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:

2. INITIAL/MODIFIED CODE FROM TASK DA6

Not applicable, there is no modified code.

3. DEVELOPED MODIFIED CODE OF TASK DA6

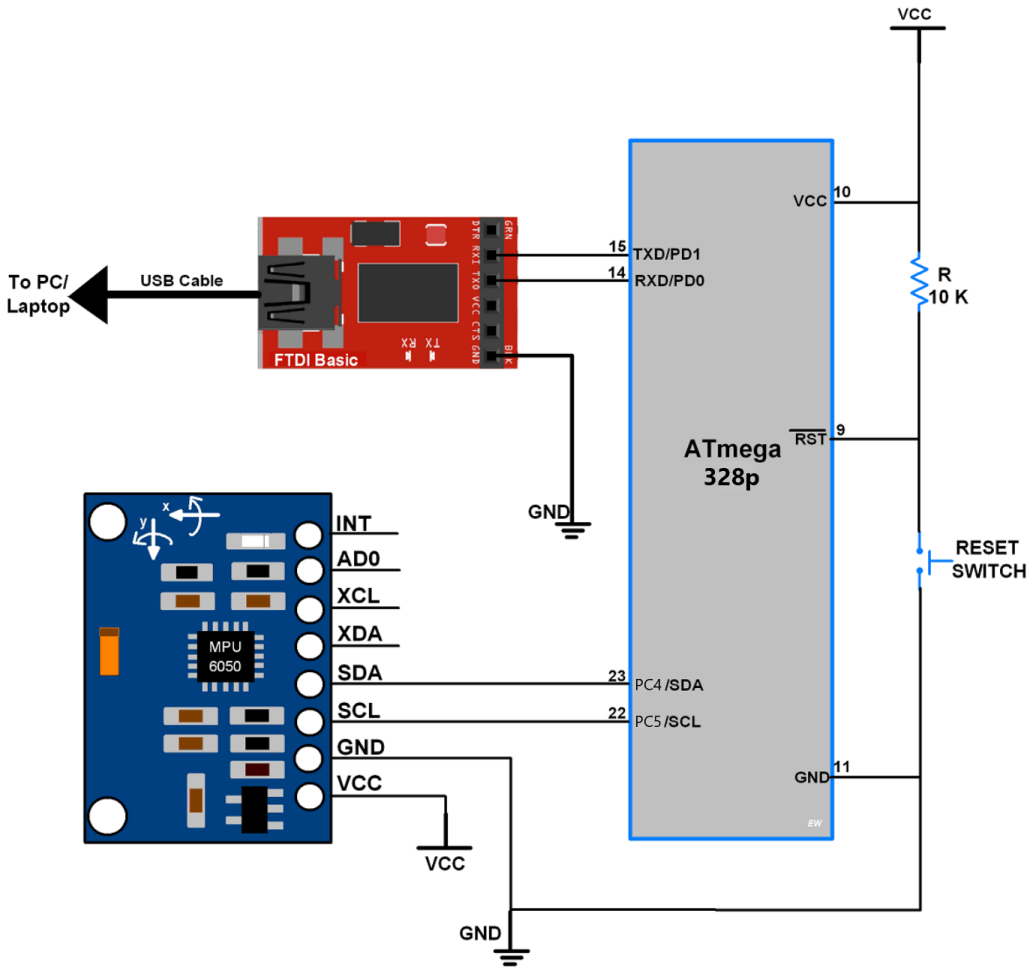
```
// c_code_MPU6050_gyro_accel.c
// Daniel Senda
// Makes use of the MPU6050 (Gyroscope, Accelerometer, Temperature) Sensor.

#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);
}
```

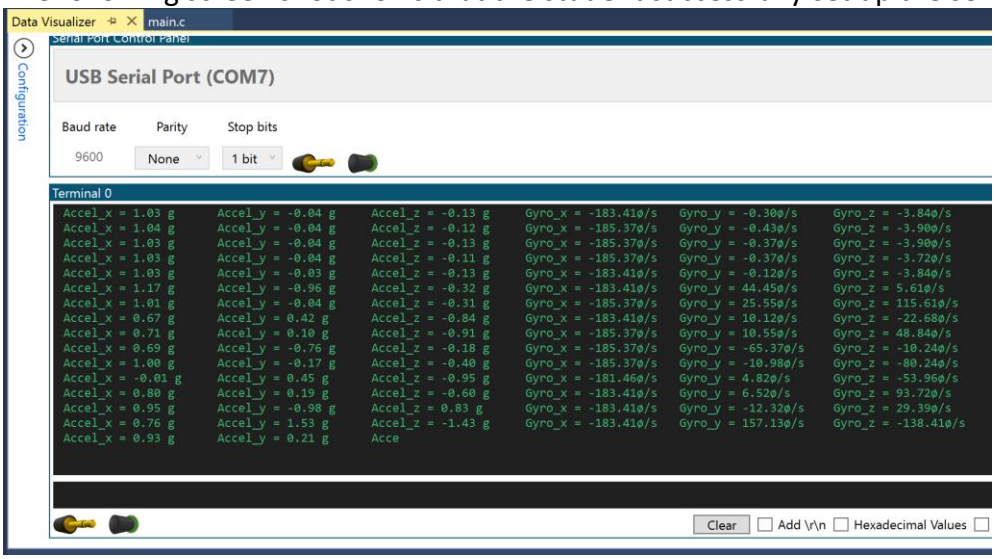
4. SCHEMATICS

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



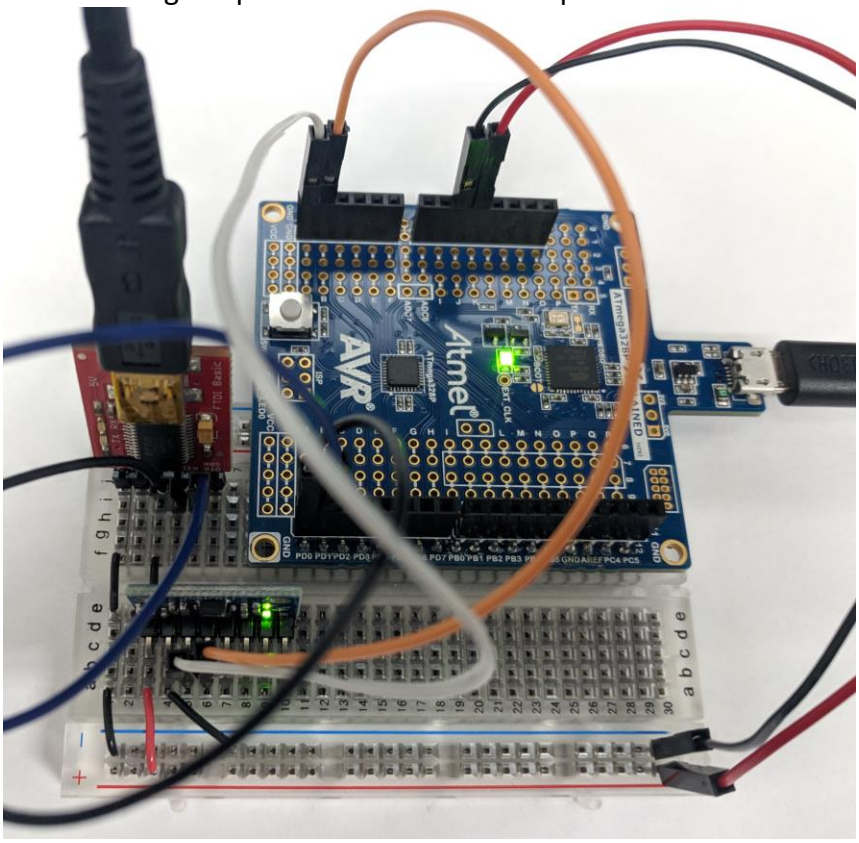
5. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)

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



6. SCREENSHOT OF EACH DEMO (BOARD SETUP)

The following is a picture of the board setup:



7. VIDEO LINKS OF EACH DEMO

https://youtu.be/LO4jahOpe4A

8. 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