CPE301 – SPRING 2019

Design Assignment DA6

Student Name: Daniel Senda

Student #: 5002362633

Student Email: sendad1@unlv.nevada.edu

Primary Github address: https://github.com/dsenda/Smiles

Directory: DA6

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**

*List of Components used:*

Atmel Studio 7

ATmega328P Xplained Mini

MPU6050 (Gyroscope, Accelerometer, Temperature) Sensor

FTDI Basic chip

Breadboard

Jumper wires

*Block diagram about the transmission connections:*

ATmega328

MPU6050 Sensor

ATmega328p

FTDI chip

1. **INITIAL/MODIFIED CODE FROM TASK DA6**

Not applicable, there is no modified code.

1. **DEVELOPED MODIFIED CODE OF TASK DA6**

// c\_code\_MPU6050\_gyro\_accel.c

// Daniel Senda

// Makes use of the MPU6050 (Gyroscope, Accelerometer, Temperature) Sensor.

#define F\_CPU 16000000UL // Defines CPU clock frequency.

// Included libraries:

#include <avr/io.h>

#include <util/delay.h>

#include <inttypes.h>

#include <stdlib.h>

#include <stdio.h>

#include "inc\MPU6050\_Res\_DefineFile.h"

#include "inc\I2C\_MasterFileH.h"

#include "inc\USART\_RS232File\_H.h"

// Variables:

char buffer[20], float\_[10];

float Accel\_x,Accel\_y,Accel\_z;

float Gyro\_x,Gyro\_y,Gyro\_z;

// Functions:

void MPU6050\_Init(void);

void values\_in(void);

void MPU\_Start\_Loc(void);

void value\_calcs(void);

void values2usart(void);

int main()

{

I2C\_Init(); // Calls the I2C\_Init function.

MPU6050\_Init(); // Initializes MPU6050 sensor.

USART\_Init(9600); // Initializes USART.

while(1)

{

values\_in(); // Reads values in from sensor.

value\_calcs(); // Calculates sensor values.

values2usart(); // Sends calculated values to USART.

*\_delay\_ms*(1000);

}

}

void MPU6050\_Init() // Gyro initialization function.

{

*\_delay\_ms*(150); // Power up time >100ms.

I2C\_Start\_Wait(0xD0); // Starts at device that will be written to address.

I2C\_Write(SMPLRT\_DIV); // Writes to sample rate register.

I2C\_Write(0x07); // Sets 1KHz sample rate.

I2C\_Stop();

I2C\_Start\_Wait(0xD0);

I2C\_Write(PWR\_MGMT\_1); // Writes to power management register.

I2C\_Write(0x01); // X axis gyroscope reference frequency.

I2C\_Stop();

I2C\_Start\_Wait(0xD0);

I2C\_Write(CONFIG); // Writes to configuration register.

I2C\_Write(0x00); // Fs = 8KHz.

I2C\_Stop();

I2C\_Start\_Wait(0xD0);

I2C\_Write(GYRO\_CONFIG); // Writes to Gyroscope configuration register.

I2C\_Write(0x18); // Full scale range +/- 2000 degree/C

I2C\_Stop();

I2C\_Start\_Wait(0xD0);

I2C\_Write(INT\_ENABLE); // Writes to interrupt enable register.

I2C\_Write(0x01);

I2C\_Stop();

}

void values\_in() // Reads values from sensor.

{

MPU\_Start\_Loc();

Accel\_x = (((int)I2C\_Read\_Ack()<<8) | (int)I2C\_Read\_Ack());

Accel\_y = (((int)I2C\_Read\_Ack()<<8) | (int)I2C\_Read\_Ack());

Accel\_z = (((int)I2C\_Read\_Ack()<<8) | (int)I2C\_Read\_Ack());

Gyro\_x = (((int)I2C\_Read\_Ack()<<8) | (int)I2C\_Read\_Ack());

Gyro\_y = (((int)I2C\_Read\_Ack()<<8) | (int)I2C\_Read\_Ack());

Gyro\_z = (((int)I2C\_Read\_Ack()<<8) | (int)I2C\_Read\_Nack());

I2C\_Stop();

}

void MPU\_Start\_Loc()

{

I2C\_Start\_Wait(0xD0); // I2C starts with device write address.

I2C\_Write(ACCEL\_XOUT\_H); // Writes start location address from where to read.

I2C\_Repeated\_Start(0xD1); // I2C starts with device read address.

}

void value\_calcs()

{

// Calculates accelerometer values into G force units.

Accel\_x = Accel\_x/16384.0;

Accel\_y = Accel\_y/16384.0;

Accel\_z = Accel\_z/16384.0;

// Calculates gyroscope values into degrees/second units.

Gyro\_x = Gyro\_x/16.4;

Gyro\_y = Gyro\_y/16.4;

Gyro\_z = Gyro\_z/16.4;

}

void values2usart()

{

// Sends values to USART.

*dtostrf*( Accel\_x, 3, 2, float\_ );

*sprintf*(buffer," Accel\_x = %s g\t",float\_);

USART\_SendString(buffer);

*dtostrf*( Accel\_y, 3, 2, float\_ );

*sprintf*(buffer," Accel\_y = %s g\t",float\_);

USART\_SendString(buffer);

*dtostrf*( Accel\_z, 3, 2, float\_ );

*sprintf*(buffer," Accel\_z = %s g\t",float\_);

USART\_SendString(buffer);

*dtostrf*( Gyro\_x, 3, 2, float\_ );

*sprintf*(buffer," Gyro\_x = %s%c/s\t",float\_,0xF8);

USART\_SendString(buffer);

*dtostrf*( Gyro\_y, 3, 2, float\_ );

*sprintf*(buffer," Gyro\_y = %s%c/s\t",float\_,0xF8);

USART\_SendString(buffer);

*dtostrf*( Gyro\_z, 3, 2, float\_ );

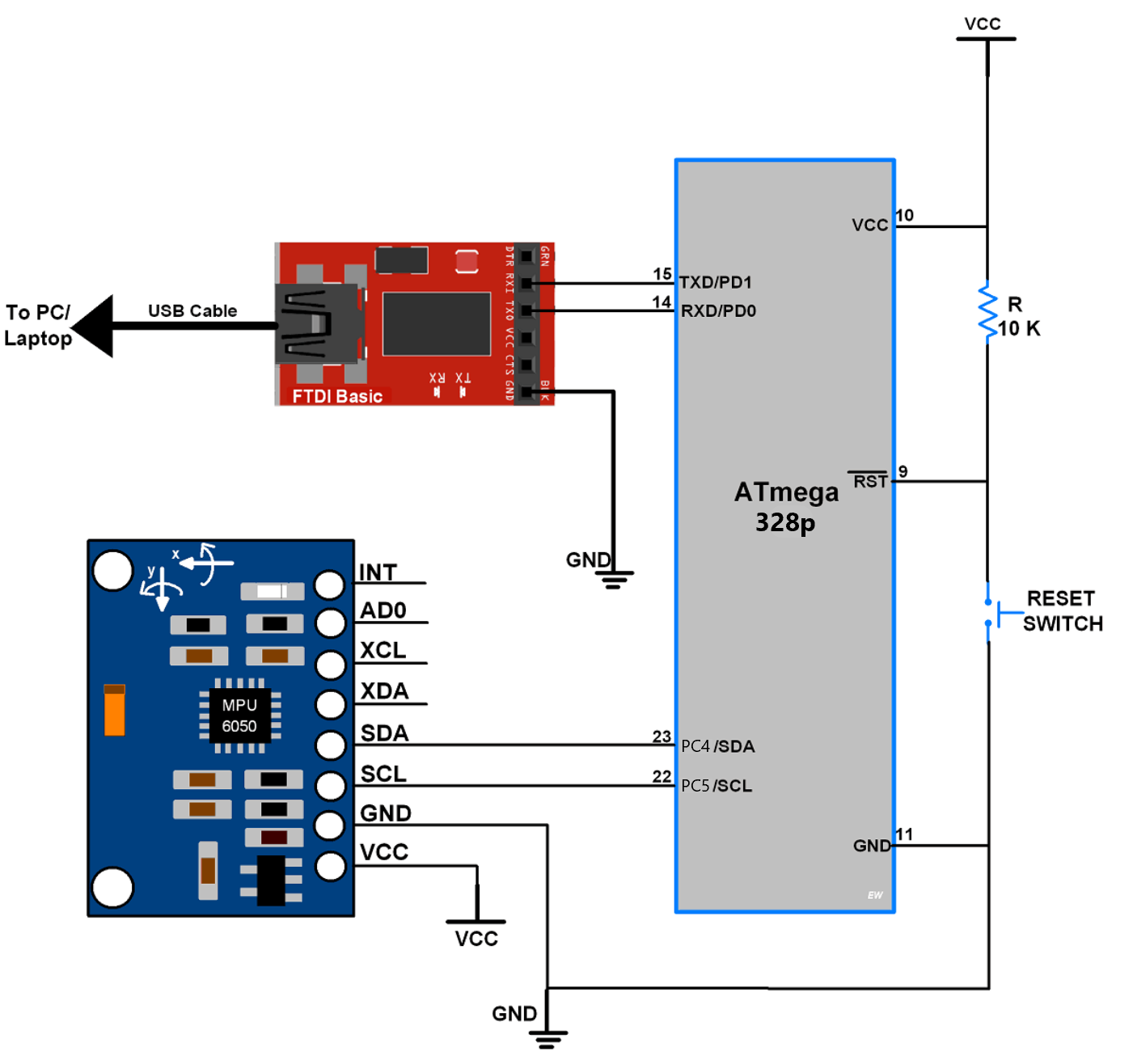
*sprintf*(buffer," Gyro\_z = %s%c/s\r\n",float\_,0xF8);

USART\_SendString(buffer);

}

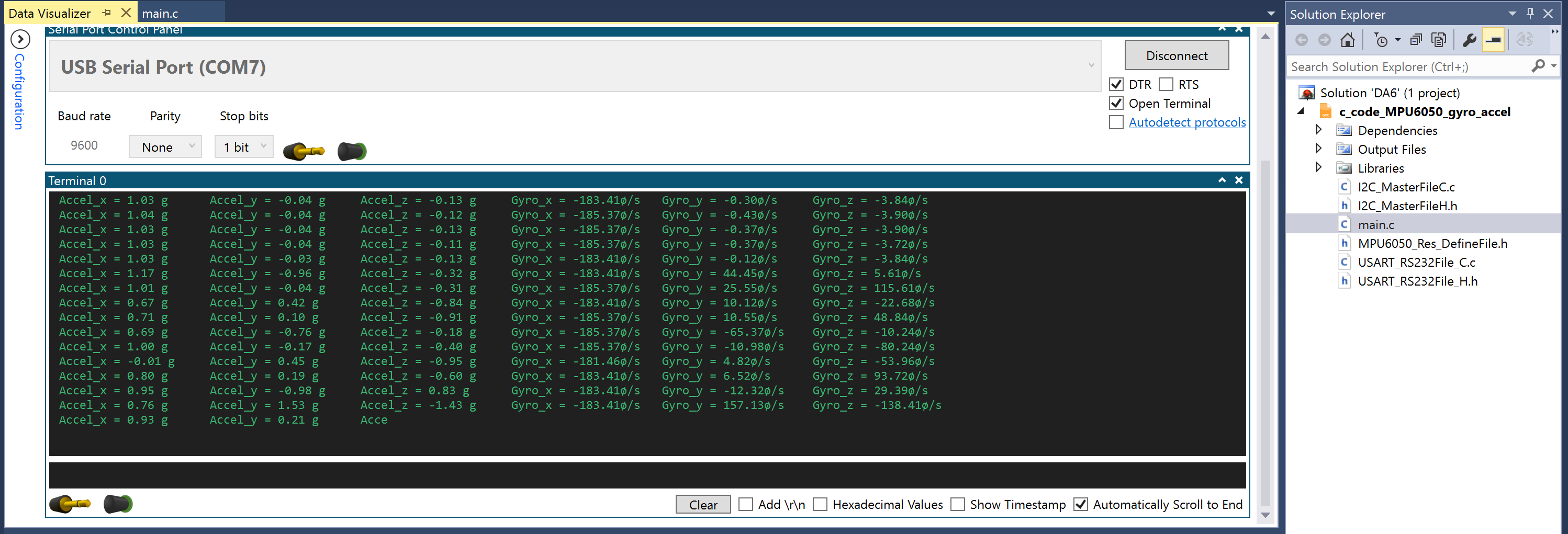
1. **SCHEMATICS**

The following schematic describes the connections that are made relatively well.



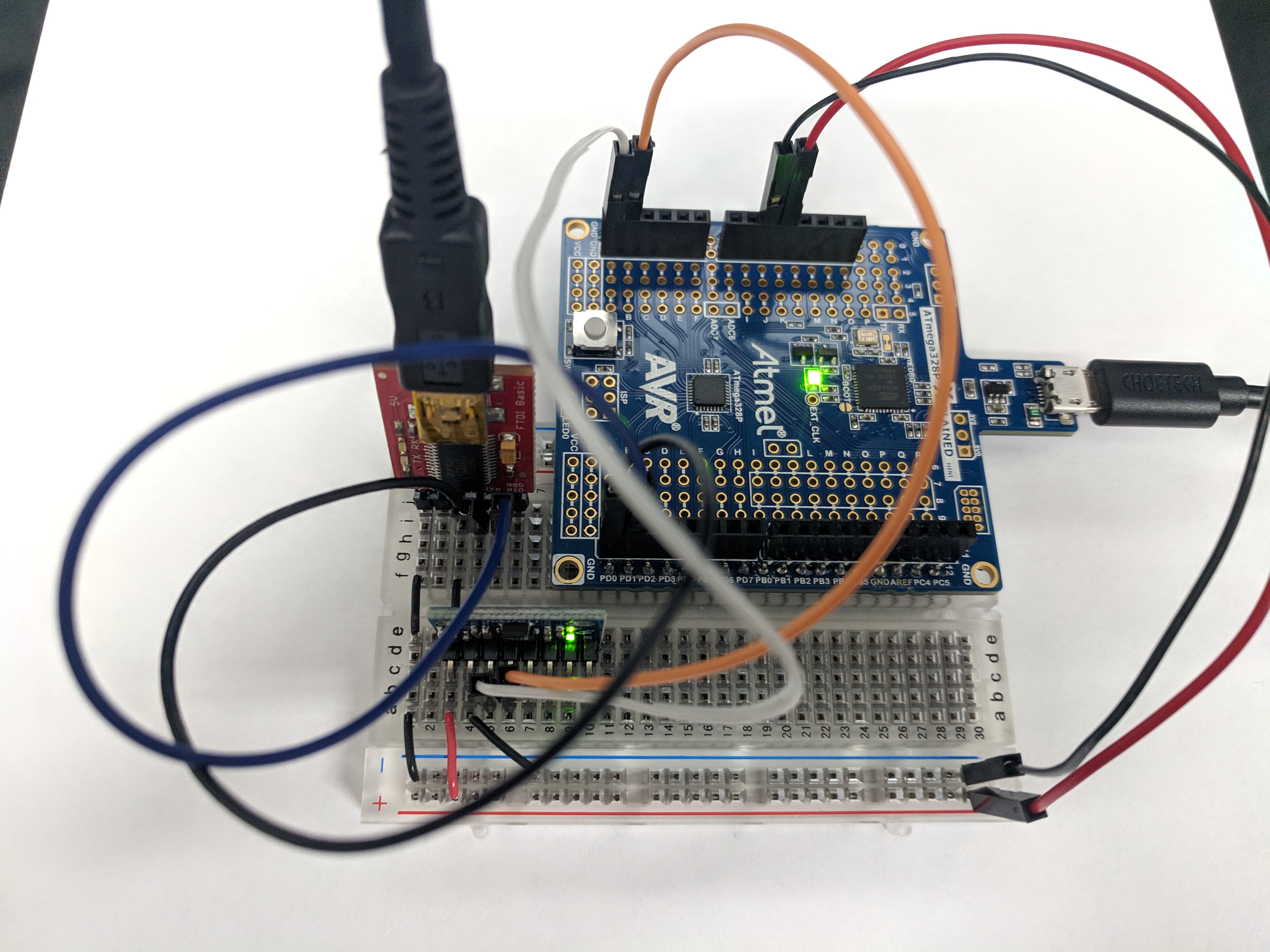
1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**

The following screen shot shows that the student successfully set up the sensor.



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

The following is a picture of the board setup:



1. **VIDEO LINKS OF EACH DEMO**

https://youtu.be/LO4jahOpe4A

1. **GITHUB LINK OF THIS DA**

<https://github.com/dsenda/Smiles/tree/master/DA6>

**Student Academic Misconduct Policy**

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

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

Daniel Senda