CPE301 – SPRING 2020

Design Assignment 6

Student Name: Minsung Cho

Student #: 2001446442

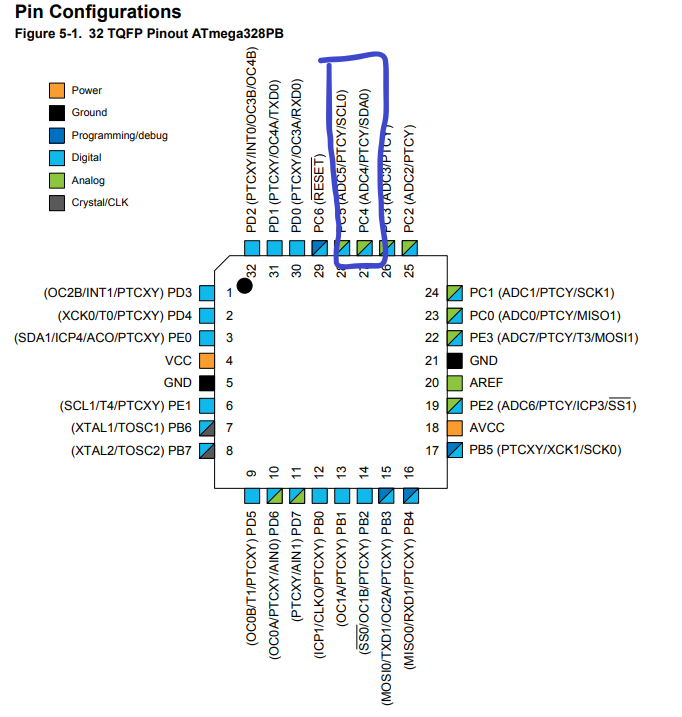
Student Email: chom3@unlv.nevada.edu

Primary Github address: <https://github.com/cho-minsung>

Directory: <https://github.com/cho-minsung/assignment6>

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

Atmega328PB



MPU-6050 6-DOF IMU Sensor

1. **Task 1 code**

#define F\_CPU 16000000UL // 16MHz CPU Clock

#include <avr/io.h>

#include <util/delay.h>

#include <inttypes.h>

#include <stdlib.h>

#include <stdio.h>

//codes from Dr. Venki's repository

#include "MPU6050\_def.h"

#include "i2c\_master.h"

#include "uart.h"

// values from MPU6050 from Dr. Venki's repo

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

// function prototypes

void MPU6050\_Init();

void MPU\_Start\_Loc();

void Read\_RawValue();

void printForPlot();

void printForTerminal();

int main()

{

//initialization codes

I2C\_Init();

MPU6050\_Init();

USART\_Init(9600);

while(1)

{

Read\_RawValue();

display();

}

}

// function to initialize MPU6050

void MPU6050\_Init(){

\_delay\_ms(150); // wait 150ms for power up

I2C\_Start\_Wait(0xD0); // start the device with write address

I2C\_Write(SMPLRT\_DIV); // write to the sample rate register

I2C\_Write(0x07); // use a 1KHz sample rate

I2C\_Stop(); // stop I2C

I2C\_Start\_Wait(0xD0);

I2C\_Write(PWR\_MGMT\_1); // write to the power management register

I2C\_Write(0x01); // reference frequency for gyro x-axis

I2C\_Stop();

I2C\_Start\_Wait(0xD0);

I2C\_Write(CONFIG); // write to config register

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

I2C\_Stop();

I2C\_Start\_Wait(0xD0);

I2C\_Write(GYRO\_CONFIG); // write to gyro config register

I2C\_Write(0x18); // use the full-scale range

I2C\_Stop();

I2C\_Start\_Wait(0xD0);

I2C\_Write(INT\_ENABLE); // write to the interrupt enable register

I2C\_Write(0x01);

I2C\_Stop();

}

// start the MPU loc

void MPU\_Start\_Loc(){

I2C\_Start\_Wait(0xD0); // start the device with write address

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

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

}

// read the gyroscope values

void Read\_RawValue(){

MPU\_Start\_Loc();

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());

Temperature = (((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();

}

// function to send to terminal

void display(){

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

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

float Ya = Acc\_y/16384.0;

float Za = Acc\_z/16384.0;

float Xg = Gyro\_x/16.4;

float Yg = Gyro\_y/16.4;

float Zg = Gyro\_z/16.4;

float t = (Temperature/340.00)+36.53; // convert temperature to C

// print Xa value

dtostrf( Xa, 3, 2, float\_ );

sprintf(buffer," Ax = %s g\t",float\_);

USART\_SendString(buffer);

// print Ya value

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

sprintf(buffer," Ay = %s g\t",float\_);

USART\_SendString(buffer);

// print Za value

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

sprintf(buffer," Az = %s g\t",float\_);

USART\_SendString(buffer);

// print temperature value

dtostrf( t, 3, 2, float\_ );

sprintf(buffer," T = %s%cC\r\n",float\_,0xF8);

USART\_SendString(buffer);

// print Xg value

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

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

USART\_SendString(buffer);

// print Yg value

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

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

USART\_SendString(buffer);

// print Zg value

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

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

USART\_SendString(buffer);

\_delay\_ms(1000);

}

* All the rest of the codes included were from Dr. Venki’s repo.

1. **Task 2 code**

#define ACCELEROMETER\_SENSITIVITY 16384.0

#define GYROSCOPE\_SENSITIVITY 16.4

#define dt 0.01

void ComplementaryFilter()

{

folat pitchAcc, rollAcc;

pitch += ((float)gyrData[0] / GYROSCOPE\_SENSITIVITY) \* dt;

roll -= ((float)gyrData[1] / GYROSCOPE\_SENSITIVITY) \* dt;

int forceMagnitudeApprox = \ abs(accData[0]) + abs(accData[1]) + abs(accData[2]);

if (forceMagnitudeApprox > 8192 && forceMagnitudeApprox < 32768)

{

pitchAcc = atan2f((float)accData[1], (float)accData[2]) \* 180 / M\_PI;

pitch = pitch \* 0.98 + pitchAcc \* 0.02;

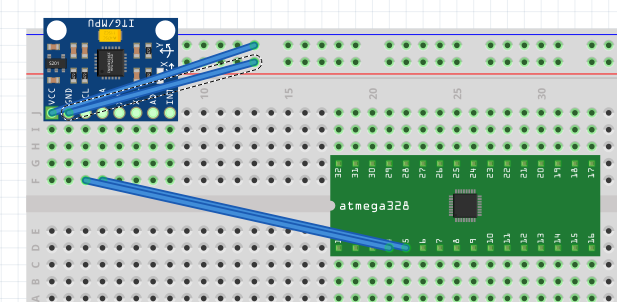
rollAcc = atan2f((float)accData[0], (float)accData[2]) \* 180 /M\_PI;

roll = roll \* 0.98 + rollAcc \* 0.02;

}

}

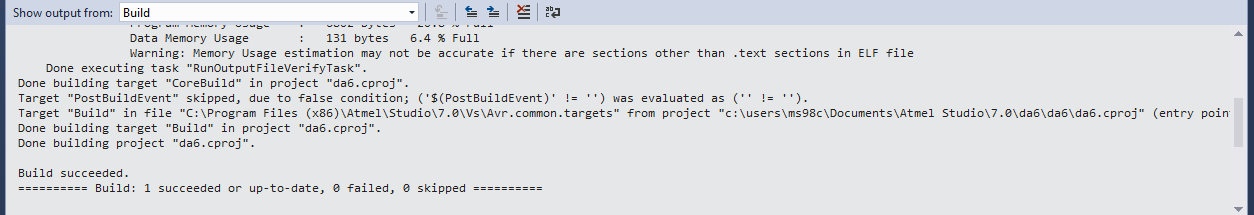
1. **SCHEMATICS**



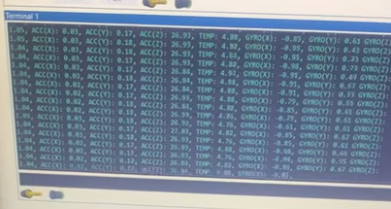
Pin C4&5 used.

VCC and GND used.

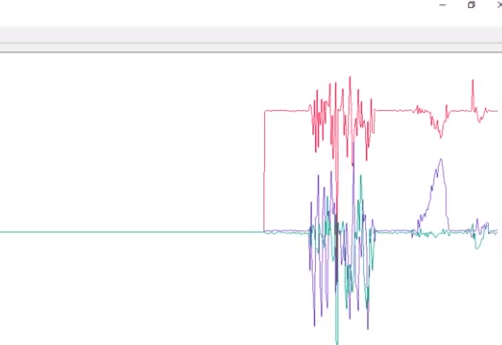
1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**



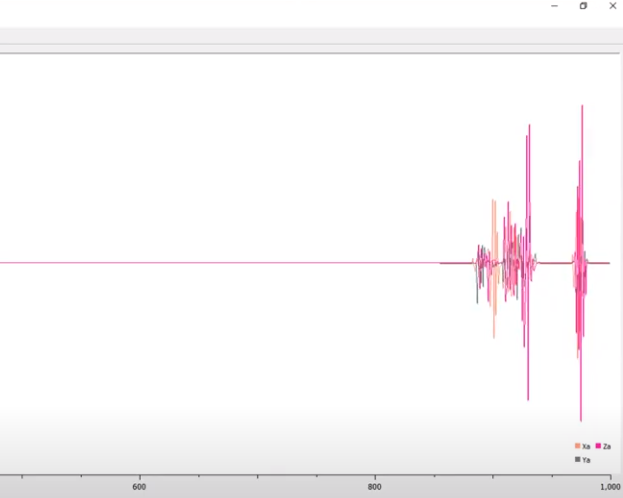
Task 1



Terminal info

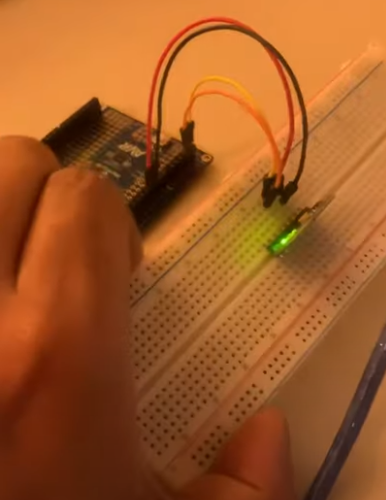


Gyro Serial plotter



Acceleration serial plotter

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



1. **VIDEO LINKS OF EACH DEMO**

<https://www.youtube.com/watch?v=vW8vL76GRbA>

1. **GITHUB LINK OF THIS DA**

<https://github.com/cho-minsung/assignment6>

**Student Academic Misconduct Policy**

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

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

Minsung Cho