CPE301 – SPRING 2019

Design Assignment 5

Student Name: Victor Martinez & Itzel Becerril

Student #: 5003131674

Student Email: [martiv6@unlv.nevada.edu](mailto:martiv6@unlv.nevada.edu)

Primary Github address: https://github.com/martiv6

Directory: https://github.com/martiv6/submissions\_da/tree/master/DesignAssignment/DA5

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**

NRF24L01+

328P

Mini

POWERSUPPLY

NRF24L01+

328P

Mini

POWERSUPPLY

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

/\*

\* DA5.c

\* Author : becerri2 and martiv6

\*/

#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;

char outs[20];

// Used in IRQ ISR

volatile bool message\_received = false;

volatile bool status = false;

int main(void)

{

char tx\_message[32]; // Define string 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();//read ADC

adc\_temp = (adc\_temp\*500)/1023 + 20; //conversion of temp

*snprintf*(outs,sizeof(outs),"%3d\r\n", adc\_temp); // 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;

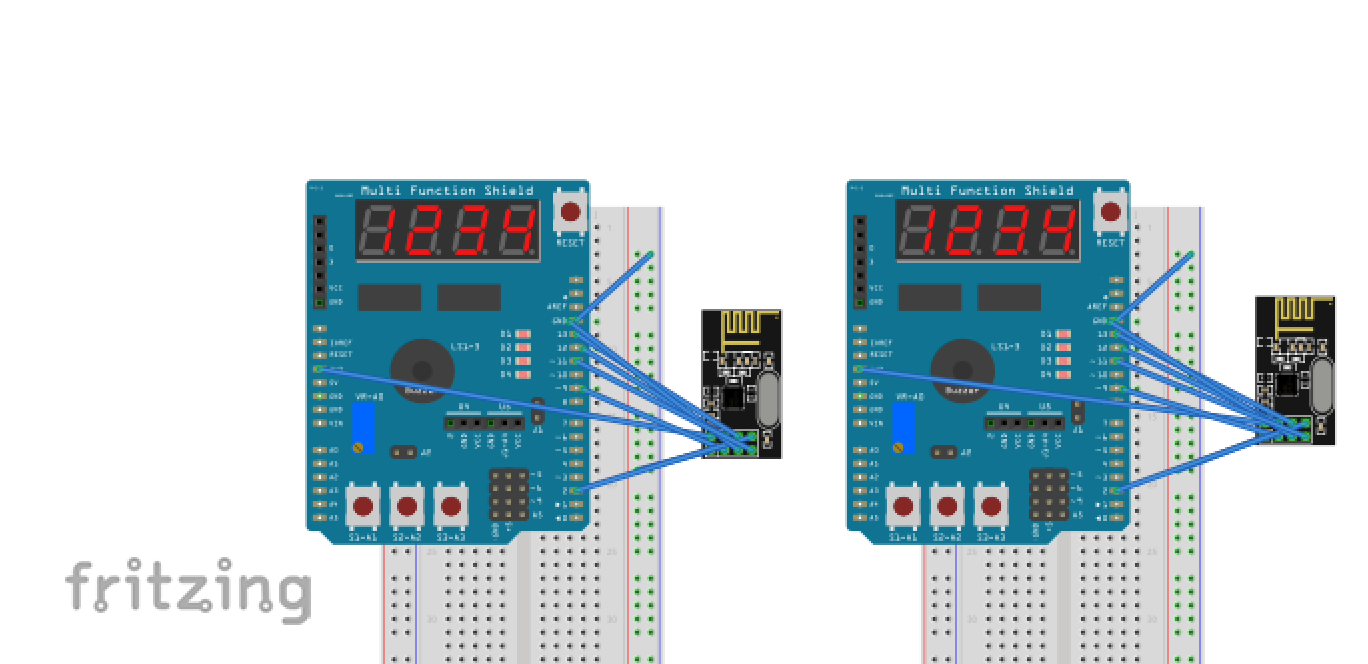
*\_delay\_ms*(50);

}

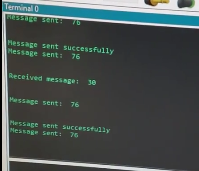
adc\_temp = adc\_temp / 4; // Average a few samples

}

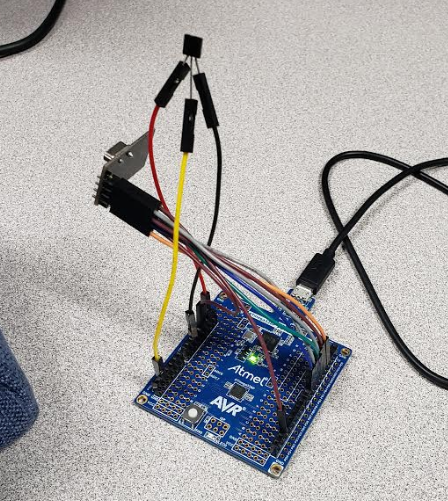
1. **SCHEMATICS**



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



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



1. **VIDEO LINKS OF EACH DEMO**

<https://www.youtube.com/watch?v=XYlI3REwjdo>

1. **GITHUB LINK OF THIS DA**

https://github.com/martiv6/submissions\_da/tree/master/DesignAssignment/DA5

**Student Academic Misconduct Policy**

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

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

VICTOR MARTINEZ