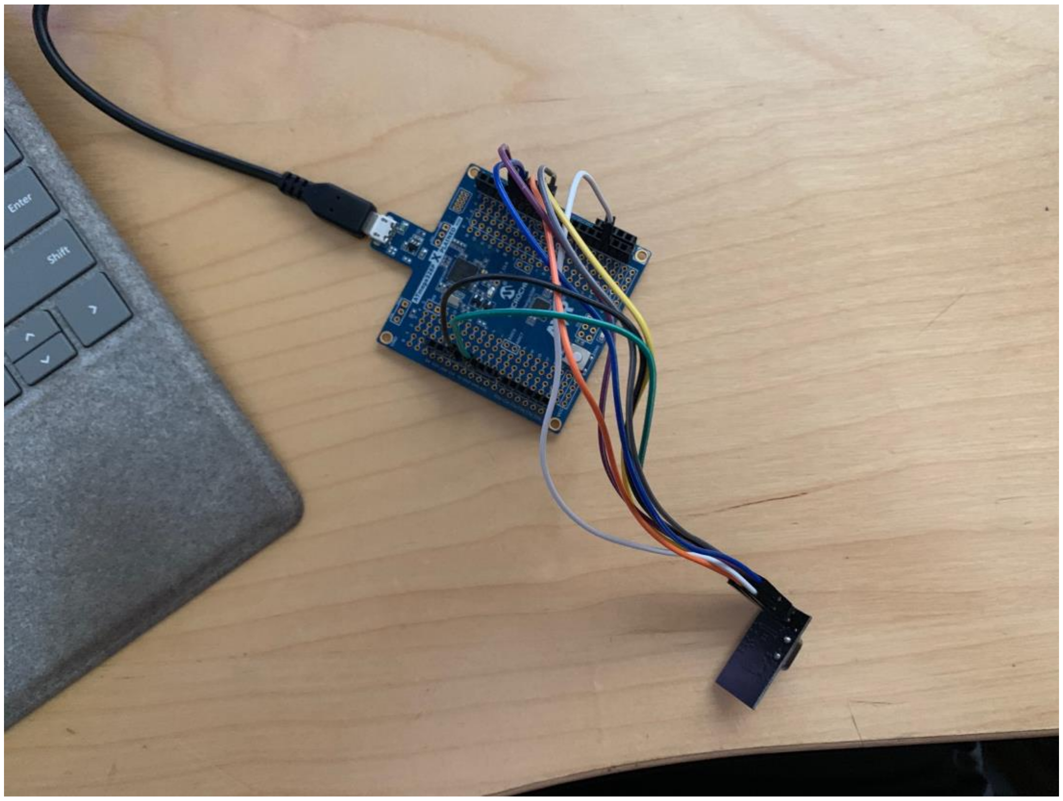
```
163 F
162 F
162 F
162 F
162 F
162 F
162 F
161 F
161 F
161 F
162 F
161 F
161 F
161 F
161 F
161 F
```

6. SCREENSHOT OF EACH DEMO (BOARD SETUP)

Transmit



Receive



7. VIDEO LINKS OF EACH DEMO

8. GITHUB LINK OF THIS DA

https://github.com/regis-shaquille/submissions-SR/tree/master/Design%20Assignments/DA5

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

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