CPE301 – SPRING 2019

Design Assignment 5A

Student Name: Cody Hudson

Student #: 8000603642

Student Email: hudsoc1@unlv.nevada.edu

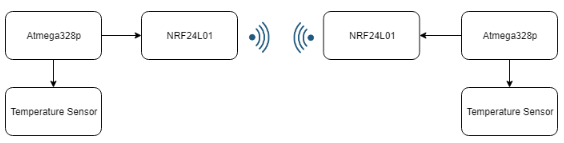
Primary Github address: https://github.com/eed911/class\_proj.git

Directory:

https://github.com/eed911/class\_proj/tree/master/DesignAssignments/DA5A/Project\_5A

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

* Task 1:
  + ATMEGA328p x2
  + NRF24L01 x2
  + Temperature Sensor x2



1. **INITIAL/MODIFIED/DEVELOPED CODE OF TASK 1**

// Set clock frequency

#ifndef *F\_CPU*

#define *F\_CPU* 16000000UL

#endif

#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();

#ifndef BAUD

#define BAUD 9600

#endif

#include "inc/STDIO\_UART.h"

// Include nRF24L01+ library

#include "inc/nrf24l01.h"

#include "inc/nrf24l01-mnemonics.h"

#include "inc/spi.h"

void print\_config(void);

void read\_adc(void); // Function Declarations

void adc\_init(void);

volatile unsigned int adc\_temp;

volatile unsigned int tempF; // Volatile Fahrenheit temperature variable

volatile unsigned int tempC; // Volatile Celcius temperature variable

char outs[20];

// Used in IRQ ISR

volatile bool message\_received = false;

volatile bool status = false;

int main(void)

{

// Set cliche message to send (message cannot exceed 32 characters)

char tx\_message[32]; // Define string array

*strcpy*(tx\_message,"Thanos"); // Copy string into array

// Initialize UART

uart\_init();

adc\_init();

// Initialize nRF24L01+ and print configuration info

nrf24\_init();

print\_config();

// Start listening to incoming messages

nrf24\_start\_listening();

while (1)

{

read\_adc();

adc\_temp = (adc\_temp\*500)/1023;

tempF = adc\_temp + 20;

*snprintf*(outs,sizeof(outs),"%3d\r\n", tempF); // print it

*strcpy*(tx\_message,outs); // Copy string into array

nrf24\_send\_message(tx\_message);

*\_delay\_ms*(1500);

if (message\_received)

{

// Message received, print it

message\_received = false;

*printf*("Received message: %s\n",nrf24\_read\_message());

// Send message as response

*\_delay\_ms*(500);

status = nrf24\_send\_message(tx\_message);

if (status == true) *printf*("Message sent successfully\n");

}

}

}

// Interrupt on IRQ pin

ISR(INT0\_vect)

{

message\_received = true;

}

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%02X\n",data);

nrf24\_read(EN\_AA,&data,1);

*printf*("EN\_AA 0x%02X\n",data);

nrf24\_read(EN\_RXADDR,&data,1);

*printf*("EN\_RXADDR 0x%02X\n",data);

nrf24\_read(SETUP\_RETR,&data,1);

*printf*("SETUP\_RETR 0x%02X\n",data);

nrf24\_read(RF\_CH,&data,1);

*printf*("RF\_CH 0x%02X\n",data);

nrf24\_read(RF\_SETUP,&data,1);

*printf*("RF\_SETUP 0x%02X\n",data);

nrf24\_read(STATUS,&data,1);

*printf*("STATUS 0x%02X\n",data);

nrf24\_read(FEATURE,&data,1);

*printf*("FEATURE 0x%02X\n",data);

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

}

/\* INIT ADC \*/

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

(1<<MUX2)| // Analog Channel Selection Bits

(0<<MUX1)| // ADC4 (PC4 PIN27)

(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);

}

/\* READ ADC PINS \*/

void read\_adc(void)

{

unsigned char i = 4;

adc\_temp = 0;

while (i--)

{

ADCSRA |= (1<<ADSC);

while(ADCSRA & (1<<ADSC));

adc\_temp+= ADC;

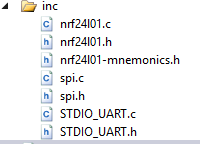
*\_delay\_ms*(50);

}

adc\_temp = adc\_temp / 4; // Average a few samples

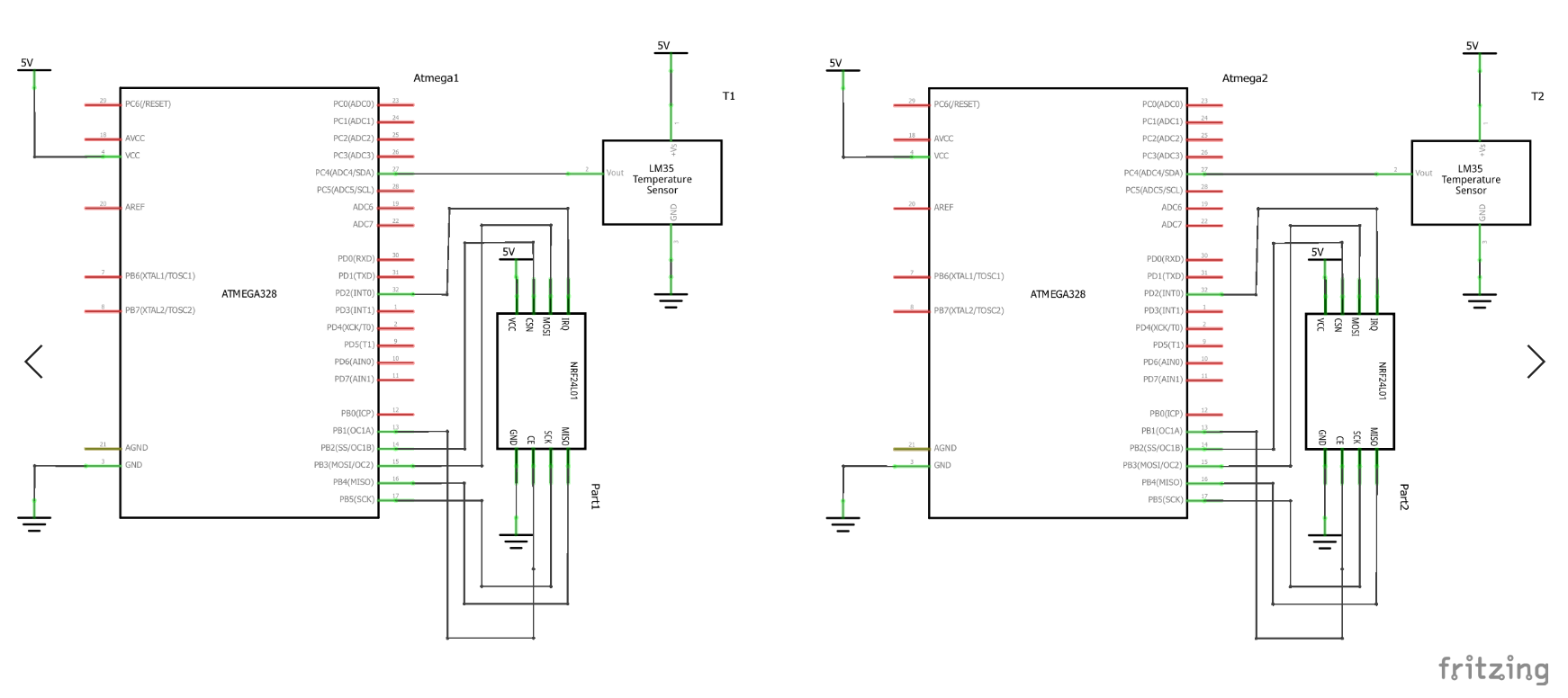
}

Libraries need to be added:



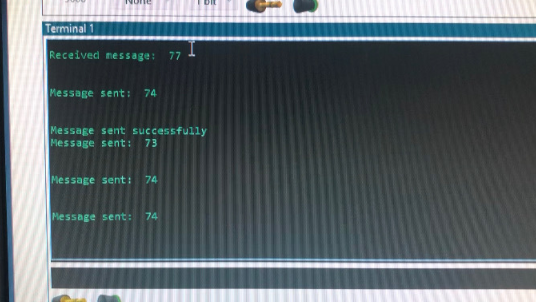
1. **SCHEMATICS**

Task 1:



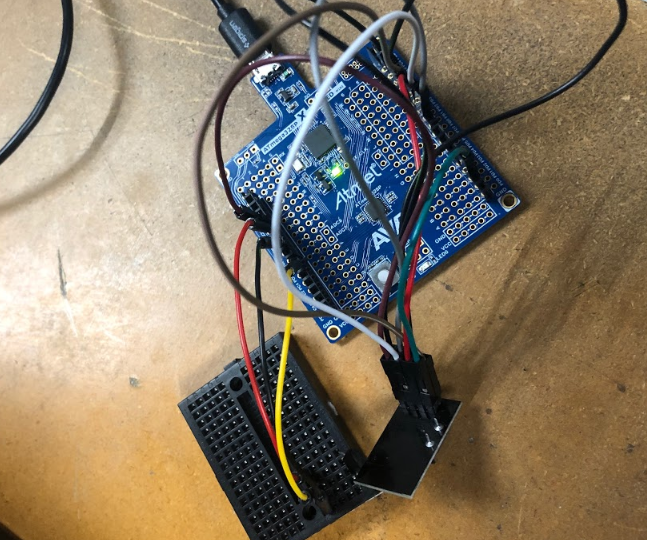
1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**

Task1:



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

Task 1:



1. **VIDEO LINKS OF EACH DEMO**

Task 1:

<https://www.youtube.com/watch?v=uNAHuSdLw2s>

1. **GITHUB LINK OF THIS DA**

https://github.com/eed911/class\_proj.git

**Student Academic Misconduct Policy**

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

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

Cody Hudson