

Domotica

- Introductie
 - Arduino en IDE
 - Eerste programma
- Digitale inputs en outputs
 - Werken met een LED
 - Werken met een knop
- Analoge inputs en outputs
 - Analoge signalen
- Werken met sensoren
 - Sensoren
- Actuatoren en motoren
 - Wat zijn actuatoren?
 - Werken met servomotoren
 - Servo besturen met een potentiometer
- Expert: Communicatie met externe apparaten
 - Wat is seriële communicatie en i2c communicatie?
 - i2c scanner
 - Voorbeeld: gebruik van een BMP280 temperatuur- en druksensor

Introductie

Aan het einde van dit hoofdstuk kun je:

- Uitleggen wat een Arduino is en waarvoor het gebruikt kan worden.
- De Arduino IDE installeren op je computer.
- Je eerste Arduino-programma schrijven en uploaden naar de Arduino.

Arduino en IDE

ARDUINO_UNO_A06-3102750200.png

Wat is Arduino?

Arduino is een open-source elektronica platform gebaseerd op gebruiksvriendelijke hardware en software. Het is ontworpen voor artiesten, ontwerpers, hobbyisten en iedereen die geïnteresseerd is in het creëren van interactieve projecten.

Waarom Arduino?

- **Gemakkelijk te gebruiken:** Simpele programmeertaal en veel online bronnen.
- **Veelzijdig:** Kan worden gebruikt voor een breed scala aan projecten.
- **Community:** Grote gemeenschap die projecten en code deelt.

Installeren van de Arduino IDE

De Arduino IDE (Integrated Development Environment) is de software waarmee je programma's (ook wel "sketches" genoemd) voor je Arduino kunt schrijven en uploaden.

Stappen:

1. **Downloaden:** Ga naar www.arduino.cc/en/software en download de juiste versie voor jouw besturingssysteem (Windows, macOS, Linux).
2. **Installeren:** Volg de installatie-instructies op het scherm.
3. **Start de IDE:** Open de Arduino IDE nadat de installatie is voltooid.

image.png

Verkennen van de Arduino IDE

- **Menu's:** Bestand, Bewerken, Schets, Hulpmiddelen, Help.
- **Knoppen:** Verifiëren, Uploaden, Nieuw, Openen, Opslaan, Seriële monitor.
- **Codegebied:** Hier schrijf je je code.
- **Meldingen:** Foutmeldingen en statusupdates verschijnen hier.

Eerste programma

Je eerste programma: "Blink"

Het "Blink" voorbeeld laat een LED op de Arduino-board knipperen.

Stappen:

1. **Hardware voorbereiden:** Sluit je Arduino aan op de computer met de USB-kabel.
2. **Board selecteren:** Ga in de IDE naar **Hulpmiddelen > Board** en selecteer jouw Arduino-model (bijv. Arduino Uno).
3. **Poort selecteren:** Ga naar **Hulpmiddelen > Poort** en selecteer de poort waaraan je Arduino is verbonden.
4. **Voorbeeld openen:** Ga naar **Bestand > Voorbeelden > 01.Basics > Blink**.
5. **Code begrijpen:** Bekijk de code en lees de commentaarregels om te begrijpen wat er gebeurt.
6. **Uploaden:** Klik op de "Uploaden" knop om de code naar je Arduino te sturen.
7. **Observeer:** De ingebouwde LED (meestal verbonden aan pin 13) zal beginnen te knipperen.

Aanpassen van de knipperfrequentie

Laten we de snelheid waarmee de LED knippert aanpassen.

Stappen:

1. **Code aanpassen:** Zoek de regels met `delay(1000);`.
2. **Verander de waarden:** Pas de vertraging aan naar `delay(500);` voor een snellere knippering.
3. **Uploaden:** Stuur de aangepaste code opnieuw naar de Arduino.
4. **Observeer:** De LED zou nu sneller moeten knipperen.

Opdracht

Doel: Pas het "Blink" programma aan zodat de LED in een ander tempo knippert.

Instructies:

1. Verander de `delay` waarden om verschillende knipperpatronen te creëren.
2. Experimenteer met zeer korte en lange vertragingen.
3. Voeg commentaar toe aan je code om uit te leggen wat je hebt aangepast.
4. Sla je aangepaste sketch op onder een nieuwe naam.

Extra Uitdaging

- Voeg een tweede LED toe aan je breadboard en laat deze in een ander tempo knipperen.
- Tip: Gebruik digitale pinnen en onthoud dat je een weerstand moet gebruiken om je LED te beschermen.

Digitale inputs en outputs

Aan het einde van dit hoofdstuk kun je:

- Uitleggen wat digitale signalen zijn.
- Werken met digitale outputs (zoals LED's) en inputs (zoals knoppen).
- Een programma schrijven dat reageert op gebruikersinvoer via een knop.

Werken met een LED

Digitale Signalen

- **Digitaal:** Signalen met slechts twee mogelijke toestanden: **hoog** (HIGH) of **laag** (LOW), oftewel aan of uit.
- **Gebruik:** Digitale signalen worden gebruikt om apparaten zoals LED's en knoppen te bedienen of uit te lezen.

Werken met LED's (Digitale Outputs)

LED aansluiten:

1. **Componenten:** LED, 220Ω weerstand, breadboard, jumper wires.
2. **Aansluiting:**
 - Verbind de **anode** (+, lange poot) van de LED via de weerstand met digitale pin 9 van de Arduino.
 - Verbind de **kathode** (-, korte poot) van de LED met de GND (ground) van de Arduino.

Code om LED te laten branden:

```
void setup() {  
  pinMode(9, OUTPUT); // Zet pin 9 als output  
}  
  
void loop() {  
  digitalWrite(9, HIGH); // Zet pin 9 hoog (LED aan)  
}
```

Werken met een knop

Knop aansluiten:

1. **Componenten:** Drukknop, 10kΩ weerstand (pull-down), breadboard, jumper wires.
2. **Aansluiting:**
 - Verbind één kant van de knop met 5V.
 - Verbind de andere kant van de knop met digitale pin 7 en met een 10kΩ weerstand naar GND.

Code om knop uit te lezen:

```
void setup() {  
  pinMode(7, INPUT); // Zet pin 7 als input  
  pinMode(9, OUTPUT); // LED pin  
}  
  
void loop() {  
  int knopStatus = digitalRead(7); // Lees de status van de knop  
  
  if (knopStatus == HIGH) {  
    digitalWrite(9, HIGH); // Zet LED aan  
  } else {  
    digitalWrite(9, LOW); // Zet LED uit  
  }  
}
```

Uitleg van de Code

- **pinMode():** Stelt de pin in als INPUT of OUTPUT.
- **digitalRead():** Leest de status van een digitale inputpin (HIGH of LOW).
- **digitalWrite():** Zet een digitale outputpin op HIGH of LOW.
- **if-else structuur:** Bepaalt welk deel van de code wordt uitgevoerd op basis van een voorwaarde.

Opbouw van de Schakeling

- Zorg ervoor dat alle verbindingen stevig zijn.
- Controleer of de weerstand op de juiste plaats zit om kortsluiting te voorkomen.
- Gebruik een schema als referentie.

Veelgemaakte Fouten

- **Verkeerde aansluiting van de LED:** Let op de anode en kathode.
- **Weerstand vergeten:** Dit kan leiden tot het doorbranden van de LED.
- **Pull-down weerstand missen:** Zonder deze kan de input "zweven" en onvoorspelbaar gedrag vertonen.

Veiligheidstips

- Controleer altijd je schakeling voordat je de stroom inschakelt.
- Gebruik de juiste weerstandswaarden om componenten te beschermen.
- Raak de Arduino niet aan op plekken waar spanning aanwezig is tijdens gebruik.

Analoge inputs en outputs

Aan het einde van dit hoofdstuk kun je:

- Uitleggen wat analoge signalen zijn.
- Werken met analoge inputs (zoals potentiometers) en outputs (zoals LED-helderheid aanpassen met PWM).
- Een programma schrijven dat een analoge sensor uitleest en daarop reageert.

Analoge signalen

image.png

Analoge Signalen

- **Analoge Signalen:** Signalen die continu variëren en een oneindig aantal waarden kunnen aannemen binnen een bereik.
- **Voorbeelden:** Geluidsgolven, lichtintensiteit, temperatuur.

Verschil met Digitale Signalen:

- **Digitaal:** Discrete waarden (aan of uit, 0 of 1).
- **Analoog:** Continu variërende waarden.

Analoge Inputs: Potentiometers

Wat is een potentiometer?

- Een variabele weerstand die je kunt draaien om de weerstand te veranderen.
- Wordt gebruikt om analoge input te genereren (bijv. volumeknop op een radio).

Aansluiten van een potentiometer:

1. **Componenten:** Potentiometer, breadboard, jumper wires.
2. **Aansluiting:**
 - Verbind de middelste pin van de potentiometer met analoge pin A0 van de Arduino.
 - Verbind één van de buitenste pinnen met 5V.
 - Verbind de andere buitenste pin met GND.

Analoge Outputs: LED-helderheid met PWM

Wat is PWM (Pulse Width Modulation)?

- Een techniek om analoge resultaten te verkrijgen met digitale middelen.

- Door de tijd dat een signaal hoog is te variëren (duty cycle), kunnen we de gemiddelde stroom naar een apparaat regelen.

LED-helderheid aanpassen:

- Door PWM kunnen we de helderheid van een LED regelen.
- Arduino heeft speciale pinnen die PWM ondersteunen (gemarkeerd met een ~, bijv. pin 9).

Aansluiten van de LED:

1. **Componenten:** LED, 220Ω weerstand, breadboard, jumper wires.
2. **Aansluiting:**
 - Verbind de anode (+) van de LED via de weerstand met PWM pin 9.
 - Verbind de kathode (-) van de LED met GND.

Code om de LED-helderheid te regelen met een potentiometer

Stap-voor-stap uitleg:

1. **Lees de waarde van de potentiometer:**

```
int sensorValue = analogRead(A0);
```

2. **Converteer de sensorwaarde naar een PWM-waarde:**

- De `analogRead()` geeft een waarde tussen 0 en 1023.
- De `analogWrite()` verwacht een waarde tussen 0 en 255.
- We gebruiken de `map()` functie om dit om te zetten.

```
int outputValue = map(sensorValue, 0, 1023, 0, 255);
```

3. **Schrijf de PWM-waarde naar de LED:**

```
analogWrite(9, outputValue);
```

Volledige code:

```
void setup() {  
  // Geen setup nodig voor dit voorbeeld  
}
```

```
void loop() {  
    int sensorValue = analogRead(A0);          // Lees de potentiometer  
    int outputValue = map(sensorValue, 0, 1023, 0, 255); // Converteer naar PWM-waarde  
    analogWrite(9, outputValue);               // Pas LED-helderheid aan  
}
```

Uitleg van de Code

- **analogRead(pin):** Leest de analoge waarde van de gegeven pin.
- **map(value, fromLow, fromHigh, toLow, toHigh):** Converteert een waarde van het ene bereik naar het andere.
- **analogWrite(pin, value):** Schrijft een PWM-sigitaal naar de gegeven pin.

Opdracht

Doel: Maak een LED-dimmer met behulp van een potentiometer.

Instructies:

1. Bouw de schakeling zoals hierboven beschreven.
2. Schrijf de code over in de Arduino IDE.
3. Upload de code naar je Arduino.
4. Draai aan de potentiometer en observeer hoe de helderheid van de LED verandert.

Extra Uitdagingen

- **Uitbreiding 1:** Voeg een tweede LED toe en laat deze tegengesteld dimmen ten opzichte van de eerste LED.
 - **Hint:** Gebruik `outputValue = 255 - outputValue;` voor de tweede LED.
- **Uitbreiding 2:** Gebruik een LDR (lichtsensor) in plaats van een potentiometer om de LED-helderheid automatisch aan te passen aan het omgevingslicht.

Veelgemaakte Fouten

- **Verkeerde pinnen gebruiken:** Zorg ervoor dat je de juiste analoge en PWM-pinnen gebruikt.
- **Vergeten te mappen:** Zonder het omzetten van de sensorwaarde naar het juiste bereik werkt de PWM niet correct.
- **LED direct op 5V aansluiten:** Dit kan de LED beschadigen; gebruik altijd een weerstand.

Veiligheidstips

- Controleer je schakeling voordat je stroom toevoert.
- Gebruik de juiste weerstanden om overbelasting te voorkomen.
- Raak geen blote draad aan tijdens het gebruik van de schakeling.

Werken met sensoren

Aan het einde van dit hoofdstuk kun je:

- Verschillende soorten sensoren identificeren en begrijpen hoe ze werken.
- Sensorwaarden uitlezen met de Arduino.
- Een programma schrijven dat reageert op sensordata.

Sensoren

Introductie tot Sensoren

- **Wat zijn sensoren?:** Apparaten die fysieke grootheden (zoals licht, temperatuur, afstand) meten en deze omzetten in elektrische signalen.
- **Veelgebruikte sensoren:**
 - **LDR (Lichtgevoelige Weerstand):** Meet lichtintensiteit.
 - **Temperatuursensor (bijv. LM35, DHT11):** Meet temperatuur (en soms luchtvochtigheid).
 - **Ultrasonische sensor (HC-SR04):** Meet afstand tot een object.

Werken met een Lichtsensor (LDR)

Hoe werkt een LDR?

- De weerstand van een LDR verandert met de hoeveelheid licht die erop valt.
- Meer licht = lagere weerstand; minder licht = hogere weerstand.

Aansluiten van een LDR:

1. **Componenten:** LDR, 10kΩ weerstand, breadboard, jumper wires.
2. **Aansluiting:**
 - Verbind één kant van de LDR met 5V.
 - Verbind de andere kant van de LDR met analoge pin A0 en met een 10kΩ weerstand naar GND (vormt een spanningsdeler).

Code om de LDR uit te lezen

Stap-voor-stap uitleg:

1. **Lees de waarde van de LDR:**

```
int sensorValue = analogRead(A0);
```

2. **Toon de waarde op de seriële monitor:**

```
Serial.println(sensorValue);
```

Volledige code:

```
void setup() {  
  Serial.begin(9600); // Start seriële communicatie  
}  
  
void loop() {  
  int sensorValue = analogRead(A0); // Lees de LDR  
  Serial.println(sensorValue);    // Toon de waarde  
  delay(500);                    // Wacht een halve seconde  
}
```

Gebruik van de Seriële Monitor:

- Open de seriële monitor in de Arduino IDE via **Hulpmiddelen > Seriële Monitor**.
- Stel de baudrate in op 9600 baud.

LED automatisch aanzetten bij weinig licht

Uitbreiding van de code:

1. Stel een drempelwaarde in:

```
int drempel = 500; // Pas deze waarde aan op basis van je metingen
```

2. Gebruik een if-else statement om de LED te sturen:

```
if (sensorValue < drempel) {  
  digitalWrite(9, HIGH); // Zet LED aan  
} else {  
  digitalWrite(9, LOW); // Zet LED uit  
}
```

Volledige code met LED-besturing:

```
void setup() {  
  Serial.begin(9600);  
  pinMode(9, OUTPUT); // Zet pin 9 als output voor de LED  
}
```

```
void loop() {  
  int sensorValue = analogRead(A0);  
  Serial.println(sensorValue);  
  
  int drempel = 500; // Stel je eigen drempelwaarde in  
  
  if (sensorValue < drempel) {  
    digitalWrite(9, HIGH); // LED aan  
  } else {  
    digitalWrite(9, LOW); // LED uit  
  }  
  
  delay(500);  
}
```

Uitleg van de Code

- **Serial.begin(9600):** Start seriële communicatie op 9600 baud.
- **Serial.println():** Stuurt data naar de seriële monitor.
- **if (sensorValue < drempel):** Controleert of de gemeten waarde onder de drempel ligt.

Opdracht

Doel: Laat een LED automatisch aangaan wanneer het donker wordt.

Instructies:

1. Bouw de schakeling met de LDR en LED zoals hierboven beschreven.
2. Schrijf of kopieer de code in de Arduino IDE.
3. Upload de code naar je Arduino.
4. Gebruik je hand of een voorwerp om licht op de LDR te blokkeren en observeer hoe de LED reageert.
5. Pas indien nodig de drempelwaarde aan voor optimale werking.

Extra Uitdagingen

- **Uitbreiding 1:** Voeg een buzzer toe die een geluid maakt wanneer het volledig donker is.
 - **Hint:** Gebruik een extra if-statement met een lagere drempelwaarde.
- **Uitbreiding 2:** Gebruik een RGB LED om verschillende kleuren weer te geven op basis van de lichtintensiteit.

Veelgemaakte Fouten

- **Verkeerde aansluiting van de LDR:** Zorg ervoor dat de spanningsdeler correct is aangesloten.
- **Drempelwaarde niet aanpassen:** De standaard drempelwaarde werkt mogelijk niet in alle lichtomstandigheden; pas deze aan op basis van je metingen.
- **Seriële monitor niet geopend:** Vergeet niet de seriële monitor te openen om de sensorwaarden te zien.

Veiligheidstips

- Wees voorzichtig met het aansluiten van de componenten om kortsluiting te voorkomen.
- Gebruik de juiste weerstanden om je componenten te beschermen.
- Werk in een goed verlichte omgeving om fouten te voorkomen.

Actuatoren en motoren

Aan het eind van dit hoofdstuk kun je:

- Uitleggen wat actuatoren zijn en hun rol in elektronische systemen.
- Een servomotor aansluiten en besturen met de Arduino.
- Een programma schrijven dat een servomotor aanstuurt op basis van sensorinput.

Wat zijn actuatoren?

image.png

Definitie: Een actuator is een apparaat dat energie omzet in beweging. Het wordt gebruikt om een systeem of mechanisme te controleren of te bewegen.

Voorbeelden van actuatoren:

- **Servomotoren:** Precisie motoren die een hoekpositie kunnen aannemen.
- **DC-motoren:** Standaard motoren die continu kunnen draaien.
- **Stepper motoren:** Motoren die in discrete stappen bewegen.
- **Relais:** Elektrische schakelaars die circuits kunnen openen of sluiten.

Waarom actuatoren gebruiken?

- Ze stellen je in staat om fysieke beweging te integreren in je projecten.
- Worden gebruikt in robotica, automatisering, en mechanische systemen.

Werken met servomotoren

Wat is een Servomotor?

- Een servomotor is een motor die een bepaalde hoekpositie kan aannemen, meestal tussen 0° en 180°.
- Hij bevat een motor, een tandwielstelsel, een potentiometer en een bestuurscircuit.

Aansluiten van een Servomotor:

1. **Componenten:** Servomotor (bijv. SG90), Arduino, breadboard, jumper wires.
2. **Draden van de servo:**
 - **Rood:** Voedingsspanning (meestal 5V).
 - **Bruin of Zwart:** Aarde (GND).
 - **Geel, Oranje of Wit:** Signaal (PWM-besturingssignaal).
3. **Aansluiting:**
 - Verbind de **rode draad** met de 5V pin van de Arduino.
 - Verbind de **bruine/zwart draad** met GND.
 - Verbind de **geel/oranje/witte draad** met digitale pin 9 (PWM pin).

Let op: Sommige servomotoren kunnen meer stroom trekken dan de Arduino kan leveren. Voor dit voorbeeld is het veilig, maar voor grotere servo's moet je een externe voeding gebruiken.

De Servo Bibliotheek

- Arduino heeft een ingebouwde **Servo** bibliotheek die het eenvoudig maakt om servomotoren aan te sturen.
- De bibliotheek zorgt voor de juiste timing van de PWM-signalen die nodig zijn om de servo te besturen.

Toevoegen van de Servo Bibliotheek:

- Ga in de Arduino IDE naar **Schets > Bibliotheek gebruiken > Servo** of voeg `#include <Servo.h>` toe aan je code.

Code om de Servomotor te Besturen

Basiscode om de servo te bewegen:

```
#include <Servo.h>

Servo mijnServo; // Maak een Servo object aan

void setup() {
  mijnServo.attach(9); // Verbind de servo met pin 9
}

void loop() {
  mijnServo.write(0); // Beweeg naar 0 graden
  delay(1000);        // Wacht 1 seconde
  mijnServo.write(90); // Beweeg naar 90 graden
  delay(1000);
  mijnServo.write(180); // Beweeg naar 180 graden
  delay(1000);
}
```

Uitleg van de Code:

- `#include <Servo.h>`: Voegt de Servo bibliotheek toe.
- `Servo mijnServo;`: Maakt een nieuw Servo object.
- `mijnServo.attach(9);`: Verbindt het Servo object met pin 9.
- `mijnServo.write(hoek);`: Stuur de servo naar de opgegeven hoek (0° - 180°).

Servo besturen met een potentiometer

Sensorinput gebruiken om de servo te besturen:

1. Aansluiting van de potentiometer:

- Verbind de middelste pin van de potentiometer met analoge pin A0.
- Verbind de buitenste pinnen met 5V en GND.

2. Code aanpassen:

```
#include <Servo.h>

Servo mijnServo;

void setup() {
  mijnServo.attach(9);
}

void loop() {
  int sensorValue = analogRead(A0);          // Lees de potentiometer
  int hoek = map(sensorValue, 0, 1023, 0, 180); // Converteer naar hoek
  mijnServo.write(hoek);                     // Beweeg servo naar hoek
  delay(15);                                // Kleine pauze voor stabiliteit
}
```

Uitleg van de Code:

- `analogRead(A0)`: Leest de waarde van de potentiometer.
- `map(sensorValue, 0, 1023, 0, 180)`: Converteert de sensorwaarde naar een hoek tussen 0° en 180°.
- `mijnServo.write(hoek)`: Beweegt de servo naar de berekende hoek.

Opdracht

Doel: Programmeer de servomotor om heen en weer te bewegen op basis van een sensorinput (potentiometer).

Instructies:

1. Bouw de schakeling met de servomotor en potentiometer zoals hierboven beschreven.
2. Schrijf of kopieer de code in de Arduino IDE.
3. Upload de code naar je Arduino.
4. Draai aan de potentiometer en observeer hoe de servomotor reageert.

Extra Uitdagingen

- **Uitbreiding 1:** Gebruik een LDR in plaats van een potentiometer om de servo te besturen op basis van lichtintensiteit.
- **Uitbreiding 2:** Laat de servo automatisch heen en weer bewegen tussen twee hoeken met een instelbare snelheid.

Veelgemaakte Fouten

- **Verkeerde voeding van de servo:** Zorg ervoor dat de servo voldoende stroom krijgt; gebruik eventueel een externe voeding.
- **Servo niet correct aangesloten:** Controleer de kleurcodering van de draden.
- **Servo beweegt schokkerig:** Voeg een delay toe om de beweging te stabiliseren.

Veiligheidstips

- Gebruik geen servomotoren die meer stroom vereisen dan de Arduino kan leveren zonder externe voeding.
- Zorg ervoor dat bewegende delen vrij kunnen bewegen zonder obstakels.
- Pas op voor scherpe randen of bewegende onderdelen die letsel kunnen veroorzaken.

Expert: Communicatie met externe apparaten

Aan het einde van dit hoofdstuk kun je:

- Uitleggen wat seriële en I2C-communicatie zijn en hoe ze werken met de Arduino.
- Data versturen naar en ontvangen van de computer via de seriële monitor.
- Sensoren en apparaten aansluiten die gebruikmaken van het I2C-protocol.
- Een programma schrijven dat sensorwaarden uitleest via I2C en weergeeft op de computer.

Wat is seriële communicatie en i2c communicatie?

image.png

Wat is Seriële Communicatie?

- **Definitie:** Een methode om data één bit per keer te verzenden over een communicatiekanaal.
- **Toepassing:** Veel gebruikt voor communicatie tussen microcontrollers en computers of andere apparaten.
- **Seriële Monitor:** Een ingebouwd hulpmiddel in de Arduino IDE om data te verzenden en ontvangen via de seriële poort.

De Seriële Monitor Gebruiken

- **Openen:** Ga in de Arduino IDE naar **Hulpmiddelen > Seriële Monitor** of klik op het vergrootglas-icoon.
- **Baudrate instellen:** Zorg ervoor dat de baudrate in de seriële monitor overeenkomt met die in je code (bijv. 9600 baud).

Wat is I2C-Communicatie?

- **Definitie:** I2C (Inter-Integrated Circuit) is een synchrone, multi-master, multi-slave, seriële communicatiebus.
- **Kenmerken:**
 - **Twee draden:** SDA (data) en SCL (klok).
 - **Adressering:** Elk apparaat op de bus heeft een uniek adres.
- **Toepassing:** Veel gebruikt voor communicatie met sensoren, displays, en andere randapparatuur.

Sensoren en Apparaten met I2C

- **Voorbeelden:**

- **BMP280/BME280:** Druk- en temperatuursensoren.
- **OLED-displays:** Kleine schermen voor tekst en grafieken.
- **RTC (Real-Time Clock) modules:** Voor tijdregistratie.
- **GY-521 (MPU-6050):** Versnellingsmeter en gyroscoop.

i2c scanner

I2C-Scanner

Voordat je met I2C-apparaten werkt, is het handig om te weten welk adres het apparaat heeft.

I2C-Scanner Code:

```
#include <Wire.h>

void setup() {
  Wire.begin();
  Serial.begin(9600);
  Serial.println("I2C Scanner gestart");
}

void loop() {
  byte error, address;
  int nDevices = 0;

  Serial.println("Apparaten zoeken...");

  for(address = 1; address < 127; address++ ) {
    Wire.beginTransmission(address);
    error = Wire.endTransmission();

    if (error == 0) {
      Serial.print("I2C apparaat gevonden op adres 0x");
      if (address<16)
        Serial.print("0");
      Serial.println(address, HEX);
      nDevices++;
    } else if (error==4) {
      Serial.print("Onbekende error op adres 0x");
      if (address<16)
```

```
        Serial.print("0");
        Serial.println(address, HEX);
    }
}
if (nDevices == 0)
    Serial.println("Geen I2C apparaten gevonden\n");
else
    Serial.println("Klaar\n");

delay(5000);
}
```

Instructies:

1. Sluit je I2C-apparaat aan op de Arduino.
2. Verbind SDA met A4 en SCL met A5 (Arduino Uno).
3. Upload de I2C-scanner code.
4. Open de seriële monitor om het I2C-adres van je apparaat te zien.

Voorbeeld: gebruik van een BMP280 temperatuur- en druksensor

Componenten:

- BMP280 module
- Arduino Uno
- Breadboard en jumper wires

Aansluiting:

- **VCC** van de sensor naar 3.3V op de Arduino
- **GND** naar GND
- **SDA** naar A4
- **SCL** naar A5

Let op: De BMP280 werkt op 3.3V; sluit deze niet aan op 5V.

Bibliotheek Installeren:

- Ga naar **Schets > Bibliotheek gebruiken > Bibliotheken beheren**.
- Zoek naar "Adafruit BMP280" en installeer deze bibliotheek.

Code om de BMP280 uit te lezen:

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BMP280.h>

Adafruit_BMP280 bmp; // I2C

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

  if (!bmp.begin(0x76)) { // Gebruik het adres dat je met de I2C-scanner hebt gevonden
```

```
Serial.println("BMP280 niet gevonden!");  
while (1);  
}  
}  
  
void loop() {  
  Serial.print("Temperatuur = ");  
  Serial.print(bmp.readTemperature());  
  Serial.println(" °C");  
  
  Serial.print("Druk = ");  
  Serial.print(bmp.readPressure()/100.0F);  
  Serial.println(" hPa");  
  
  Serial.println();  
  delay(2000);  
}
```

Uitleg van de Code:

- `Adafruit_BMP280 bmp;` : Maakt een nieuw BMP280 object aan.
- `bmp.begin(0x76);` : Initialiseert de sensor op het gegeven I2C-adres.
- `bmp.readTemperature()` : Leest de temperatuur uit.
- `bmp.readPressure()` : Leest de luchtdruk uit.

Data Versturen via Seriële en I2C-Communicatie

- **Serieel:** Data wordt naar de seriële monitor gestuurd voor weergave.
- **I2C:** Data wordt via de I2C-bus opgehaald van de sensor.

Opdracht

Doel: Schrijf een programma dat temperatuur- en drukwaarden uitleest van een BMP280 sensor via I2C en deze weergeeft op de computer via de seriële monitor.

Instructies:

1. Bouw de schakeling met de BMP280 sensor zoals hierboven beschreven.
2. Installeer de benodigde bibliotheken.

3. Gebruik de I2C-scanner om het adres van je sensor te bepalen.
4. Pas de code aan met het juiste adres indien nodig.
5. Upload de code naar je Arduino.
6. Open de seriële monitor en observeer de temperatuur- en drukwaarden.
7. Experimenteer met het veranderen van de omgeving (bijv. verwarmen van de sensor) en observeer de veranderingen.

Extra Uitdagingen

- **Uitbreiding 1:** Voeg een OLED-display toe (I2C) en toon de sensorwaarden op het scherm.
Hint: Gebruik de **Adafruit SSD1306** bibliotheek.
- **Uitbreiding 2:** Sluit meerdere I2C-apparaten aan op dezelfde bus en lees data uit van elk apparaat.

Belangrijk: Alle I2C-apparaten worden parallel aangesloten op SDA en SCL. Zorg ervoor dat elk apparaat een uniek adres heeft.

Veelgemaakte Fouten

- **Verkeerde aansluitingen:** Controleer altijd de SDA en SCL pinnen; op een Arduino Uno zijn dit A4 en A5.
- **Verkeerde voedingsspanning:** Sommige sensoren werken op 3.3V in plaats van 5V.
- **Bibliotheken niet geïnstalleerd:** Vergeet niet de benodigde bibliotheken te installeren.

Veiligheidstips

- Controleer de datasheet van je sensor voor de juiste voedingsspanning.
- Sluit geen 3.3V sensoren aan op 5V; dit kan de sensor beschadigen.
- Gebruik pull-up weerstanden op de SDA en SCL lijnen als je communicatieproblemen ondervindt (meestal 4.7kΩ).