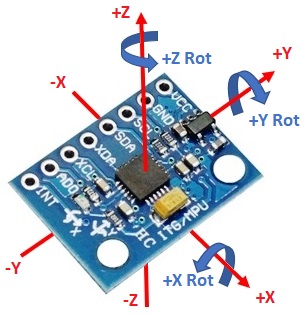
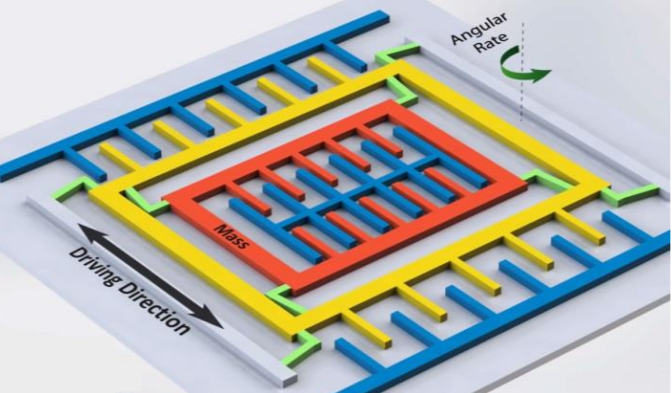
**Labo MPU6050 Gyrosensor**

**Doelstelling**

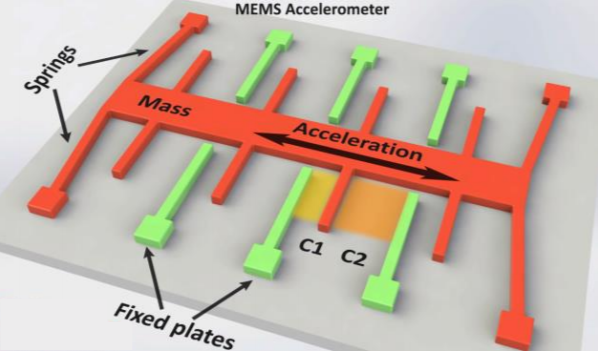
Het doel van dit labo is om te begrijpen hoe een gyrosensor en accelerometer werken en de waarden van de sensoren (gyrosensor, accelerometer en temperatuursensor) kunnen uitlezen via I2C en die weer te geven als grafiek.

**Theorie**

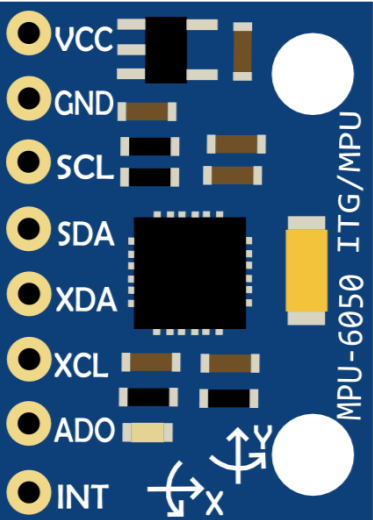
Hier zien we de 3 assen waarop de gyrosensor en accelerometer zullen meten. De blauwe pijlen stellen de meetpunten van de gyrosensor voor en de rode pijlen stellen de meetpunten van de accelerometer voor. De gyrosensor meet draaiingen t.o.v. de zwaartekracht. De accelerometer meet lineaire bewegingsverschillen. Dit zal later wel duidelijker worden.



Dit is een foto die de werking van een gyrosensor laat zien. Het gele deel is een constant oscillerende kern. Wanneer er dan een draaibeweging voorkomt dan verschuift de binnenste (oranje) plaat. Er wordt constant de capaciteit tussen de 2 binnenste (oranje en blauwe) platen gemeten en die zal veranderen als er een draaibeweging voorkomt.



Dit is een foto die de werking van een accelerometer laat zien. De accelerometer bestaat uit (groene) vaste plaatjes en (oranje) bewegende plaatsjes. Er wordt ook constant de capaciteit tussen de 2 plaatjes gemeten en als er dan een beweging voorkomt dan zal de capaciteit veranderen.

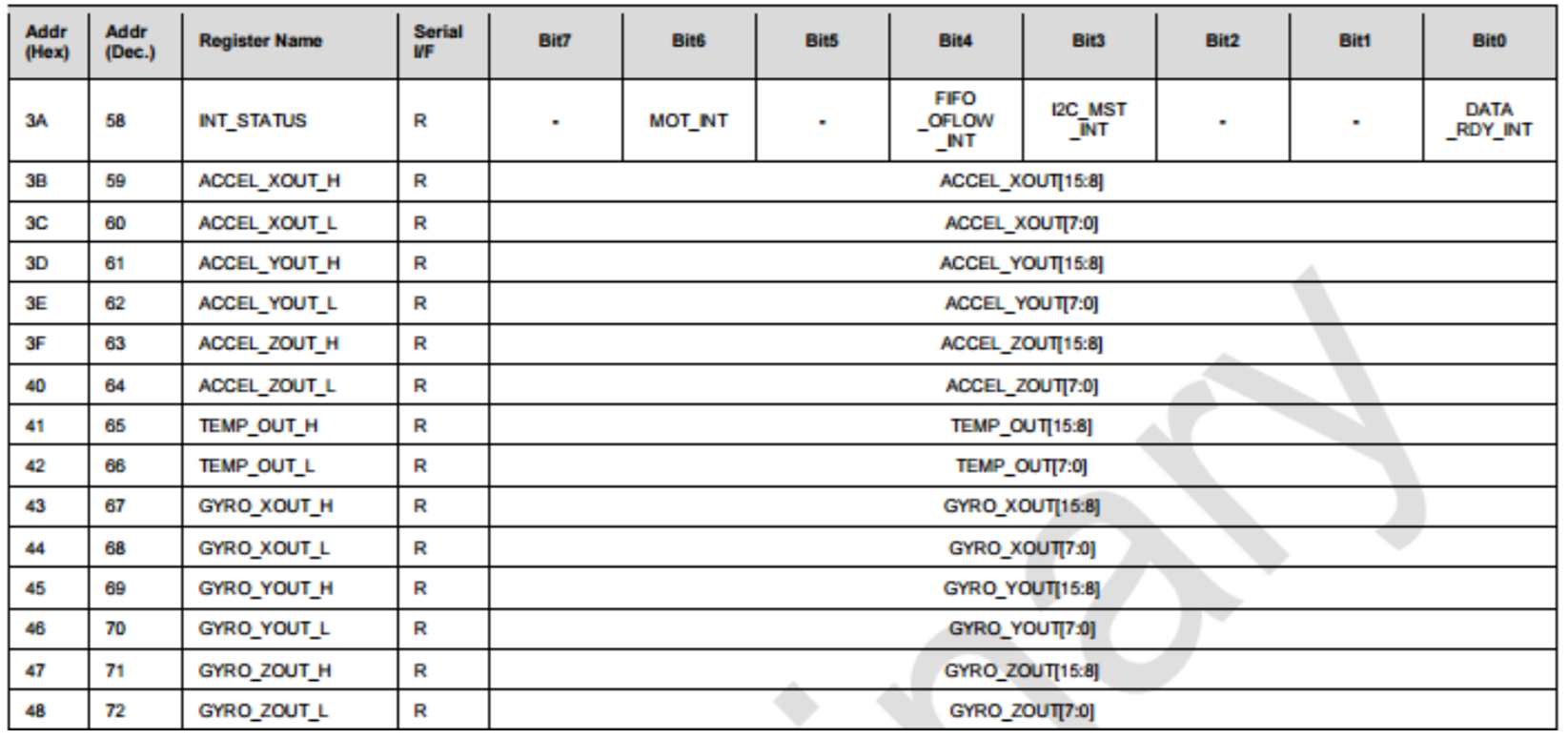
**MPU6050**

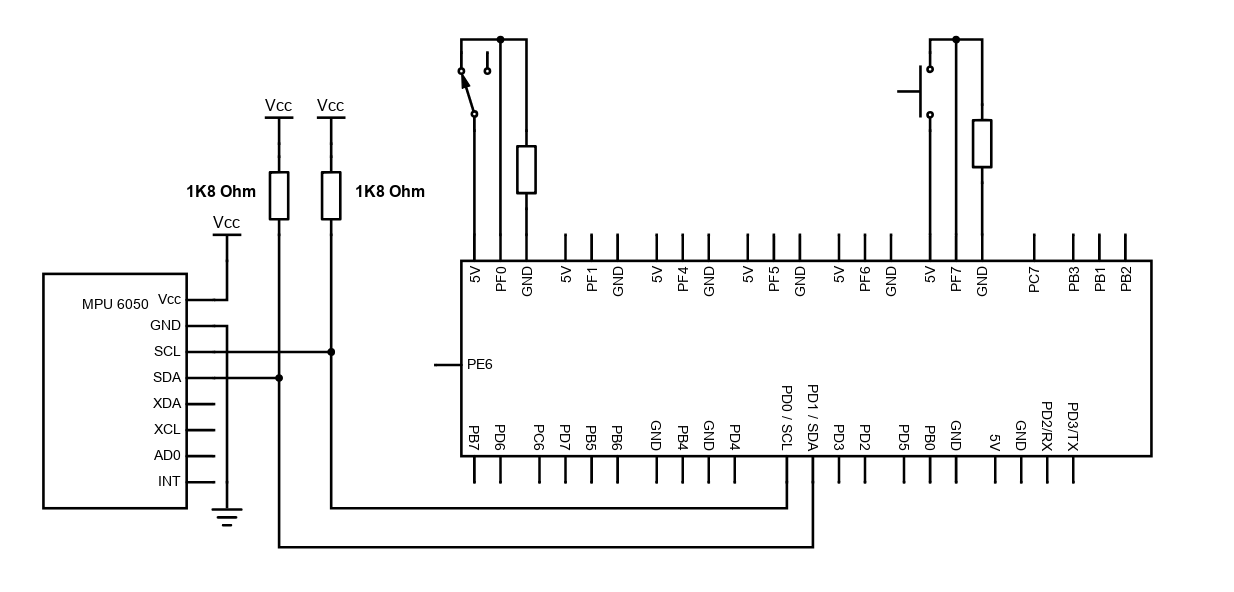
Hier zie je de pin lay-out van MPU6050:

* Vcc is waar je +5V moet aansluiten.
* GND is de ground.
* SCL is de “Serial Clock” voor I2C, hier komt het kloksignaal van de master binnen.
* SDA is de “Serial Data” voor I2C, hier komt de data voor de master er uit.
* XDA is “Auxiliary Serial Data” en wordt gebruikt om met andere I2C modules te communiceren.
* XCL is “Auxiliary Serial Clock” idem als XDA.
* AD0 is voor het slave address aan te passen als je 2 MPU6050s wilt aansluiten aan 1 master.
* INT is de interrupt pin wanneer er nieuwe leesbare data is voor de master

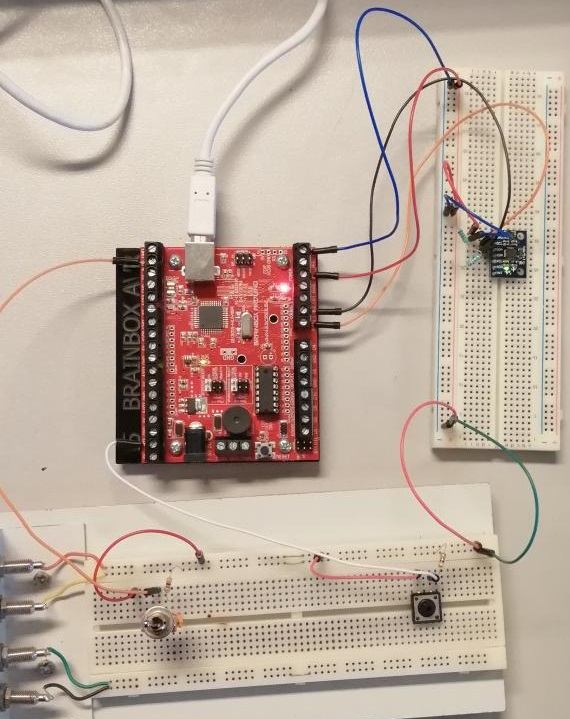
In mijn situatie ga ik alleen de eerste 4 pinnen (Vcc, GND, SCL en SDA) gebruiken.

Het slave address van de MPU6050 is standaard 0x68 = 0110 100X (0x69 als AD0 hoog is), de laatste bit is dan de R/W bit. Dus als de MPU6050 in “read mode” moet zijn is het address 0xD0 en als de MPU6050 in “write mode” moet zijn zal het address 0xD1 zijn. (Slave address is te vinden in de datasheet)

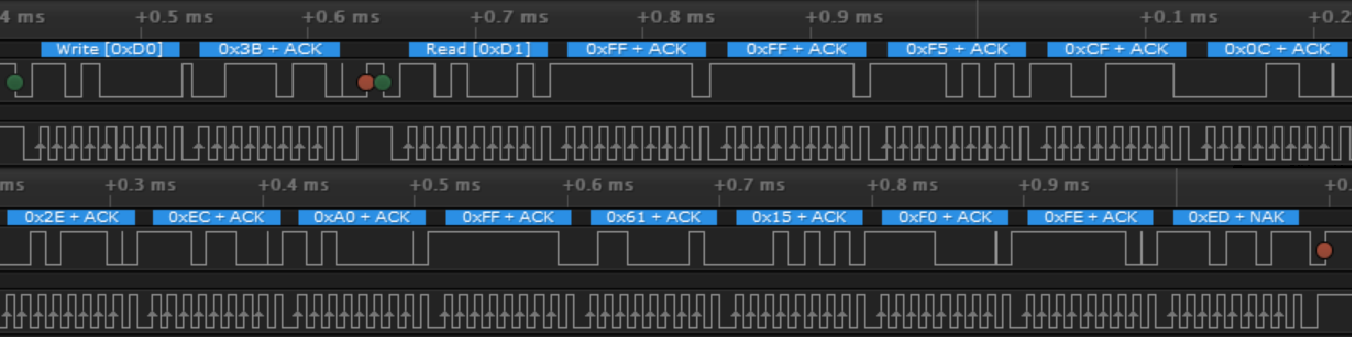
Hier is een foto van het index register van de MPU6050. Ik gebruik elk register behalve het bovenste want ik gebruik de INT pin niet. Elke as bij een sensor heeft 2 aparte indexen die een byte groot zijn, de 8 MSB (H) en de 8 LSB (L), dus de totale index voor 1 as is 16 bits lang.

**Schema**

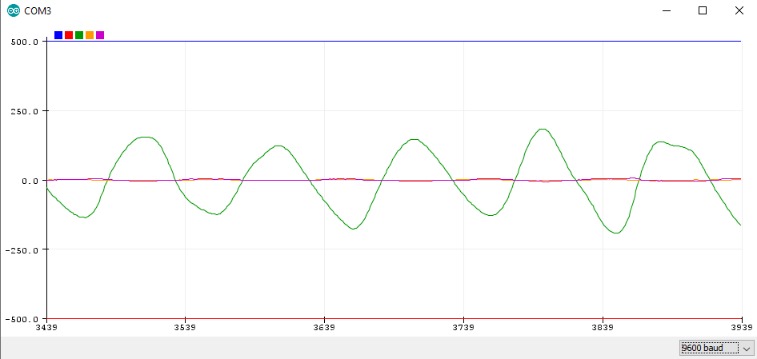
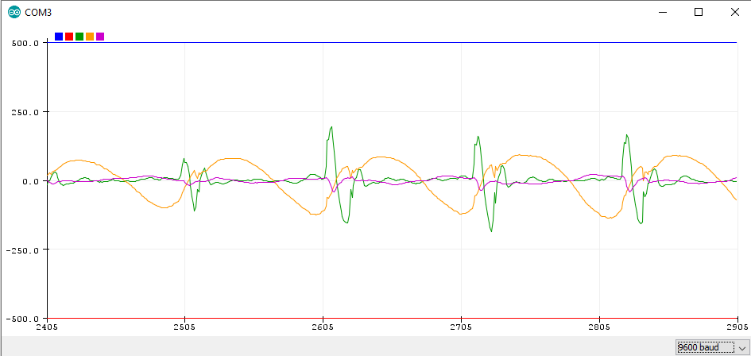
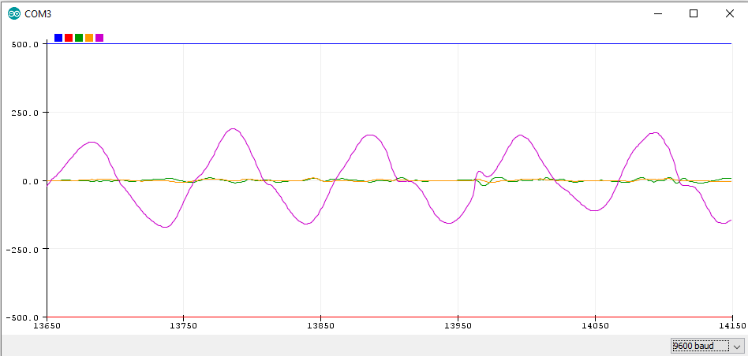
Meetpunten

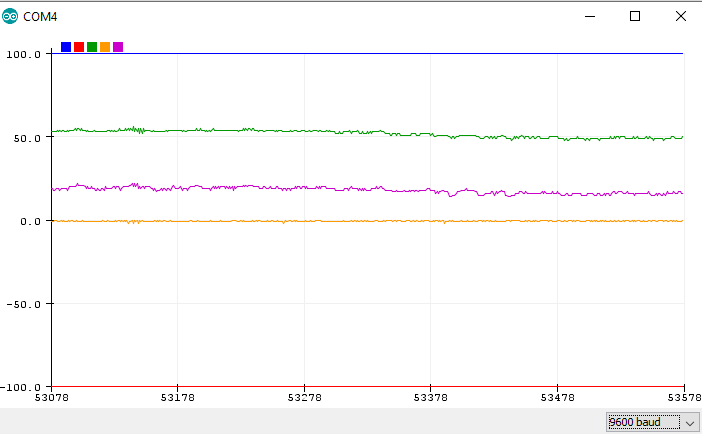
**Beschrijving**

Ik gebruikt de 3 sensoren in dit labo. Ik gebruik de accelerometer om hoeken te meten (zoals een waterpas), de gyrosensor meet de hoeksnelheid en de temperatuursensor meet de kamertemperatuur. Er zijn ook 2 schakelaars, een drukknop en een bistabiele schakelaar. De drukknop pauzeert de meting als je er op drukt en de schakelaar bepaalt wanneer de metingen beginnen bij de start van het programma.

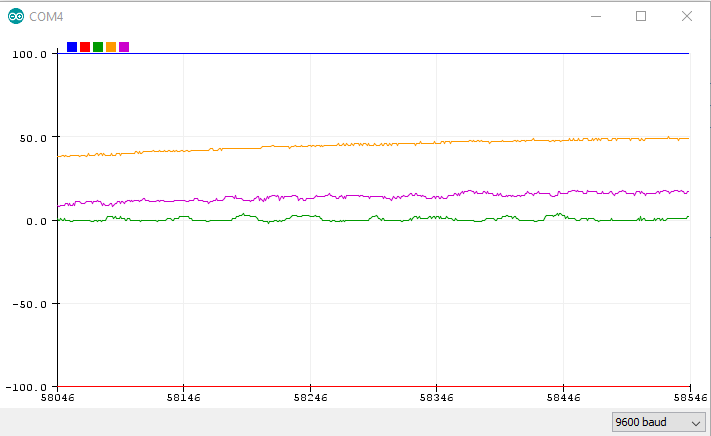
**Metingen**

Hier is een foto van de 14 bytes die worden gestuurd naar de brainbox. Eerst wordt 0xD0 en 0x3B gestuurd, dat is het write address en de eerste index van de MPU en zal elke keer als ik data inlees hetzelfde zijn. Daarna is er een stop (rode en groene bol) en 0xD1, dat is om te veranderen naar read mode en zal ook elke keer hetzelfde zijn. Nu komen de 14 bytes. De eerste 6 bytes zijn van de accelerometer (2 per as), de 2 hierna zijn voor de temp. sensor en de laatste 6 zijn die van de gyro sensor (2 per as).

****Hier zijn ook nog metingen van de seriële plotter van arduino. De eerste 3 zijn de X Y en Z as metingen van de accelerometer en de laatste 2 zijn de X en Y as van de gyrosensor (Z as maakt niet uit met mijn programma). Bij de eerste 3 doe ik hetzelfde, ik beweeg het breadboard op en neer op de as van de meting. Dus eerst gaat het naar boven (dan stijgt de gemeten waarde) en daarna terug naar beneden (de meting daalt en wordt negatief want ik ga de tegenovergestelde richting op).

****

Bij deze 2 metingen (X is groene en Y is oranje) heb ik het breadboard gekanteld tot +-50° en het zo even laten staan. Dit zou bvb gebruikt kunnen worden om te meten hoeveel graden een object is gekanteld.

****

Wat niet perfect is aan deze metingen is dat ik de accelerometer gebruik voor het aantal graden en de gyrosensor voor versnellingen te meten. Mijn reden hiervoor zijn de formules die ik heb gebruikt. Er zijn formules nodig om de rauwe waardes die van de MPU6050 komen om te vormen naar versnelling. De juiste formules vinden was helemaal niet simpel en koste veel tijd. De enige werkende formules die ik vond zijn degene die ik nu gebruik. Ook al is het een onorthodoxe manier om de data uit te lezen, het werkt nog steeds heel goed dus zie ik er geen probleem mee. Het enige probleem dat ik zie is dat de gyrosensor beter zal zijn in hoeken meten dan de accelerometer en vice versa.

**Software**

#include <Wire.h> // I2C lib

boolean Hold**;** // Variable om de metingen te laten stoppen en in hold mode gaan

float GyX**,** GyY**,** GyZ**;** // Variable om de rauwe accelerometer data in op te slaan

float AcX**,** AcY**,** AcZ**;** // Variable om de rauwe gyro sensor data in op te slaan

float Temp**;** // Variable om de rauwe temperatuur sensor data in op te slaan

int GyX\_Int**,** GyY\_Int**,** GyZ\_Int**;** // Variable om de omgerekende afgeronde accelerometer data in op de slaan

int AcX\_Int**,** AcY\_Int**,** AcZ\_Int**;** // Variable om de omgerekende afgeronde gyro sensor data in op de slaan

int Temp\_Int**;** // Variable om de omgerekende afgeronde temperatuur sensor data in op de slaan

float GyX\_Offset**,** GyY\_Offset**,** GyZ\_Offset**;** // Variable om de offset van de accelerometer data in op de slaan

float AcX\_Offset**,** AcY\_Offset**,** AcZ\_Offset**;** // Variable om de offset van de gyro sensor data in op de slaan

void Read\_Data**();** // Function waar de 14 bytes worden ingelezen van de MPU6050

void setup**()**

**{**

**while(**PINF **&** 0b10000000**)** // Zolang PF7 hoog blijft doe dan niets

**{**

Serial**.**println**(**"Haal rode schakelaar over om te starten."**);** // Print instructies op de monitor

**}**

Serial**.**begin**(**9600**);** // Baut rate van de seriele monitor is 9600

Wire**.**begin**();** // Start de I2C connectie

// Start de MPU6050

Wire**.**beginTransmission**(**0x68**);** // Start de communicatie met de MPU-6050 (write)

Wire**.**write**(**0x6B**);** // Selecteer de index

Wire**.**write**(**0**);** // Stuur data naar geselcteerde index

Wire**.**endTransmission**();** // Eindig de transmissie

// Zet de accelerometer zijn Full Scale Range op 8g (FS\_SEL = 2)

Wire**.**beginTransmission**(**0x68**);** // Start de communicatie met de MPU-6050 (write)

Wire**.**write**(**0x1C**);** // Selecteer de index

Wire**.**write**(**0x10**);** // Stuur data naar geselcteerde index

Wire**.**endTransmission**();** // Eindig de transmissie

// Zet de gyrosensor zijn Full Scale Range op 500 (FS\_SEL = 1)

Wire**.**beginTransmission**(**0x68**);** // Start de communicatie met de MPU-6050 (write)

Wire**.**write**(**0x1B**);** // Selecteer de index

Wire**.**write**(**0x08**);** // Stuur data naar geselcteerde index

Wire**.**endTransmission**();** // Eindig de transmissie

// Berekend offset

**for** **(**int Repeat **=** 0**;** Repeat **<** 2000**;** Repeat **++)** // Lees 2000 keer de 14 bytes in in

**{**

Read\_Data**();** // Lees de 14 bytes in

GyX\_Offset **+=** GyX**;** // Tel de gemete waarde van GyX elke keer bij elkaar op / integreer de data

GyY\_Offset **+=** GyY**;** // Tel de gemete waarde van GyY elke keer bij elkaar op / integreer de data

GyZ\_Offset **+=** GyZ**;** // Tel de gemete waarde van GyZ elke keer bij elkaar op / integreer de data

AcX\_Offset **+=** AcX**;** // Tel de gemete waarde van AcX elke keer bij elkaar op / integreer de data

AcY\_Offset **+=** AcY**;** // Tel de gemete waarde van AcY elke keer bij elkaar op / integreer de data

AcZ\_Offset **+=** AcZ**;** // Tel de gemete waarde van AcZ elke keer bij elkaar op / integreer de data

Serial**.**println**(**"BEREKENEN, MPU NIET BEWEGEN!"**);** // Print instructies op monitor

**}**

GyX\_Offset **/=** 2000**;** // Deel de totale GyX\_Offset waarde door 2000 voor het gemiddelde te krijgen

GyY\_Offset **/=** 2000**;** // Deel de totale GyY\_Offset waarde door 2000 voor het gemiddelde te krijgen

GyZ\_Offset **/=** 2000**;** // Deel de totale GyZ\_Offset waarde door 2000 voor het gemiddelde te krijgen

AcX\_Offset **/=** 2000**;** // Deel de totale AcX\_Offset waarde door 2000 voor het gemiddelde te krijgen

AcY\_Offset **/=** 2000**;** // Deel de totale AcY\_Offset waarde door 2000 voor het gemiddelde te krijgen

AcZ\_Offset **/=** 2000**;** // Deel de totale AcZ\_Offset waarde door 2000 voor het gemiddelde te krijgen

**}**

void Read\_Data**()**

**{**

Wire**.**beginTransmission**(**0x68**);** // Start de communicatie met de MPU-6050 (write)

Wire**.**write**(**0x3B**);** // Selecteer de index (0x3B is waar de eerste van de 14 bytes in staat)

Wire**.**endTransmission**();** // Eindig de transmissie

Wire**.**requestFrom**(**0x68**,** 14**);** // Vraag de 14 opvolgende bytes van de MPU6050, 14 want 6 van de gyro, 6 van de accelero en 1 voor temp (read) (van 0x3B tot 0x48)

**while(**Wire**.**available**()** **<** 14**);** **{}** // Wachten tot alle 14 bytes beschikbaar zijn

// Inlezen

AcX **=** **-(**Wire**.**read**()** **<<** 8 **|** Wire**.**read**());** // Eerst worden de 8 LSB ingelezen en daarna de 8 MSB, eerst worden de LSBs toegevoegt en dan de MSBs die 8 plaatsen naar links zijn geschift

AcY **=** **-(**Wire**.**read**()** **<<** 8 **|** Wire**.**read**());** // Eerst worden de 8 LSB ingelezen en daarna de 8 MSB, eerst worden de LSBs toegevoegt en dan de MSBs die 8 plaatsen naar links zijn geschift

AcZ **=** **-(**Wire**.**read**()** **<<** 8 **|** Wire**.**read**());** // Eerst worden de 8 LSB ingelezen en daarna de 8 MSB, eerst worden de LSBs toegevoegt en dan de MSBs die 8 plaatsen naar links zijn geschift

Temp **=** Wire**.**read**()** **<<** 8 **|** Wire**.**read**();** // Eerst worden de 8 LSB ingelezen en daarna de 8 MSB, eerst worden de LSBs toegevoegt en dan de MSBs die 8 plaatsen naar links zijn geschift

GyY **=** **-(**Wire**.**read**()** **<<** 8 **|** Wire**.**read**());** // Eerst worden de 8 LSB ingelezen en daarna de 8 MSB, eerst worden de LSBs toegevoegt en dan de MSBs die 8 plaatsen naar links zijn geschift

GyX **=** Wire**.**read**()** **<<** 8 **|** Wire**.**read**();** // Eerst worden de 8 LSB ingelezen en daarna de 8 MSB, eerst worden de LSBs toegevoegt en dan de MSBs die 8 plaatsen naar links zijn geschift

GyZ **=** Wire**.**read**()** **<<** 8 **|** Wire**.**read**();** // Eerst worden de 8 LSB ingelezen en daarna de 8 MSB, eerst worden de LSBs toegevoegt en dan de MSBs die 8 plaatsen naar links zijn geschift

**}**

void loop**()**

**{**

Read\_Data**();** // Lees de 14 bytes in

// Offset aftrekken

GyX **-=** GyX\_Offset**;** // Trek de offset af van de gemete waarde voor een meer accurate gemete waarde

GyY **-=** GyY\_Offset**;** // Trek de offset af van de gemete waarde voor een meer accurate gemete waarde

GyZ **-=** GyZ\_Offset**;** // Trek de offset af van de gemete waarde voor een meer accurate gemete waarde

AcX **-=** AcX\_Offset**;** // Trek de offset af van de gemete waarde voor een meer accurate gemete waarde

AcY **-=** AcY\_Offset**;** // Trek de offset af van de gemete waarde voor een meer accurate gemete waarde

AcZ **-=** AcZ\_Offset**;** // Trek de offset af van de gemete waarde voor een meer accurate gemete waarde

// Berekeningen

GyX **/=** 65.5**;** // Dit zet de output om naar °/s want als de sensor draait aan 1°/S dan zal de output waarde 65.5 zijn (datasheet p12)

GyY **/=** 65.5**;** // Dit zet de output om naar °/s want als de sensor draait aan 1°/S dan zal de output waarde 65.5 zijn (datasheet p12)

GyZ **/=** 65.5**;** // Dit zet de output om naar °/s want als de sensor draait aan 1°/S dan zal de output waarde 65.5 zijn (datasheet p12)

AcX **/=** 45.51111**;** // Dit zet de output om naar ° want als de sensor 90°zou zijn gekanteld dan zal de output waarde 4096 zijn(4096/90 = 45.51) (datasheet p13)

AcY **/=** 45.51111**;** // Dit zet de output om naar ° want als de sensor 90°zou zijn gekanteld dan zal de output waarde 4096 zijn(4096/90 = 45.51) (datasheet p13)

AcZ **/=** 45.51111**;** // Dit zet de output om naar ° want als de sensor 90°zou zijn gekanteld dan zal de output waarde 4096 zijn(4096/90 = 45.51) (datasheet p13)

Temp **/=** 340**;** // Dit zet de output om naar °C

Temp **+=** 36.53**;** // Dit zet de output om naar °C

// Afronden

AcX\_Int **=** **(**int**)**AcX**;** // Zet AcX om naar een int zodat het wordt afgerond tot een eenheid zonder kommagetallen

AcY\_Int **=** **(**int**)**AcY**;** // Zet AcY om naar een int zodat het wordt afgerond tot een eenheid zonder kommagetallen

AcZ\_Int **=** **(**int**)**AcZ**;** // Zet AcZ om naar een int zodat het wordt afgerond tot een eenheid zonder kommagetallen

Temp\_Int **=** **(**int**)**Temp**;** // Zet Temp om naar een int zodat het wordt afgerond tot een eenheid zonder kommagetallen

GyX\_Int **=** **(**int**)**GyX**;** // Zet GyX om naar een int zodat het wordt afgerond tot een eenheid zonder kommagetallen

GyY\_Int **=** **(**int**)**GyY**;** // Zet GyY om naar een int zodat het wordt afgerond tot een eenheid zonder kommagetallen

GyZ\_Int **=** **(**int**)**GyZ**;** // Zet GyZ om naar een int zodat het wordt afgerond tot een eenheid zonder kommagetallen

// Data printen

Serial**.**print**(**"AcX = "**);**

Serial**.**print**(**AcX\_Int**);**

Serial**.**print**(**" "**);**

Serial**.**print**(**"AcY = "**);**

Serial**.**print**(**AcY\_Int**);**

Serial**.**print**(**" "**);**

Serial**.**print**(**"AcZ = "**);**

Serial**.**print**(**AcZ\_Int**);**

Serial**.**print**(**" "**);**

Serial**.**print**(**"Temp = "**);**

Serial**.**print**(**Temp\_Int**);**

Serial**.**print**(**" "**);**

Serial**.**print**(**"GyX = "**);**

Serial**.**print**(**GyX\_Int**);**

Serial**.**print**(**" "**);**

Serial**.**print**(**"GyY = "**);**

Serial**.**print**(**GyY\_Int**);**

Serial**.**print**(**" "**);**

Serial**.**print**(**"GyZ = "**);**

Serial**.**println**(**GyZ\_Int**);**

**if(**PINF **&** 0b00000001**)** // Als je drukt op de drukknop

**{**

delay**(**250**);** // Delay voor niet direct in nieuwe loop te springen

Hold **=** HIGH**;** // Maak de Hold varible hoog

Serial**.**println**(**"Hold"**);** // Print hold op de seriele monitor

**while(**Hold **==** 1**)** // Zolang Hold hoog is

**{**

**if(**PINF **&** 0b00000001**)** // En als je dan terug op de drukknop drukt

**{**

Hold **=** LOW**;** // Maak dan Hold laag

**}**

**}**

delay**(**250**);** // Delay voor niet direct in nieuwe loop te springen

**}**

**}**

**Besluit**

Door dit labo heb ik uit eigen ervaring en testen geleerd hoe de MPU6050 werkt. Door dit labo heb ik ook veel geleerd over I2C en ermee werken in Arduino en ook C code, maar om een of andere reden willen de waardes niet correct gemeten worden in C code (dit heb ik ondervonden in mijn I2C project voor EAC-ICT). De reden hiervoor weet ik echter niet. Mijn hypothese is dat het iets te maken heeft met de verschillen in libraries van C code en Arduino, maar dat kan ook niet kloppen.