MEMS-sensoren

MEMS staat voor Microelectromechanical systems en zijn zeer kleine (micrometers) mechanische systemen waar fysische veranderingen elektrisch meetbaar zijn. Zo bestaan er verscheidene Toepassingen van MEMS, van kleine microfoons in smartphones tot licht diffractie filters in optic fiber switches. MEMS kunnen ook gebruikt worden voor het meten van fysieke bewegingen zoals versnellingen (accelerometer), hoekverdraaiing (gyroscoop) en kompas (magnetometer).

Er bestaat een breed aanbod aan MEMS sensoren die enkel versnellingen meet ofwel alles in huis heeft en zowel versnellingen en hoekverdraaiing meet als ook fungeert als een digitaal kompas.

Hierbij wordt vaak over vrijheidsgraden gesproken. Een accelerometer zoals de MMA845X heeft zo drie vrijheidsgraden en de MPU-9150 heeft negen vrijheidsgraden en deze fungeert dan ook als accelerometer, gyroscoop en magnetometer. Deze sensoren kunnen op verschillende manieren uitgelezen worden (I²C, analoog, …).

https://en.wikipedia.org/wiki/Six_degrees_of_freedom

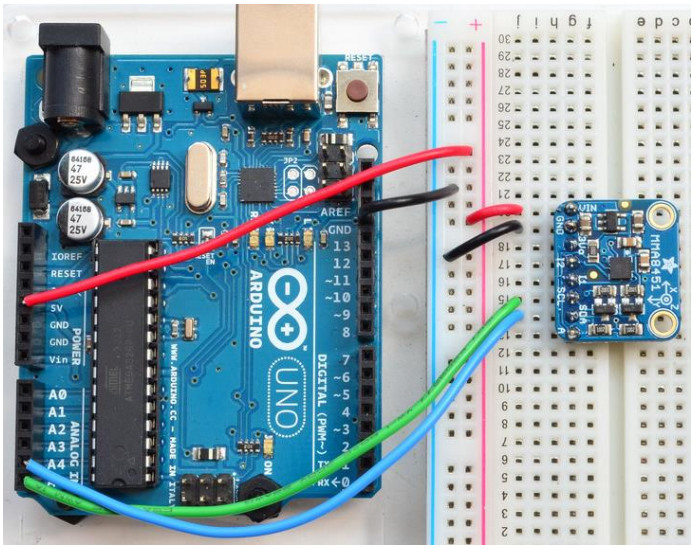
MMA845X (I²C)

De MMA845X is een accelerometer met 3 vrijheidsgraden, de X kan een getal voor stellen en deze aanwezig in het lab is MMA8451. Adafruit heeft hiervoor een library ter beschikking (Adafruit_MMA8451.h). Om deze sensor uit te lezen kan onderstaande voorbeeld code gebruikt worden en dienen drie includes gebruikt te worden: Wire.h die een I²C communicatie mogelijk maakt, Adafruit_MMA8451.h de library van de sensor zelf en Adafruit_Sensor.h die zorgt voor de juiste eenheden en dergelijke. Deze libraries kunnen op onderstaande link gedownload worden.

https://learn.adafruit.com/adafruit-mma8451-accelerometer-breakout/wiring-and-test

De sensor heeft een adres voor het I²C communicatie protocol deze is afhankelijk of the pin SA0 aan GND of de **3.3V** hangt (niet 5V), 0x1C (SA0 = 0), 0x1D (SA0 = 1) bij onderstaande voorbeeld code dient de SA0 pin hoog te zijn. De twee wires SCL en SDA voor de I²C communicatie dienen bij de Arduino UNO aangesloten te worden op de pinnen A4 & A5, hier voor zorgt de include wire.h.

https://www.arduino.cc/en/Reference/Wire



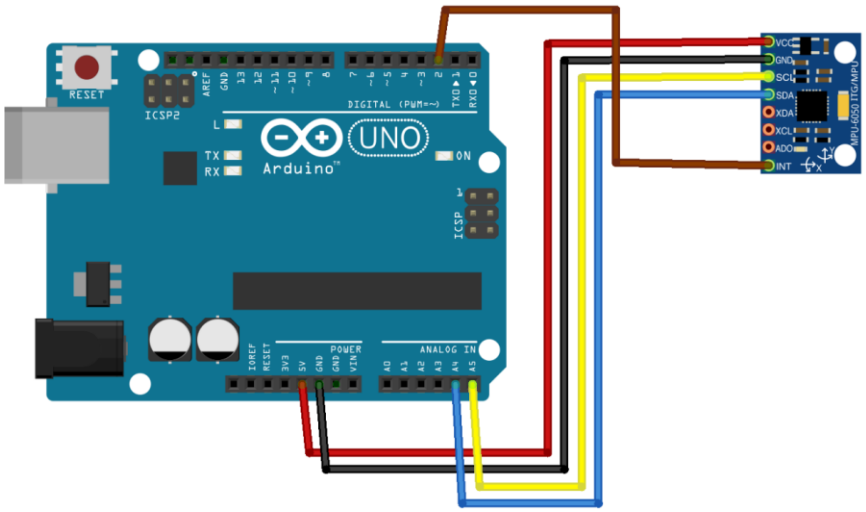
Voorbeeld code

```
/***************************************************************************
    @file     Adafruit_MMA8451.h
    @author   K. Townsend (Adafruit Industries)
    @license  BSD (see license.txt)
***************************************************************************/
#include <Wire.h>
#include <Adafruit_MMA8451.h>
#include <Adafruit_Sensor.h>

Adafruit_MMA8451 mma = Adafruit_MMA8451();

void setup(void) {
  Serial.begin(9600);

  Serial.println("Adafruit MMA8451 test!");


  if (! mma.begin()) {
    Serial.println("Couldnt start");
    while (1);
  }
  Serial.println("MMA8451 found!");

  mma.setRange(MMA8451_RANGE_2_G);

  Serial.print("Range = "); Serial.print(2 << mma.getRange());
  Serial.println("G");

}

void loop() {
  // Read the 'raw' data in 14-bit counts
  mma.read();
  Serial.print("X:\t"); Serial.print(mma.x);
  Serial.print("\tY:\t"); Serial.print(mma.y);
  Serial.print("\tZ:\t"); Serial.print(mma.z);
  Serial.println();

  /* Get a new sensor event */
  sensors_event_t event;
  mma.getEvent(&event);

  /* Display the results (acceleration is measured in m/s^2) */
  Serial.print("X: \t"); Serial.print(event.acceleration.x);
Serial.print("\t");
  Serial.print("Y: \t"); Serial.print(event.acceleration.y);
Serial.print("\t");
  Serial.print("Z: \t"); Serial.print(event.acceleration.z);
Serial.print("\t");
  Serial.println("m/s^2 ");

  /* Get the orientation of the sensor */
  uint8_t o = mma.getOrientation();

  switch (o) {
    case MMA8451_PL_PUF:
      Serial.println("Portrait Up Front");
      break;
    case MMA8451_PL_PUB:
```

```
      Serial.println("Portrait Up Back");
      break;
    case MMA8451_PL_PDF:
      Serial.println("Portrait Down Front");
      break;
    case MMA8451_PL_PDB:
      Serial.println("Portrait Down Back");
      break;
    case MMA8451_PL_LRF:
      Serial.println("Landscape Right Front");
      break;
    case MMA8451_PL_LRB:
      Serial.println("Landscape Right Back");
      break;
    case MMA8451_PL_LLF:
      Serial.println("Landscape Left Front");
      break;
    case MMA8451_PL_LLB:
      Serial.println("Landscape Left Back");
      break;
    }
  Serial.println();
  delay(500);
}
```

MPU-6050 (I²C)

De MPU-6050 is de voorloper van de MPU-9150, waarbij de MPU-6050 gebruikt wordt in samenwerking met een magnetometer. De MPU-6050 is een MEMS accelerometer en gyroscoop die via I²C uitgelezen kan worden met behulp van de wire.h library. Zowel de library I2Cdev.h en MPU6050.h dienen voor het gebruik van de MPU-6050 ook gebruikt te worden.



Voorbeeld code MPU-6050

```
// Arduino Wire library is required if I2Cdev I2CDEV_ARDUINO_WIRE
implementation
// is used in I2Cdev.h
#include "Wire.h"

// I2Cdev and MPU6050 must be installed as libraries, or else the .cpp/.h
files
```

```
// for both classes must be in the include path of your project
#include "I2Cdev.h"
#include "MPU6050.h"

// class default I2C address is 0x68
// specific I2C addresses may be passed as a parameter here
// AD0 low = 0x68 (default for InvenSense evaluation board)
// AD0 high = 0x69
MPU6050 accelgyro;

int16_t ax, ay, az;
int16_t gx, gy, gz;

#define LED_PIN 13
bool blinkState = false;

void setup() {
    // join I2C bus (I2Cdev library doesn't do this automatically)
    Wire.begin();

    // initialize serial communication
    // (38400 chosen because it works as well at 8MHz as it does at 16MHz,
but
    // it's really up to you depending on your project)
    Serial.begin(38400);

    // initialize device
    Serial.println("Initializing I2C devices...");
    accelgyro.initialize();

    // verify connection
    Serial.println("Testing device connections...");
    Serial.println(accelgyro.testConnection() ? "MPU6050 connection
successful" : "MPU6050 connection failed");

    // configure Arduino LED for
    pinMode(LED_PIN, OUTPUT);
}

void loop() {
    // read raw accel/gyro measurements from device
    accelgyro.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);

    // these methods (and a few others) are also available
    //accelgyro.getAcceleration(&ax, &ay, &az);
    //accelgyro.getRotation(&gx, &gy, &gz);

    // display tab-separated accel/gyro x/y/z values
    Serial.print("a/g:\t");
    Serial.print(ax); Serial.print("\t");
    Serial.print(ay); Serial.print("\t");
    Serial.print(az); Serial.print("\t");
    Serial.print(gx); Serial.print("\t");
    Serial.print(gy); Serial.print("\t");
    Serial.println(gz);

    // blink LED to indicate activity
    blinkState = !blinkState;
    digitalWrite(LED_PIN, blinkState);
}
```

MPU-9150 (I²C)

De MPU-9150 is een accelerometer, gyroscoop en een magnetometer in één en heeft dus negen vrijheidsgraden (3.3=9). De MPU-9150 is een combinatie van twee chips, de MPU-6050 voor de accelerometer en de gyro gecombineerd met een magnetometer. Deze chip geeft ook de mogelijkheid om de temperatuur te monitoren.

https://www.sparkfun.com/products/retired/11486

http://kingtidesailing.blogspot.be/2015/09/how-to-setup-mpu-9150-9-axis.html

Voor het gebruik van de MPU-9150 dienen I2Cdev.h, MPU9150.h, helper_3dmath.h en wire.h gebruikt te worden. De file helper_3dmath.h zit in de zip file van de helper_3dmath.h en zal bij het inbrengen van deze automatisch ook geïnstalleerd worden.

Met onderstaande voorbeeld code kan de MPU-9150 uitgelezen worden (temperatuur, accelerometer, magnetometer en gyroscoop). De AD0 pin dient aan 3.3V aangesloten te worden. er kan altijd voorbeeld code van op het net gebruikt worden maar deze gebruiken zeer vaak verkeerde registers, onderstaande code leest de juiste I²C registers uit en werd getest. De aansluiting van de MPU-9150 is de zelfde als de MPU-6050

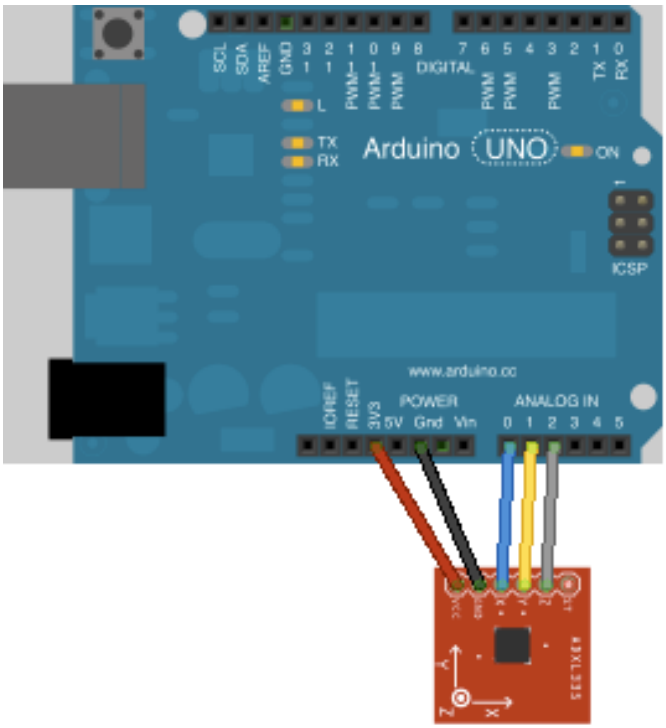
Voorbeeld code uitlezen van MPU-9150

```
// Arduino Wire library is required if I2Cdev I2CDEV_ARDUINO_WIRE
implementation
// is used in I2Cdev.h
#include "Wire.h"

// I2Cdev and MPU9150 must be installed as libraries, or else the .cpp/.h
files
// for both classes must be in the include path of your project
#include "I2Cdev.h"
#include "MPU9150.h"
#include "helper_3dmath.h"

// class default I2C address is 0x68
// specific I2C addresses may be passed as a parameter here
// AD0 low = 0x68 (default for InvenSense evaluation board)
// AD0 high = 0x69
MPU9150 accelGyroMag;

int16_t ax, ay, az;
int16_t gx, gy, gz;
int16_t mx, my, mz;
int16_t temp;

#define LED_PIN 13
bool blinkState = false;



void setup() {
    // join I2C bus (I2Cdev library doesn't do this automatically)
    Wire.begin();

    // initialize serial communication
    // (38400 chosen because it works as well at 8MHz as it does at 16MHz,
but
```

```
    // it's really up to you depending on your project)
    Serial.begin(115200);

    // initialize device
    Serial.println("Initializing I2C devices...");
    accelGyroMag.initialize();

    // verify connection
    Serial.println("Testing device connections...");
    Serial.println(accelGyroMag.testConnection() ? "MPU9150 connection
successful" : "MPU9150 connection failed");

    // configure Arduino LED for
    pinMode(LED_PIN, OUTPUT);
}

void loop() {

    // read raw accel/gyro/mag measurements from device
    accelGyroMag.getMotion9(&ax, &ay, &az, &gx, &gy, &gz, &mx, &my, &mz);
    double dT = ( (double) accelGyroMag.getTemperature() + 12412.0) /
340.0;

    // Print values

    Serial.print(dT); Serial.print("\t");

    Serial.print(ax); Serial.print("\t");
    Serial.print(ay); Serial.print("\t");
    Serial.print(az); Serial.print("\t | ");

    Serial.print(gx); Serial.print("\t");
    Serial.print(gy); Serial.print("\t");
    Serial.print(gz); Serial.print("\t | ");

    Serial.print((mx)); Serial.print("\t");
    Serial.print((my)); Serial.print("\t");
    Serial.print((mz)); Serial.print("\t | ");
    Serial.print("\n");

}
```

MMA7361 (Analoog)

De MMA7361 is een accelerometer met drie vrijheidsgraden, deze is analoog uit leesbaar in tegenstelling tot de bovenstaande modellen. Hierdoor zijn er geen extra labraries en dergelijke vereist bij het gebruik van deze component, wel zijn er meer aansluitingen vereist.



Voorbeeld code

```
const int Sleep=2;
void setup()
{
  Serial.begin(9600); // 9600 bps
  pinMode(Sleep, OUTPUT);
  digitalWrite(Sleep, HIGH);
}
void loop()
{
  int x,y,z;
  x=analogRead(0);
  y=analogRead(1);
  z=analogRead(2);
  Serial.print("x= ");
  Serial.print(x ,DEC);
  Serial.print(',');
  Serial.print("y= ");
  Serial.print(y ,DEC);
  Serial.print(',');
  Serial.print("z= ");
  Serial.println(z ,DEC);
  delay(100);
}
```