CPE301 – SPRING 2019

Design Assignment 5

Student Name: Robert Sander

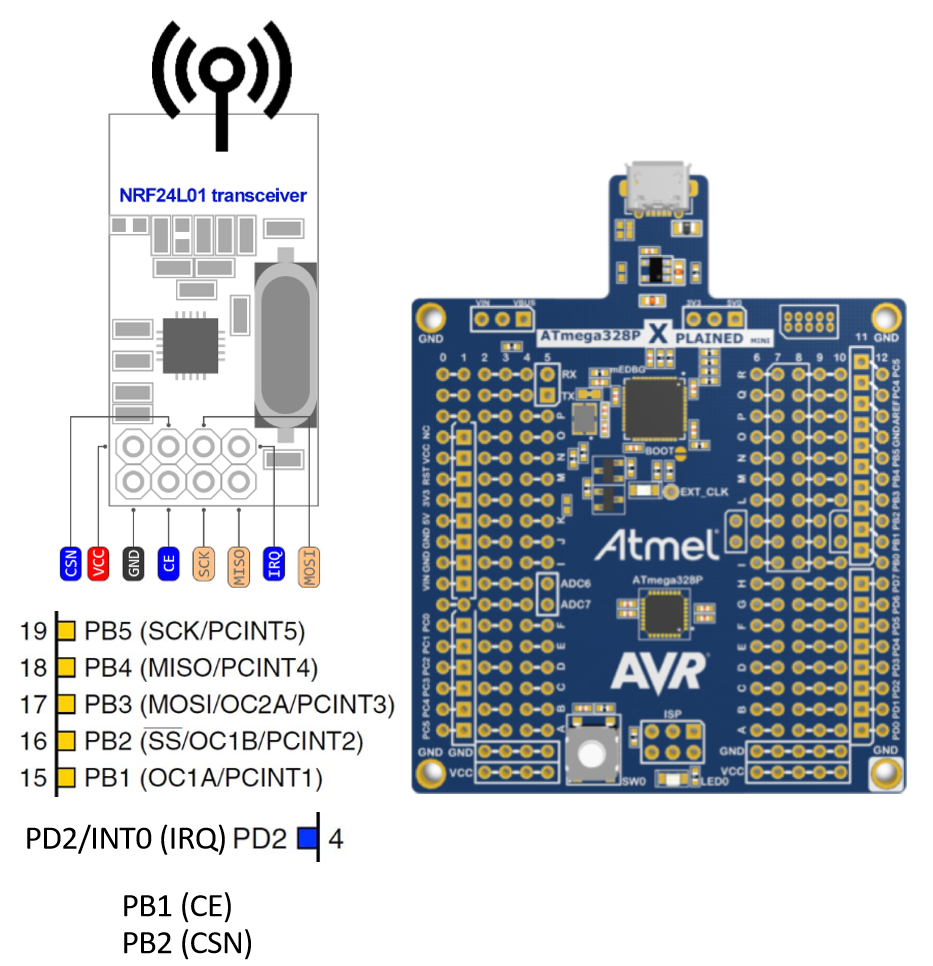
Student #: 5002102412

Student Email: sander1@unlv.nevada.edu

Primary Github address: <https://github.com/sanderUNLV/submission_DA.git>

Youtube link: <https://youtu.be/pN7AYRIvz6g>

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



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

// MIT License Geo Vanni Portillo Rocky Gonzalez

//

// Copyright (c) 2018 Helvijs Adams

//

// Permission is hereby granted, free of charge, to any person obtaining a copy

// of this software and associated documentation files (the "Software"), to deal

// in the Software without restriction, including without limitation the rights

// to use, copy, modify, merge, publish, distribute, sublicense, and/or sell

// copies of the Software, and to permit persons to whom the Software is

// furnished to do so, subject to the following conditions:

//

// The above copyright notice and this permission notice shall be included in all

// copies or substantial portions of the Software.

//

// THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR

// IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,

// FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE

// AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER

// LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,

// OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE

// SOFTWARE.

//

// Software was tested on ATmega328P and ATmega328PB (PB needs few changes in SPI)

// RF module software was tested on - cheap nRF24L01+ from China

// All the relevant settings are defined in nrf24l01.c file

// Some features will be added later, at this moment it is bare minimum to send/receive

//

// 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 "STDIO\_UART.h"

// Include nRF24L01+ library

#include "nrf24l01.h"

#include "nrf24l01-mnemonics.h"

#include "spi.h"

void print\_config(void);

void adc\_init(void);

void read\_adc(void);

volatile double adc\_temp = 0;

// Used in IRQ ISR

volatile bool message\_received = false;

volatile bool status = false;

int main(void)

{

adc\_init(); // Initialize the ADC

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

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

*strcpy*(tx\_message,"Hello World!"); // Copy string into array

// Initialize UART

uart\_init();

// Initialize nRF24L01+ and print configuration info

nrf24\_init();

print\_config();

*\_delay\_ms*(1000);

// Start listening to incoming messages

nrf24\_start\_listening();

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

//if (status == true) printf("Connection established\n");

while (1)

{

read\_adc();

*snprintf*(tx\_message, sizeof(tx\_message), "%3f\r\n", adc\_temp); // print it

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

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");

//USART\_putadc();

}

}

}

// 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%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 and enable ADC

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

(1<<REFS0)| // Internal 1.1

(0<<ADLAR)| // ADC Left Adjust Result

(0<<MUX3)| // Analog Channel Selection Bits

(1<<MUX2)|

(0<<MUX1)| // ADC5

(0<<MUX0);

ADCSRA = (1<<ADEN)| // ADC ENable

(1<<ADSC)| // ADC Start Conversion

(1<<ADATE)| // ADC Auto Trigger Enable

(0<<ADIF)| // ADC Interrupt Flag

(0<<ADIE)| // ADC Interrupt Enable

(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0); // ADC Prescaler - 128

}

void read\_adc(void)

{

unsigned char i = 4;

while (i--)

{

//ADCSRA |= (1<<ADSC);

while((ADCSRA&(1<<ADIF))==0);//wait for conversion to finish

PORTB ^= (1 << 2);

adc\_temp += ADC;

*\_delay\_ms*(100);

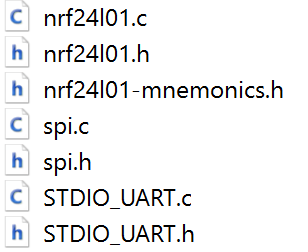
//ADCSRA |= (1<<ADIF);

}

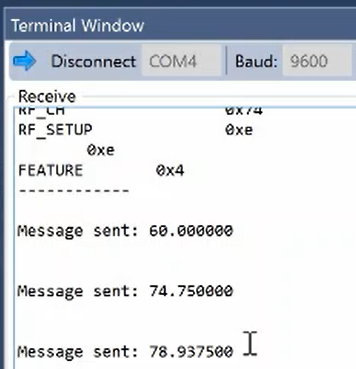
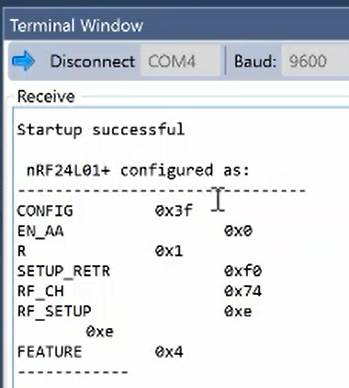
adc\_temp = adc\_temp / 4; // Average a few samples

}

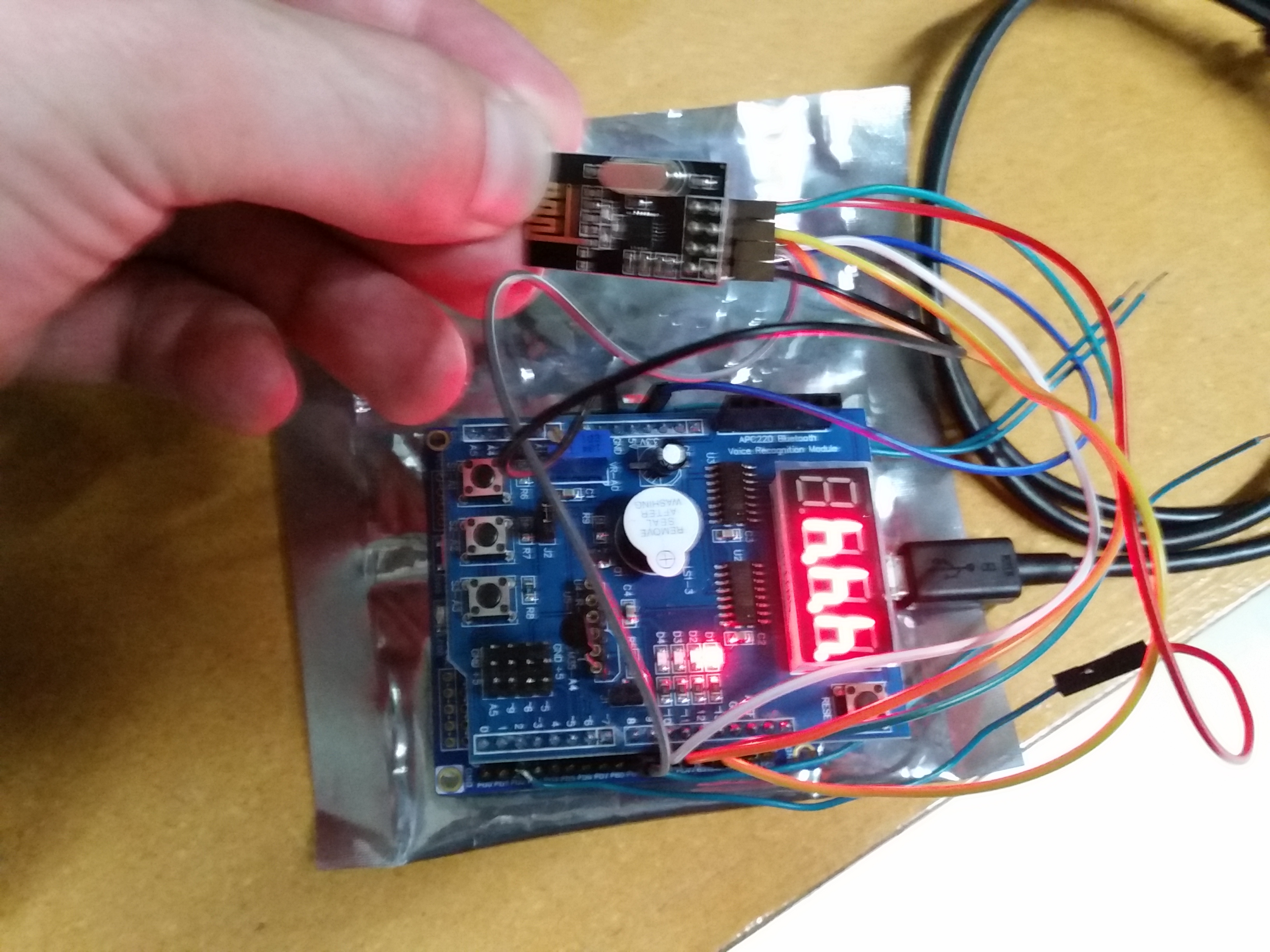
Included files:



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



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



1. **VIDEO LINKS OF EACH DEMO**

<https://youtu.be/pN7AYRIvz6g>

1. **GITHUB LINK OF THIS DA**

<https://github.com/sanderUNLV/submission_DA.git>

**Student Academic Misconduct Policy**

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

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

-Robert Sander