THOMAS MORE

Robot grasmaaier cursus

Dylan Geldhof
&
Mathias Vermeulen

2024 - 2025

Inhoudstafel

1. Inleiding Project

Extra info ook te vinden op Github: https://github.com/DylanGhf/TM_Robot_Mower
Daar kan je alle code en schema’s terugvinden

1.1.1. Motor Driver

Voor dit project werd een H-brug ontworpen om de motoren aan te sturen. Deze H-brug bestaat uit MOSFET's en is aangesloten op een microcontroller via een I²C-bus. Er werd echter gekozen voor de verkeerde soort MOSFET's voor de high-side switches: NPN-MOSFET's in plaats van PNP-MOSFET's. Het ontwerp werkte wel voor het aansteken van de motoren, maar het was niet mogelijk om voldoende spanning door te laten, omdat de gate-spanning van de high-side MOSFET's niet hoog genoeg was (5V i.p.v. meer dan 12V). Om dit op te lossen, werd een kant-en-klare H-brug gebruikt, maar het zou handig zijn geweest om de eigen module te gebruiken. De motor driver bevat ook twee IC's voor het meten van het stroomverbruik en de spanning voor elke motor.

1.1.2. Power-board

Het power-board zet de batterijspanning om naar 5V en 3.3V met behulp van LM1117T IC's. Het bord is ontworpen om gevoed te worden door een 3S LIPO-batterij. Om de cellen van de batterij afzonderlijk uit te lezen, is er een JST-poort voorzien, maar deze is momenteel nog niet bruikbaar, aangezien de spanning te hoog is om direct door de microcontroller (Attiny) uit te lezen. Een mogelijke oplossing is het gebruik van spanningsdelers om de spanning binnen het bereik van de Attiny te brengen. De Attiny op het power-board kan via de I²C-bus communiceren met andere modules, zoals een ESP32.

1.1.3. MCU (Microcontroller Unit)

Dit board is het brein van het systeem. Hier worden de signalen van de sensoren ontvangen en de actuatoren aangestuurd. De gebruikte microcontroller is de ESP32S3 Wroom-1-N8R8, die standaard wordt gebruikt in ESP32S3 ontwikkelboards. In het ontwerp was oorspronkelijk voorzien om een aangepaste H-brug via I²C aan te sturen, maar er waren geen aansluitingen voor de standaard H-brug. Er zijn echter extra IO-pinnen voorzien die gebruikt konden worden om de H-brug rechtstreeks aan de ESP32 aan te sluiten.

Alle communicatie tussen de microcontroller en de andere componenten zoals het powerboard, HC-SR04-ultrasonische sensoren, enz., wordt geregeld via een I²C-bus. Deze bus maakt het mogelijk om berichten te versturen en ontvangen van aangesloten apparaten. De busconnector heeft zowel 5V- als 3.3V-pinnen om sensoren van verschillende spanningen van stroom te voorzien. De SCL- en SDA-lijnen van de bus hebben pull-ups naar 3.3V. Het is belangrijk om deze lijnen **nooit** met 5V te verbinden, omdat dit kan leiden tot een kortsluiting van de ESP32. Aangezien I²C werkt met een pulldown om een digitale 1 door te geven, is dit geen

probleem zolang de aangesloten apparaten een digitale 1 op een lager voltage dan 3.3V kunnen lezen.

Dit board bevat ook een USB-B-aansluiting voor het uploaden van programma's naar de ESP32 en voor het gebruik van de seriële monitor. Daarnaast is er een SPI-connector voorzien, bijvoorbeeld om een SD-kaart te gebruiken voor dataverwerking.

1.1.4. HC-SR04 Ultrasonic Sensor

Dit board neemt de leeswerkdruk van de microcontroller door 3 HC-SR04 afstandssensoren in te lezen via een Attiny. Dit maakt het programma op de ESP32 kleiner en verlaagt de werkdruk. De Attiny heeft geen interne timer, waardoor de sensoren niet op de gebruikelijke manier uitgelezen kunnen worden. Hiervoor wordt de "NewPing"-bibliotheek gebruikt, waarmee een timer op de Attiny wordt ingesteld. Wanneer de microcontroller een verzoek voor 11 bytes aan data doet, stuurt de Attiny deze geformatteerd door.

De PCB is eenvoudig en bevat alleen het noodzakelijke voor de Attiny om te functioneren, evenals de I²C-connectoren voor doorverbinding en de drie connectors voor de HC-SR04 sensoren. Hoewel de lay-out verbeterd zou kunnen worden, wordt het ontwerp als geslaagd beschouwd.

2. Inleiding i2c

I2C (inter-integrated ciruit) is een populair communicatieprtocol dat wordt gebruikt om apparaten zoals sensoren en displays met elkaar te laten communiceren. Dit wordt gedaan via een twee-draads bus systeem.

2.1. I2c

Het systeem bestaat uit twee draden (SDA en SCL). Dit zorgt voor bidirectionele buscommunicatie. De werking van de twee draden resulteert in lage kosten wat meteen een plus punt is voor projecten in de Iot.

Het protocol zorgt ervoor dat een master apparaat kan communiceren met een slave-apparaat. Hierbij levert de master de klok die de dataoverdrachtssnelheid of klokfrequentie wordt. Het is een bidirectionele bus, zodat de master naar de slave kan schrijven en van de slave kan lezen. Ook is het een seriële bus, wat betekent dat data een klok is en bit voor bit wordt verschoven. De SDA lijn verzendt en ontvangt gegevens. De SCL lijn bepaalt de timing van de communicatie. Beide lijnen hebben een pull-up weerstand nodig (4.7kOhm – 10kOhm). Zo blijft de lijn HIGH in rusttoestand. Lagere weerstand kan ook voor snelheden te verhogen.
Belangrijkste elementen:

- Twee draden (SDA en SCL)
- Ondersteunt meerdere masters en slaves op een bus
- Unieke adressen voor de slaves
- Werkt op verschillende snelheden
- Eenvoudig te implementeren

2.2. Communicatieproces

De communicatie tussen een master en slave vlogt volgens een vast patroon. Eerst begint de master de communicatie door de SDA-lijn laag te trekken terwijl de SCL-lijn hoog blijft. Als tweede stuurt de master een 7-bit adres van de slave gevolgd door een lees of schrijfbit (0 = schrijven, 1 = lezen). Daarna reageert de slave door de SDA-lijn laag te houden als bevestiging dat het bericht goed ontvangen is. In de vierde stap wordt data in blokken van 8 bits verzonden en als gevolg stuurt de ontvanger een acknowledge bit. Op het einde wordt de communicatie beïndigd door de SDA-lijn hoog te laten terwijl de SCL-lijn ook hoog blijft.

2.3. Adressering

Alle slaves op de bus hebben een eigen uniek adres. Deze is meestal 7-bits lang. Zo kan een master een of meerdere slaves aansturen. Sommige ondersteunen ook 10-bit adressen. Sensoren kunnen een vast adres hebben terwijl anderen een instelbaar andres kunnen hebben, dit kan dan ingesteld worden via jumpers.

Voordat je kan beginnen met i2c, moet je eerst het adres vinden. Dat kan gebeuren met onderstaande code. Zo krijg je op de seriële monitor een adres of de adressen van de aangesloten apparaten.

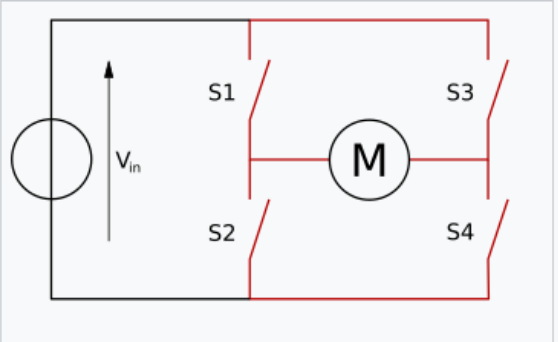
```
#include <Wire.h>

void setup() {
  Serial.begin(115200);
  Wire.begin();
  Serial.println("I2C Scanner gestart...");
}

void loop() {
  Serial.println("Scannen...");
  for (byte adres = 1; adres < 127; adres++) {
    Wire.beginTransmission(adres);
    if (Wire.endTransmission() == 0) {
      Serial.print("I2C-apparaat gevonden op adres 0x");
      Serial.println(adres, HEX);
    }
  }
  delay(5000);
}
```

3. Inleiding H-brug

Een H-brug is een veel gebruikte schakeling in electronica. Het wordt gebruikt om de draairichting en snelheid van gelijkstrommotoren (DC-motoren) te regelen. De schakelijk wordt vaak gebruikt bij robots en industriële automatisering.

3.1. Wat is een H-brug?

Deze schakeling is opgebouwd uit vier schakelelementen zoals transistoren en mosfets. Deze kunnen een motor aansturen en de stroomrichting veranderen. Dit is nodig bij de robotmaaier zodat deze van richting kan veranderen. De naam komt voor uit de typische schakelschema’s, waar de motor in het midden staan en de vier schakelelementen aan de zijkanten worden afgebeeld.



Structuur van een H-brug

3.2. Werking van een H-brug

De basiswerking van een H-brug is niet heel moeilijk. Door de juiste combinatie van schakelaars te openen en te sluiten, kan de stroom in twee richtingen door de motor gaan. Om bijvoorbeeld vooruit te gaan, worden S1 en S4 geschakelt. Om achteruit te draaien worden S2 en S3 geschakelt en staan de andere uit.

3.3. Opdracht H-brug PNP-MOSFETS

In deze opdracht gaan we een gloeilamp of led aan de hand van vier PNP-MOSFETS aansturen. De led zal dienen als vervanging voor de motor.

Wat gaan we doen?
We gaan een **H-brug schakeling** bouwen met **4 PNP-MOSFETs** die een **LED** aanstuurt. We zullen de H-brug gebruiken om de **richting van de spanning over de LED te veranderen**. Dit zal ervoor zorgen dat de LED **omkeert van licht naar uit** of **gedimd** afhankelijk van de richting van de stroom.

Wat is een MOSFET?
Een **MOSFET** is een elektronische schakelaar. Het werkt een beetje zoals een gewone schakelaar, maar in plaats van handmatig in- en uit te schakelen, wordt het elektronisch bestuurd. We gebruiken **PNP-MOSFETs** in deze oefening, en ze worden geactiveerd wanneer er een negatieve spanning op de **gate** wordt gezet ten opzichte van de **source** van de MOSFET.
Hoe werkt deze schakeling?

We gaan nu uitleggen hoe de stroom door de **LED** wordt gestuurd en hoe de richting van de spanning ervoor zorgt dat de LED **aan of uit gaat**.

Draairichting 1 (LED licht op):

- **Q1** en **Q4** worden ingeschakeld (aan), en de stroom kan van **Vcc** via **Q1** door de LED en naar **Q4** naar de grond vloeien.
- **Q2** en **Q3** worden uitgeschakeld (uit), dus er kan geen stroom door die MOSFETs vloeien.
- Dit zorgt ervoor dat de LED **licht geeft**.

Draairichting 2 (LED dimt of gaat uit):

- **Q2** en **Q3** worden ingeschakeld (aan), en de stroom kan van **GND** via **Q3** door de LED en naar **Q2** naar **Vcc** vloeien.
- **Q1** en **Q4** worden uitgeschakeld (uit), dus er kan geen stroom door die MOSFETs vloeien.
- Dit zorgt ervoor dat de LED **uitgaat** of minder fel licht geeft, afhankelijk van de stroom.

Hoe sluiten we de schakeling aan?

- **Bovenkant van de H-brug**: De **source** van **Q1** en **Q2** wordt verbonden met de positieve spanning (**Vcc**, bijvoorbeeld +12V).
- **Onderkant van de H-brug**: De **source** van **Q3** en **Q4** wordt verbonden met **GND** (de negatieve spanning).
- **LED**: De LED wordt tussen de uitgangen van de H-brug geplaatst.

Het gebruik van een Arduino om de H-brug aan te sturen

Om de MOSFETs te bedienen, gebruiken we een **microcontroller** zoals een **Arduino**. De Arduino kan de **gates** van de MOSFETs besturen en bepalen of de LED aan of uit is, afhankelijk van welke MOSFETs we aansteken.

De Arduino code

De Arduino code die we gebruiken om de H-brug aan te sturen ziet er als volgt uit:

```
int Q1 = 2; // Gate van Q1
int Q2 = 3; // Gate van Q2
int Q3 = 4; // Gate van Q3
int Q4 = 5; // Gate van Q4

void setup() {
  pinMode(Q1, OUTPUT);
  pinMode(Q2, OUTPUT);
  pinMode(Q3, OUTPUT);
  pinMode(Q4, OUTPUT);
}

void loop() {
  // Draai de LED aan (licht op)
  digitalWrite(Q1, LOW);  // Zet Q1 aan (negative voltage)
  digitalWrite(Q2, HIGH); // Zet Q2 uit
```

```
  digitalWrite(Q3, HIGH); // Zet Q3 uit
  digitalWrite(Q4, LOW);  // Zet Q4 aan (negative voltage)
  delay(2000);  // LED blijft 2 seconden aan

  // Zet de LED uit (dim of uit)
  digitalWrite(Q1, HIGH);  // Zet Q1 uit
  digitalWrite(Q2, LOW);   // Zet Q2 aan
  digitalWrite(Q3, LOW);   // Zet Q3 aan
  digitalWrite(Q4, HIGH);  // Zet Q4 uit
  delay(2000);  // LED blijft 2 seconden uit
}
```

Wat doet deze code?

- **De eerste sectie (LED aan)**:
  - Zet **Q1 en Q4 aan** en **Q2 en Q3 uit**.
  - Dit zorgt ervoor dat de stroom door de LED vloeit in de juiste richting, waardoor de LED **oplicht**.
- **De tweede sectie (LED uit)**:
  - Zet **Q2 en Q3 aan** en **Q1 en Q4 uit**.
  - Dit zorgt ervoor dat de stroom in de omgekeerde richting door de LED vloeit, waardoor de LED **uitgaat** of gedimd is.

Testen en observeren

Wanneer je de schakeling in elkaar zet en de code op de Arduino laadt, zou je moeten zien dat de **LED afwisselend aan en uit gaat** of dimt, afhankelijk van hoe de stroom door de LED wordt geleid. Dit gebeurt doordat je de MOSFETs aan- en uitzet via de Arduino, waardoor je de richting van de stroom door de LED verandert.

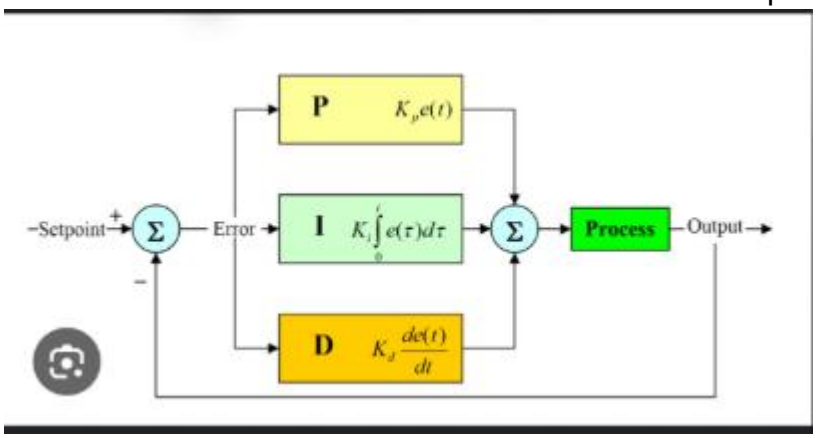
4. Gebruikt van PID

PID staat voor Proportioneel, Integrerend, Differentieel.
Het is een feedbacksysteem dat wordt gebruikt om systemen te sturen, zoals bijvoorbeeld een robotmaaier, waarbij we de fout (verschil tussen de gewenste en werkelijke waarde) zo klein mogelijk willen houden.
Het doel van PID is om de robot nauwkeurig te sturen door continu te reageren op de afwijkingen van een doelwaarde.

4.1. Basisprincipes PID

PID bestaat uit drie onderdelen, die elk op een andere manier reageren op de fout:

1. **Proportioneel (P)**
    - Reageert op de **huidige fout**.
    - Hoe groter de fout, hoe groter de correctie.
    - Bijvoorbeeld, als de robot ver van het pad is, wordt de correctie groter om snel naar het pad te sturen.
2. **Integrerend (I)**
    - Reageert op de **oplopende fout** over tijd.
    - Het helpt kleine, constante afwijkingen die blijven bestaan, zoals sensorfouten, te corrigeren.
    - Dit voorkomt dat de robot steeds een kleine fout maakt die zich opstapelt.
3. **Differentieel (D)**
    - Reageert op de **verandering in de fout**.
    - Het helpt te voorspellen hoe snel de fout zal toenemen en corrigeert voordat de fout te groot wordt.
    - Dit voorkomt dat de robot te ver van het pad afwijkt of overshoot maakt



4.2. Wiskunde van PID

De PID-regelaar is eenvoudig te begrijpen door naar de formules te kijken. De volledige PID-regelaar kan als volgt worden uitgedrukt:

$$u(t) = K_p \times e(t) + K_i \times \int e(t)\,dt + K_d \times \frac{de(t)}{dt}$$

waarbij:

- $u(t)$is de stuuractie (hoe de robot wordt aangestuurd).
- $e(t)$is de fout (verschil tussen de gemeten waarde en de gewenste waarde).

- K_p, K_I, K_D zijn de respectieve PID-parameters (proportioneel, integrerend en differentieel).
- $\int e(t)dt$ is de cumulatieve fout over tijd (integraal).
- $\frac{de(t)}{dt}$ is de snelheid waarmee de fout verandert (afgeleide).

4.3. Toepassen van PID

Snelheidsregeling

- De robotmaaier moet een constante snelheid behouden, zelfs als het terrein oneffen is. De PID-regelaar kan helpen door de motoren aan te passen zodat de robot niet te snel of te langzaam gaat, ongeacht het terrein.
- De fout $e(t)$ zou in dit geval het verschil zijn tussen de gewenste snelheid en de werkelijke snelheid van de robot. Het doel is dat de robot de gewenste snelheid behoudt door PID te gebruiken om de motoren continu aan te passen.

Positieregulatie (zoals een willekeurige beweging)

- Stel je voor dat je robotmaaier een willekeurig pad volgt en constant zijn positie probeert te behouden (bijvoorbeeld in een vierkant of willekeurige beweging in een tuin). De PID-regelaar kan worden gebruikt om te corrigeren als de robot uit zijn pad dreigt te raken.
- De fout kan in dit geval de **hoek** of **positie ten opzichte van een startpunt** zijn. Bijvoorbeeld, als de robot verder van zijn gewenste positie komt, zal de PID-regelaar de stuurmotoren aansteken om de robot terug te brengen naar zijn doelgebied.

Stabiliteit en besturing

- Stel je voor dat de robotmaaier een bepaald gebied moet afbakenen, maar niet per se een vaste lijn volgt. Hier wordt PID gebruikt om de **hoek** of **oriëntatie** van de robot te stabiliseren, bijvoorbeeld om ervoor te zorgen dat de robot niet in een cirkel draait of plotseling van richting verandert.
- De fout in dit geval kan het **verschil in hoek** zijn (de huidige oriëntatie van de robot vs. de gewenste oriëntatie).

Actueel

- PID controllers zitten eigenlijk overal. Elk systeem dat een mechanische toepassing heeft gebruikt waarschijnlijk een PID controller. Thermostaten, Cruisecontrol in auto's, Drones, Industriële robots, 3D-printers, Elektrische oven, Waterniveaucontrole in reservoirs, Zelfbalancerende robots, Gimbal-systemen voor camera's, Luchtdrukregeling in pneumatische systemen

4.4. Opdracht PID

In deze opdracht ga je een eenvoudige PID-regelaar implementeren die een sensor leest en op basis van de gemeten waarde een actuator aanstuurt. Het doel is om te

leren hoe PID-regelaars werken en hoe ze kunnen worden toegepast voor het controleren van systemen op basis van sensorinformatie.

Benodigdheden:

- Microcontroller (bijvoorbeeld ESP32, Arduino, of een vergelijkbaar bord)
- Sensor (bijvoorbeeld een temperatuursensor, afstandssensor, of lichtsensor)
- Actuator (bijvoorbeeld een motor, servo, of LED)
- PID-algoritme (code om PID te implementeren)
- Verbindingsdraden en breadboard (indien nodig)

Opdrachtbeschrijving:

1. Sensor Configureren: Kies een sensor die een meetwaarde kan leveren, zoals een temperatuursensor (bijvoorbeeld de DHT11 of DHT22), een afstandssensor (bijvoorbeeld de HC-SR04) of een lichtsensor. Je kunt de sensor aansluiten op de microcontroller en de juiste bibliotheek gebruiken om de sensorwaarde uit te lezen.
2. PID-regelaar Implementeren:
    - Implementeer een PID-regelaar in de code. De PID-regelaar zal de fout berekenen, die het verschil is tussen de gewenste waarde en de gemeten waarde van de sensor.
    - De PID-regelaar heeft de drie parameters: K_p, K_I, K_D die je kunt afstemmen.
3. Aansturing van de Actuator: Gebruik de PID-uitvoer om een actuator aan te sturen. Dit kan bijvoorbeeld een motor zijn die sneller of langzamer draait, een servo die beweegt naar een bepaalde positie, of een LED die helderder of dimmer wordt.
    - De PID-uitvoer kan de snelheid of de positie van de actuator sturen. Bijvoorbeeld, als je een temperatuurregeling doet, zou de actuator (bijvoorbeeld een ventilator) harder moeten draaien als de temperatuur te hoog is en langzamer draaien als de temperatuur dichtbij de gewenste waarde komt.
4. Afstemming van PID-parameters: Afhankelijk van je systeem, zul je de PID-parameters moeten afstemmen om de juiste werking te verkrijgen:
    - K_p (proportioneel): Hoeveel de actuator direct reageert op de fout.
    - K_I (integrerend): Hoeveel de actuator reageert op de geaccumuleerde fout over tijd.
    - K_D (differentieel): Hoeveel de actuator reageert op de snelheid van de verandering van de fout.
5. Resultaten Observatie:
    - Observeer hoe de actuator reageert op veranderingen in de sensorwaarde.
    - Pas de PID-parameters aan om te zorgen dat de actuator soepel reageert zonder te veel oscillatie of te traag reageren.

Stap-voor-stap Uitleg:

1. Sensor uitlezen: Lees de waarde van de sensor in je code uit.
    - Voor een temperatuursensor zou de code bijvoorbeeld zoiets kunnen zijn:

Bereken de fout:
De fout is het verschil tussen de gewenste waarde en de gemeten waarde.

2. Bijvoorbeeld, als de gewenste temperatuur 25°C is en de gemeten temperatuur is 30°C, is de fout:

Toepassen van PID-regeling: Bereken de PID-uitvoer op basis van de fout.

3. Je hebt de drie parameters nodig, die de werking van de PID-regelaar bepalen.

Aansturing actuator: Gebruik de PID-uitvoer om de actuator te sturen.

4. Als je een motor hebt, kun je de snelheid aanpassen aan de PID-uitvoer:

Vraag voor Reflectie:

- Hoe reageert je systeem als je de PID-parameters verandert? Probeer verschillende waarden voor K_p, K_I, K_D om te zien welke het beste werkt voor jouw systeem.
- Wat gebeurt er als de PID-parameters te hoog of te laag zijn ingesteld? Kun je de robot soepel laten bewegen door de juiste afstemming te vinden?

Extra uitdaging:

- Implementeer extra aansturingen voor deze controller, bijvoorbeeld je wilt dat de controller maar maximaal 10% veranderd, hoe zou je dit aanpakken?
- De wiskundige formule is complex maar ‘simpel’ in uitvoer. Bekijk online eens een korte video om de werking nog beter te begrijpen.

THOMAS MORE