CPE301 – SPRING 2019

Design Assignment 6

Student Name: Chris Barr

Student #: 2000682859

Student Email: barrc1@unlv.nevada.edu

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

Directory: https://github.com/BarrChris/submission\_da.git

Submit the following for all Labs:

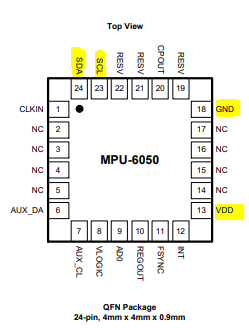
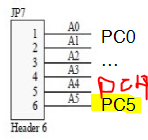
1. In the document, for each task submit the modified or included code (only) with highlights and justifications of the modifications. Also, include the comments.
2. Use the previously create a Github repository with a random name (no CPE/301, Lastname, Firstname). Place all labs under the root folder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

* Atmega328P
* 4 M-F wires
* MPU 6050

Block diagram with pins used in the Atmega328P

SCL was connected to the PC5

SDA was connected to the PC4

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

#ifndef F\_CPU

#define F\_CPU 16000000UL

#endif

#include <avr/io.h>

#include <util/delay.h>

#include <math.h>

#include <stdlib.h> /\* Include standard library file \*/

#include <stdio.h> /\* Include standard library file \*/

#include "MPU6050\_def.h" /\* Include MPU6050 register define file \*/

#include "i2c\_master.h" /\* Include I2C Master header file \*/

#include "uart.h" /\* Include USART header file \*/

#define MPU6050\_WRITE 0xD0

#define MPU6050\_READ 0xD1

float Acc\_x;

float Acc\_z;

float Acc\_y;

float Gyro\_x;

float Gyro\_z;

float Gyro\_y;

void init\_uart(*uint16\_t* baudrate){

*uint16\_t* UBRR\_val = (F\_CPU/16)/(baudrate-1);

UBRR0H = UBRR\_val >> 8;

UBRR0L = UBRR\_val;

UCSR0B |= (1<<TXEN0) | (1<<RXEN0) | (1<<RXCIE0); // UART TX (Transmit - senden) einschalten

UCSR0C |= (1<<USBS0) | (3<<UCSZ00); //Modus Asynchron 8N1 (8 Datenbits, No Parity, 1 Stopbit)

}

void uart\_putc(unsigned char c){

while(!(UCSR0A & (1<<UDRE0))); // wait until sending is possible

UDR0 = c; // output character saved in c

}

void uart\_puts(char \*s){

while(\*s){

uart\_putc(\*s);

s++;

}

}

void init\_MPU6050(void){

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

i2c\_start(MPU6050\_WRITE); // Set Gyroscope Sample Rate = 1 KHz, Accelerometer Sample Rate = 1 KHz (default)

i2c\_write(SMPLRT\_DIV); // Sample Rate is generated by dividing the gyroscope output rate by SMPLRT\_DIV

i2c\_write(0x07); // Gyroscope Output Rate = 8kHz, Sample Rate = Gyroscope Output Rate / (1 + SMPLRT\_DIV)

i2c\_stop();

i2c\_start(MPU6050\_WRITE);

i2c\_write(PWR\_MGMT\_1);

i2c\_write(0x01); // PLL with X axis gyroscope reference

i2c\_stop();

i2c\_start(MPU6050\_WRITE);

i2c\_write(CONFIG); //Frame Synchronization & Digital Low Pass Filter (DLPF) setting

i2c\_write(0x00);

i2c\_stop();

i2c\_start(MPU6050\_WRITE);

i2c\_write(GYRO\_CONFIG); //gyroscopes’ scale range = FS\_SEL selects = 11 = ± 2000 °/s

i2c\_write(0x18); // accelerometer range = ± 2g (default)

i2c\_stop();

i2c\_start(MPU6050\_WRITE);

i2c\_write(INT\_ENABLE); // DATA\_RDY\_EN = 1

i2c\_write(0x01);

i2c\_stop();

}

void getreading(void){

i2c\_start(MPU6050\_WRITE);

i2c\_write(ACCEL\_XOUT\_H); // set pointer

i2c\_write(ACCEL\_YOUT\_H); // set pointer

i2c\_write(ACCEL\_ZOUT\_H); // set pointer

i2c\_stop();

i2c\_start(MPU6050\_READ);

Acc\_x = (((int)i2c\_read\_ack()<<8) | (int)i2c\_read\_ack());

Acc\_y = (((int)i2c\_read\_ack()<<8) | (int)i2c\_read\_ack());

Acc\_z = (((int)i2c\_read\_ack()<<8) | (int)i2c\_read\_ack());

i2c\_stop();

//=========================================================

i2c\_start(MPU6050\_WRITE);

i2c\_write(GYRO\_XOUT\_H);

i2c\_write(GYRO\_YOUT\_H);

i2c\_write(GYRO\_ZOUT\_H);

i2c\_stop();

i2c\_start(MPU6050\_READ);

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();

}

int main(void){

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

float Xa, Ya, Za;

float Xg, Yg, Zg;

init\_uart(9600);

i2c\_init();

init\_MPU6050();

while(1){

getreading();

Xa = Acc\_x/16384.0; /\* Divide raw value by sensitivity scale factor to get real values \*/

Ya = Acc\_y/16384.0;

Za = Acc\_x/16384.0;

Xg = Gyro\_x/16.4; /\* Divide raw value by sensitivity scale factor to get real values \*/

Yg = Gyro\_y/16.4;

Zg = Gyro\_z/16.4;

*dtostrf*( Xa, 3, 2, float\_ ); /\* Take values in buffer to send all parameters over USART \*/

*sprintf*(buffer,"Ax: %s, ",float\_);

USART\_SendString(buffer);

*dtostrf*( Ya, 3, 2, float\_ );

*sprintf*(buffer,"Ay: %s, ",float\_);

USART\_SendString(buffer);

*dtostrf*( Za, 3, 2, float\_ );

*sprintf*(buffer,"Az: %s, \n",float\_);

USART\_SendString(buffer);

*dtostrf*( Xg, 3, 2, float\_ );

*sprintf*(buffer,"Gx: %s, ",float\_);

USART\_SendString(buffer);

*dtostrf*( Yg, 3, 2, float\_ );

*sprintf*(buffer,"Gy: %s, ",float\_);

USART\_SendString(buffer);

*dtostrf*( Zg, 3, 2, float\_ );

*sprintf*(buffer,"Gz: %s, \n\n",float\_);

USART\_SendString(buffer);

*\_delay\_ms*(1000);

}

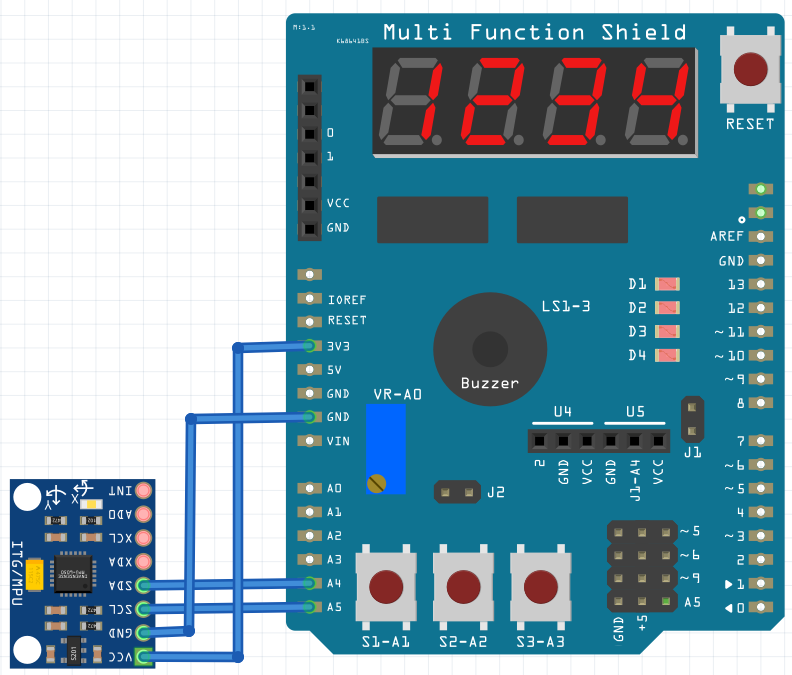
return 0;

}

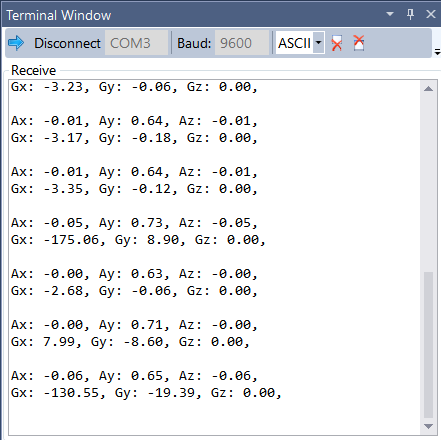
1. **DEVELOPED MODIFIED CODE OF TASK 2/A from TASK 1/A**

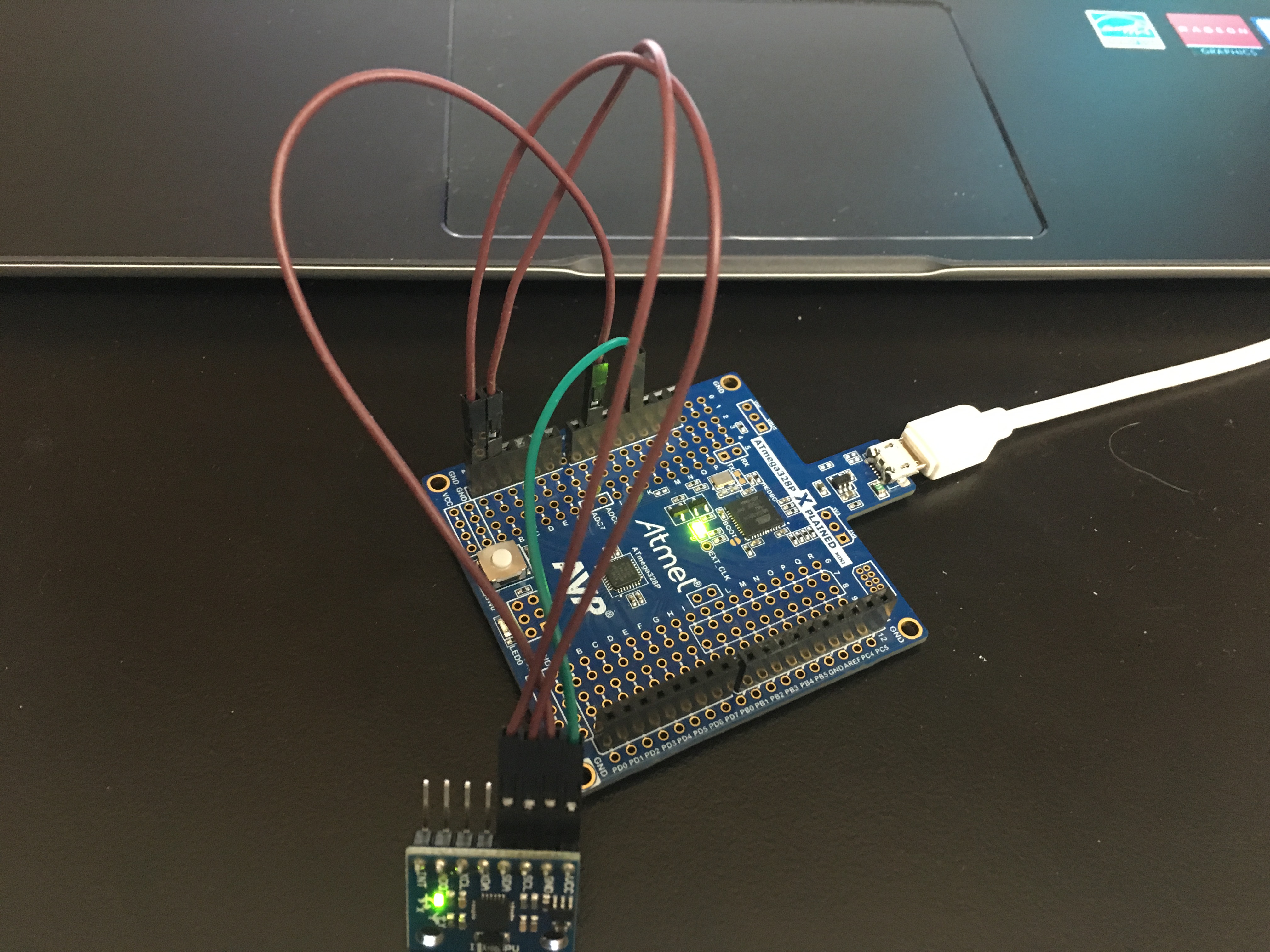
N/A

1. **SCHEMATICS**



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



1. **SCREENSHOT OF EACH DEMO (BOARD SETUP)** 
2. **VIDEO LINKS OF EACH DEMO**

<https://www.youtube.com/watch?v=q2rA5xOTPCk>

1. **GITHUB LINK OF THIS DA**

<https://github.com/BarrChris/submission_da/tree/master/DesignAssignments/DA>6

**Student Academic Misconduct Policy**

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

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

Chris Barr