CPE301 – SPRING 2019

Design Assignment 5

Student Name: Geovanni Portillo

Student #: 8000603824

Student Email: [portig1@unlv.nevada.edu](mailto:portig1@unlv.nevada.edu)

Primary Github address: <https://github.com/portig1/submissions_E>

Directory: submissions\_E/DA/LAB5/

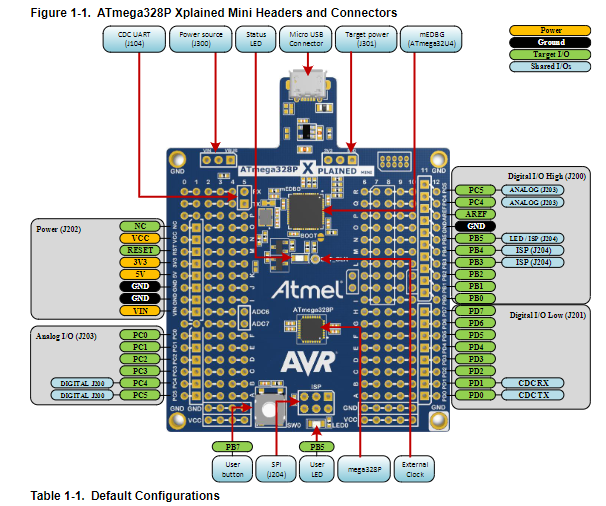
Submit the following for all Labs:

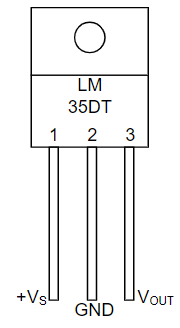
1. In the document, for each task submit the modified or included code (only) with highlights and justifications of the modifications. Also, include the comments.
2. Use the previously create a Github repository with a random name (no CPE/301, Lastname, Firstname). Place all labs under the root folder 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**

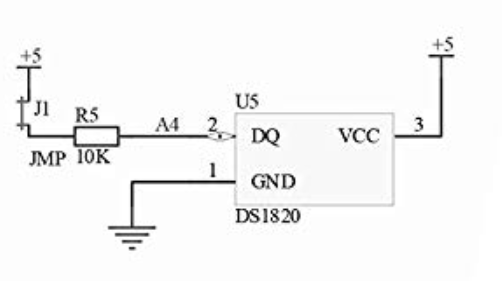
Atmel Studio 7

ATmega328PB Xplained mini

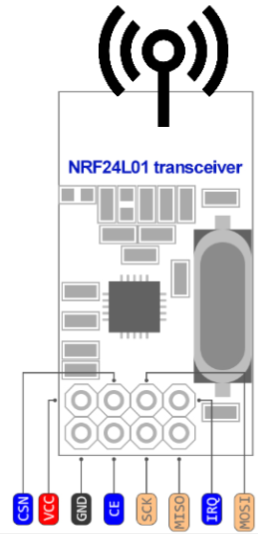




LM35 Schematic



Arduino Shield U5 Schematic



NRF2424L01 Schematic

1. **MODIFIED CODE OF TASK 1**

/\*

\* Task\_1.c

\*

\* Created: 4/22/2019 2:28:58 PM

\* Author : gausp

\*/

//Worked with Robert Sander and Rocky Gonzalez trying to send and

//receive data using the ATmega328 and NRF24L01 module.

//The code remains mostly unchanged from the files at https://github.com/venki666/cpe301Demo/tree/master/Libraries/NRF24L01\_LIB/NRF24L01\_BM\_RX

//The spi and nrf24l01 codes required changes to the registers such as SPSR being changed to SPSR0 to use the 328PB

//We noticed that the configuration would be properly setup on the 328P but for the 328PB the configuration would be done wrong

//For example, RF\_CH is supposed to be 0x74 but for the PB it gets set to 0x30.

//We tested this by only swapping out the 328P with the 328PB (only one of use had a 328P), the only changes for the code

//were to revert the register name changes previously mentioned.

//The datasheet for the 328PB showed that the SPI pins for the 328P pins such as MOSI

//just become known as MOSI0 but are otherwise the same so we ruled out that as being

//the issue. We also checked the connections several times.

// MIT License

//

// 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);

// Used in IRQ ISR

volatile bool message\_received = false;

volatile bool status = false;

//setup ADC

void adc\_init();

int adc\_getValue();

int main(void)

{

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

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

*strcpy*(tx\_message,"Hello from the Other Side!"); // Copy string into array

// Initialize UART

uart\_init();

// Initialize ADC and test

adc\_init(); //Set up ADC

*printf*("Testing ADC, Temperature is: %d Celsius\n\n",adc\_getValue());

// Initialize nRF24L01+ and print configuration info

nrf24\_init();

print\_config();

// Start listening to incoming messages

nrf24\_start\_listening();

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

if (status == true) *printf*("Connection Established\n");

while (1)

{

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);

//Perform ADC and store in tx\_message

*snprintf*(tx\_message, sizeof(tx\_message), "Temp value from Geo: %d Celsius", adc\_getValue());

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

if (status == true) *printf*("Message Transmitted\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%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()

{

/\*\* 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)| // CLK/32

(1<<ADPS0);

}

int adc\_getValue()

{

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

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

ADCSRA |= (1<<ADIF);

int adcValue = ADCL;

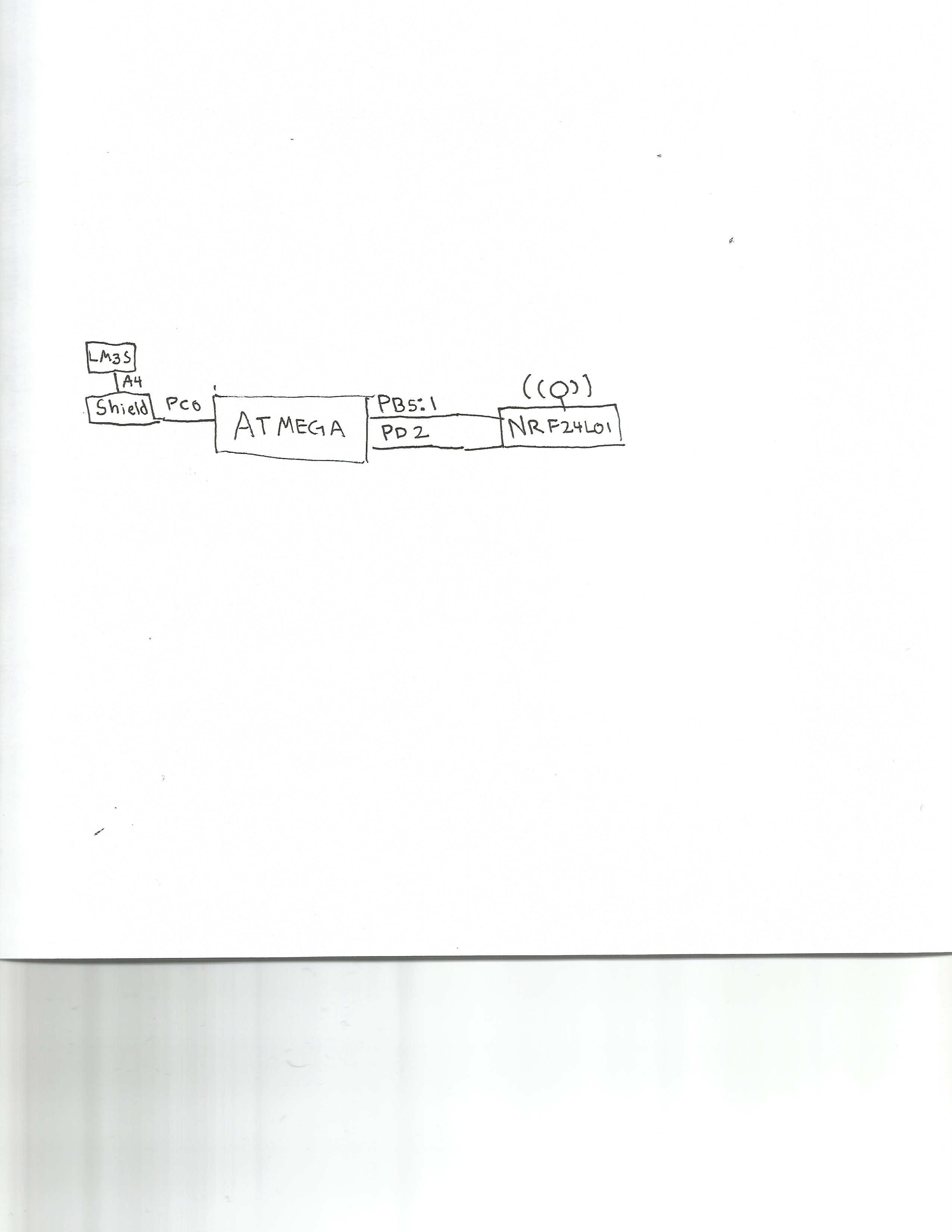
adcValue = adcValue | (ADCH<<8);

adcValue = (adcValue/1024.0) \* 5000/10; //finish properly formatting tempC value

return adcValue;

}

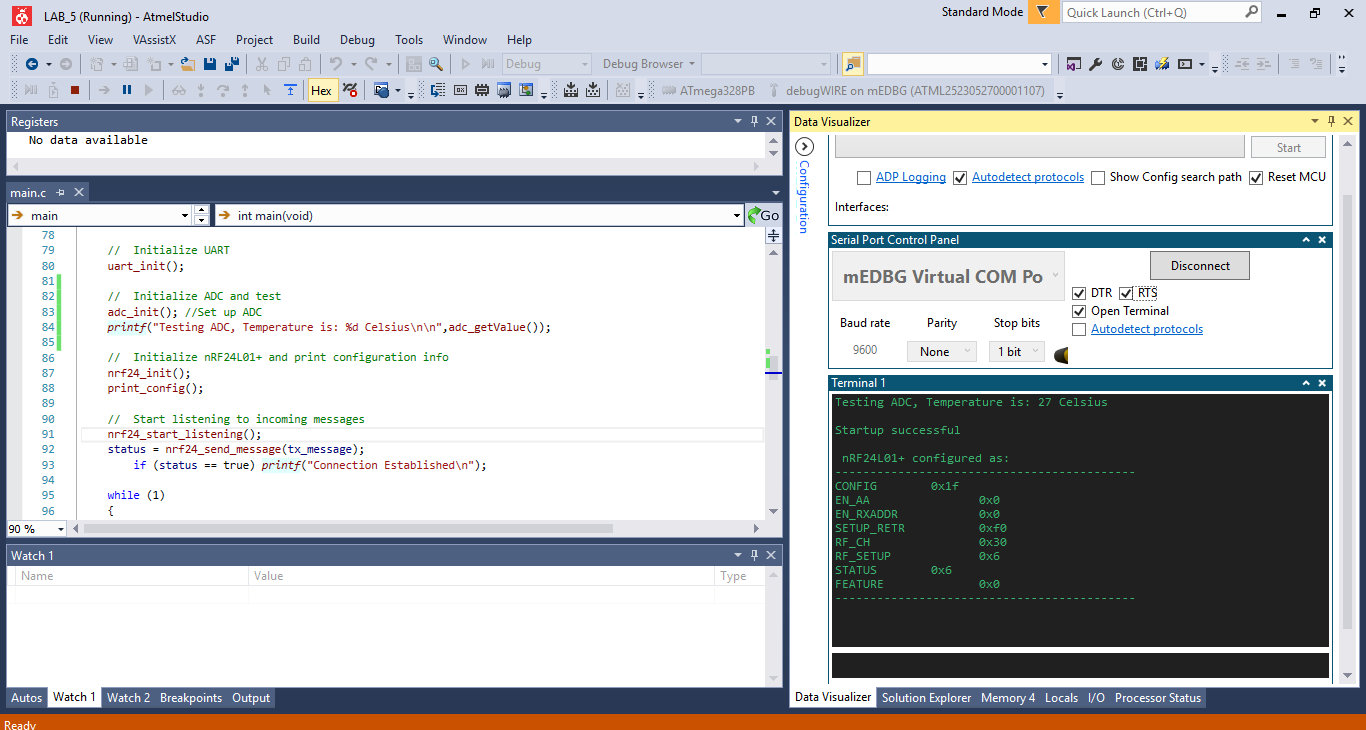
1. **SCHEMATICS**



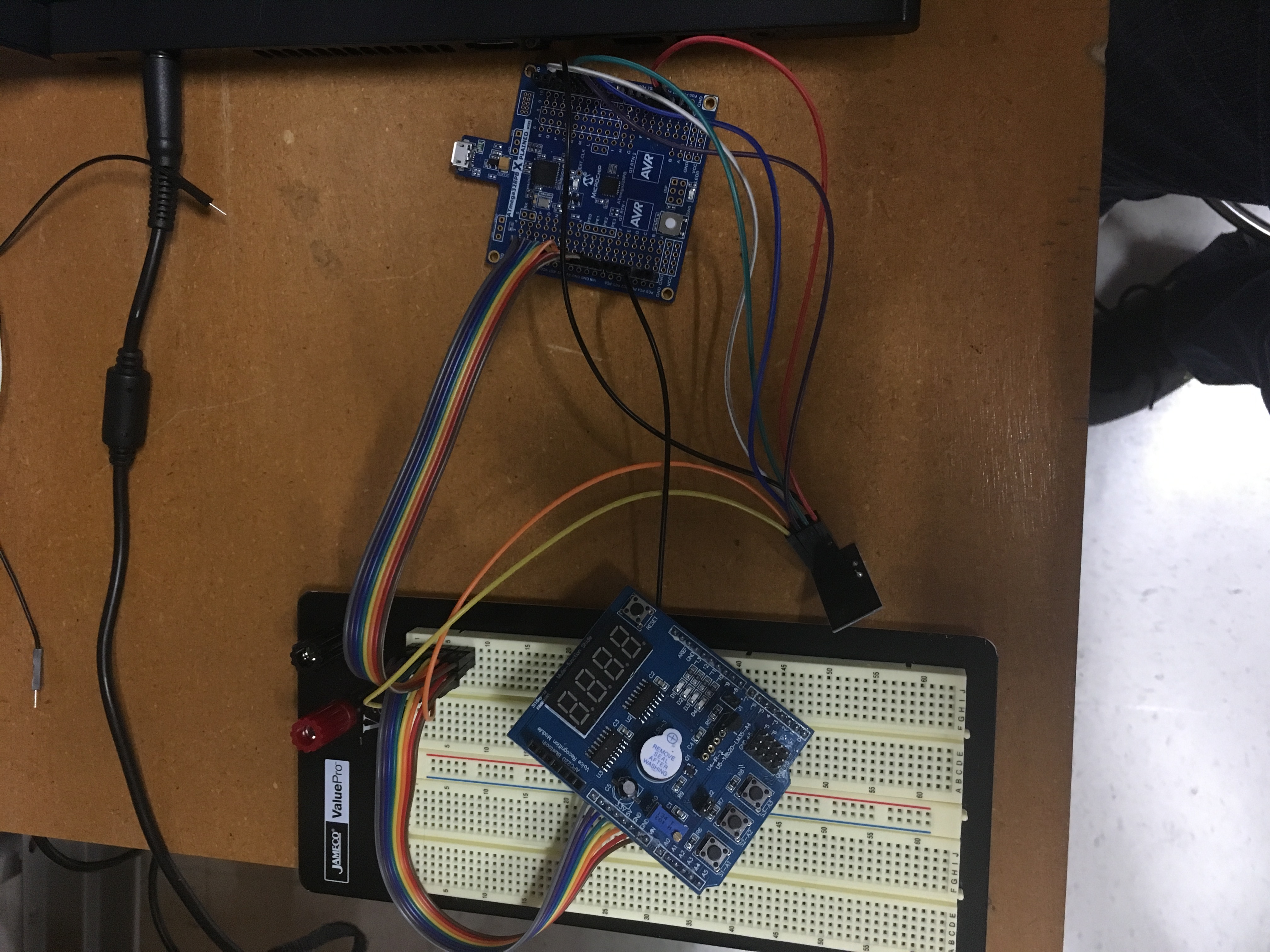
Task 1 Schematic

1. **SCREENSHOTS OF EACH TASK OUTPUT**

**Task 1:**



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



1. **GITHUB LINK OF THIS DA**

https://github.com/portig1/submissions\_E/tree/master/DA/LAB5

**Student Academic Misconduct Policy**

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

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

Geovanni Portillo