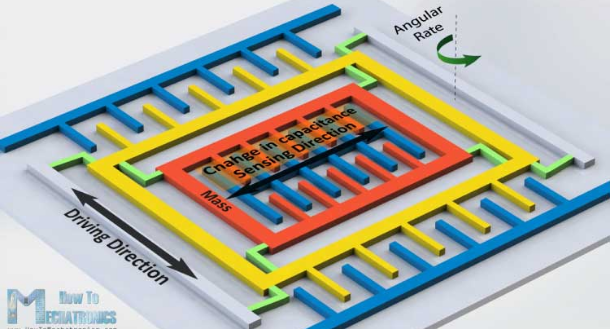
**Technocrats Task Report for Week 3 and Week 4**

**Gyroscope:**

It measures the speed in which the device is rotated about a particular axis. It can have three different axis(x, y and z). On more technical words it measure the angular. The gyro sensor in its MEMS structure has a capacitive based transduction system. The suspended mass based on the orientation of the device shows variation in the capacitance and the the required field (angular velocity) is found out. When there is an angular velocity a particular axis for a mass there is always a force acting on the mass normal to it. This force exerted on the mass causes change in capacitance of the MEMS sensor.



So, L3G4200D is a gyroscope sensor module.

We will learn how to interface it with arduino.

The L3G4200D uses I2C and SPI communication protocol. We already know I2C communication has a data pin and an external clock pin for communicating the data. Its the protocol that can operate in full duplex mode(transmitting and receiving simultaneously).

I2C is more recommnded because it has its own advantage(more importantly it uses only two wires).

It is a tri-axis device which means that it can calculate angular velocity for three directions independent of each other. The device can be operated in three different scales(the maximum value it can read upto). The full scale can be chosen based on the requirement of the project. One point to consider is that "Higher the full scale reading lower the sensitivity of the instrument"

In I2C there is a register address to access values from the gyro sensor. The address value can be known from the datasheet of the module. Here the address is 105.

SDA pin is connected to A4 and SCL pin is connected to A5.

#include <Wire.h>

#define CTRL\_REG1 0x20

#define CTRL\_REG2 0x21

#define CTRL\_REG3 0x22

#define CTRL\_REG4 0x23

#define CTRL\_REG5 0x24

int L3G4200D\_Address = 105; //I2C address of the L3G4200D

int x;

int y;

int z;

void setup(){

Wire.begin();

Serial.begin(9600);

Serial.println("starting up L3G4200D");

setupL3G4200D(2000); // Configure L3G4200 - 250, 500 or 2000 deg/sec

delay(1500); //wait for the sensor to be ready

}

void loop(){

getGyroValues(); // This will update x, y, and z with new values

Serial.print("X:");

Serial.print(x);

Serial.print(" Y:");

Serial.print(y);

Serial.print(" Z:");

Serial.println(z);

delay(100); //Just here to slow down the serial to make it more readable

}

void getGyroValues(){

byte xMSB = readRegister(L3G4200D\_Address, 0x29);

byte xLSB = readRegister(L3G4200D\_Address, 0x28);

x = ((xMSB << 8) | xLSB);

byte yMSB = readRegister(L3G4200D\_Address, 0x2B);

byte yLSB = readRegister(L3G4200D\_Address, 0x2A);

y = ((yMSB << 8) | yLSB);

byte zMSB = readRegister(L3G4200D\_Address, 0x2D);

byte zLSB = readRegister(L3G4200D\_Address, 0x2C);

z = ((zMSB << 8) | zLSB);

}

int setupL3G4200D(int scale){

//From Jim Lindblom of Sparkfun's code

// Enable x, y, z and turn off power down:

writeRegister(L3G4200D\_Address, CTRL\_REG1, 0b00001111);

// If you'd like to adjust/use the HPF, you can edit the line below to configure CTRL\_REG2:

writeRegister(L3G4200D\_Address, CTRL\_REG2, 0b00000000);

// Configure CTRL\_REG3 to generate data ready interrupt on INT2

// No interrupts used on INT1, if you'd like to configure INT1

// or INT2 otherwise, consult the datasheet:

writeRegister(L3G4200D\_Address, CTRL\_REG3, 0b00001000);

// CTRL\_REG4 controls the full-scale range, among other things:

if(scale == 250){

writeRegister(L3G4200D\_Address, CTRL\_REG4, 0b00000000);

}else if(scale == 500){

writeRegister(L3G4200D\_Address, CTRL\_REG4, 0b00010000);

}else{

writeRegister(L3G4200D\_Address, CTRL\_REG4, 0b00110000);

}

// CTRL\_REG5 controls high-pass filtering of outputs, use it

// if you'd like:

writeRegister(L3G4200D\_Address, CTRL\_REG5, 0b00000000);

}

void writeRegister(int deviceAddress, byte address, byte val) {

Wire.beginTransmission(deviceAddress); // start transmission to device

Wire.write(address); // send register address

Wire.write(val); // send value to write

Wire.endTransmission(); // end transmission

}

int readRegister(int deviceAddress, byte address){

int v;

Wire.beginTransmission(deviceAddress);

Wire.write(address); // register to read

Wire.endTransmission();

Wire.requestFrom(deviceAddress, 1); // read a byte

while(!Wire.available()) {

// waiting

}

v = Wire.read();

return v;

}