**CPE301 – SPRING 2019**

Design Assignment 5

Student Name: Prachi Patel

Student #: 500280222

Student Email: patelp3@unlv.nevada.edu

Primary Github address: <https://github.com/prachi173/>

Directory: <https://github.com/prachi173/da_sp18/>

Submit the following for all Labs:

* In the document, for each task submit the modified or included code (only) with highlights and justifications of the modifications. Also, include the comments.
* 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.
* If multiple asm/c files or other libraries are used, create a folder LabXX-TYY and place these files inside the folder.
* 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).

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

Atmega328-p Xplained Mini

FTDI Basic UART

LM25 Temperature Sensor

NRF24L01+ RF

A close up of text on a white background

Description automatically generated

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

//Prachi Patel

//DA5

#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>

// UART Baud setup and include file

#ifndef BAUD

#define BAUD 9600

#endif

#include "lib\STDIO\_UART.c"

// Include nRF24L01+ library

#include "lib\nrf24l01.c"

#include "lib\nrf24l01-mnemonics.h"

#include "lib\spi.c"

void print\_config(void); // Function Declarations

void read\_adc(void);

void adc\_init(void);

volatile unsigned int temp;

char outs[20];

// IRQ ISR

volatile bool message\_received = false;

volatile bool status = false;

int main(void)

{

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

uart\_init(); // Initialize UART

adc\_init(); //initialize adc

nrf24\_init(); // Initialize nRF24L01+

print\_config(); //print configuration info

nrf24\_start\_listening(); // Start listening to incoming messages

while (1)

{

read\_adc();//read ADC

temp = (((5000/1024)\*temp)/10) + 2;

temp = temp \* 1.8 + 32; //conversion of temp

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

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

nrf24\_send\_message(tx\_message);

*\_delay\_ms*(1500);

if (message\_received)

{

//if message is received, print it

message\_received = false;

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

// and then send message response

*\_delay\_ms*(500);

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

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

}

}

}

// Interrupt using 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%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)

{

//setup 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)

(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

void read\_adc(void)

{

unsigned char i = 4;

temp = 0;

while (i--)

{

ADCSRA |= (1<<ADSC);

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

temp+= ADC;

*\_delay\_ms*(50);

}

temp = temp / 4; // get average of 4 samples

}

* **SCHEMATICS**

A close up of text on a white background

Description automatically generated

* **SCREENSHOT OF EACH DEMO (ATMEL SETUP)**



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

**A circuit board

Description automatically generated**

* **VIDEO LINKS OF EACH DEMO**

<https://youtu.be/mCbO2l-bkUQ>

* **GITHUB LINK OF THIS DA**

<https://github.com/prachi173/da_sp18/tree/master/Design%20Assignments/DA5>

**Student Academic Misconduct Policy**

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

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

Prachi Patel