EDUGO SCHOLENGROEP
EDUGO Campus Glorieux
SCHOOL VOOR WETENSCHAP & TECHNIEK
Sint-Jozefstraat 7 9041 Gent-Oostakker
Tel. school: 09/255 91 15

Verslag geïntegreerde proef

Titel: E-chess

Naam: Art Van der Vennet
Studierichting: Elektriciteit-Elektronica
Klas: 6EE
GIP-begeleiders: Geert Eggermont, Luigi Vallozzi, Alain Van Eetvelde
Schooljaar: 2022-2023

Woord vooraf

Dit verslag heb ik geschreven als onderdeel van de geïntegreerde proef in het 6de middelbaar met als doel het eindwerk toe te lichten.

Het einde van het laatste jaar voor we allemaal afstuderen is nabij en dit is één van de belangrijkere taken voor het behalen van je diploma.

Mijn doelstelling is om een GIP te maken die mij dermate interesseert, dat ik gemotiveerd blijf bij het praktisch gedeelte ervan.

Ik wens volgende personen te bedanken voor alle hulp die zij me geboden hebben:

- het directieteam: Dhr. Bollaert, Mevr. Van De Meulebrouck, Mevr. Vissenaeken, Dhr. Malfliet;
- afdelingsverantwoordelijke: Dhr. Dufour;
- mijn klastitularis: dhr. Eggermont;
- mijn taalleerkracht: dhr. Wulleman;
- mijn leerkracht elektriciteit: dhr. Van Eetvelde;
- mijn leerkracht Telecommunicatie: dhr. Vallozzi;
- medeleerlingen;
- ouders, vrienden;
- alle andere leerkrachten die ik onderweg naar het afstuderen heb gehad;
- ...

Tekstverwerkingsprogramma: Microsoft Office Word 2016

Printer: Brother MFC-J5320DW

Inhoudsopgave

Inleiding

Het spel schaken kent een lange geschiedenis die teruggaat tot eeuwen geleden. Echter, het vereist altijd twee spelers om het spel te kunnen spelen. Dit is een probleem dat mijn vriend ondervindt. Hij heeft een enorme passie voor schaken, maar heeft niemand om tegen te spelen. Het enige alternatief is digitaal spelen, maar dat biedt niet dezelfde ervaring. Om dit te veranderen wil ik een manier bedenken om alleen te kunnen schaken op een fysiek schaakbord.

Sinds vorig jaar ben ik begonnen met het bedenken van oplossingen voor mogelijke uitdagingen die hierbij zouden kunnen komen kijken. In januari ben ik daadwerkelijk begonnen met het uitwerken van mijn plannen. Hierbij ben ik onverwachte problemen tegengekomen, maar dankzij begeleiding van vrienden en leerkrachten heb ik mijn uiterste best gedaan om deze problemen zo goed mogelijk op te lossen.

In dit verslag ga ik jullie meenemen doorheen mijn denkproces en hopelijk op die manier de werking van mijn GIP duidelijk te maken. Ik zal jullie tonen wat mijn origineel idee was, en hoe het is uitgegroeid tot het finale resultaat. Ik wens jullie al vast veel leesplezier.

1 Drie GIP-voorstellen

1.1 Automatisch schaakbord

Mijn eerste keuze ging uit naar een fysiek schaakbord waarin een schaakcomputer wordt verwerkt. Dit project zat al een paar maanden op voorhand in mijn hoofd waardoor ik hier veel zin in had om dit te maken. Dit is dan uiteindelijk ook mijn effectieve GIP geworden

1.2 Holographic fan

foto 1: Holographic fan

Mijn tweede keuze was een hologram dat werd geprojecteerd doormiddel van 4 ronddraaiende LED-strips. Dit had ik gezien op een beurs voor technologie in Duitsland een paar jaar geleden en leek me wel leuk om te maken.

1.3 Automatisch slot

Als derde optie dacht ik aan een slot om een deur open te maken dat ik zou kunnen bedienen met een app. Dit idee was mijn laatste keuze omdat ik hier het minste over nagedacht had.

2 Het schaakspel

Het schaakspel wordt individueel gespeeld met als doel de tegenstanders koning te elimineren met behulp van één van de eigen stukken. Gewoonlijk wordt dit spel gespeeld door twee spelers, maar mijn intentie is om één van de spelers te vervangen door een computer die in het bord zit.

2.1 Spelregels

2.1.1 Het bord

Het bord bestaat uit een acht bij acht veld. De X-as zal van links naar rechts benoemd worden door de letters A tot H. De Y-as zal van onder naar boven benoemd worden door cijfers 1 tot 8. De kleuren van de vakjes zullen telkens wisselen van zwart naar wit zodat je een geruit vlak krijgt. Zo krijg elk vakje een naam op basis van coördinaten zoals ‘A1’ of ‘E4’.

2.1.2 De stukken

Elke speler heeft acht pionnen, twee torens, twee paarden, twee lopers, één koningin en één koning.

2.1.2.1 Pionnen

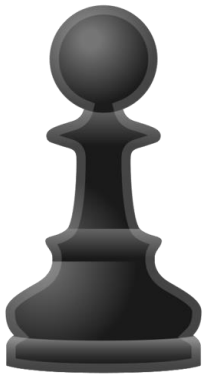


foto 2: pion

De pionnen zullen bij wit staan op de vakjes van A2 tot H2 en bij zwart van A7 tot H7.

De pionnen kunnen alleen vooruit bewegen, telkens één vakje. Behalve bij de eerste zet die de pion maakt kan de speler kiezen om de pion één of twee vakjes vooruit te bewegen.

De pion kan alleen schuin-vooruit aanvallen.

Als de pion het bord volledig kan oversteken, dus als de Y-coördinaat bij wit acht is of bij zwart één is, dan kan die gepromoveerd worden. Als een pion gepromoveerd wordt kan die vervangen worden door elk ander stuk behalve een koning. Zo kunnen er twee koninginnen of drie torens op het bord staan.

2.1.2.2 Torens

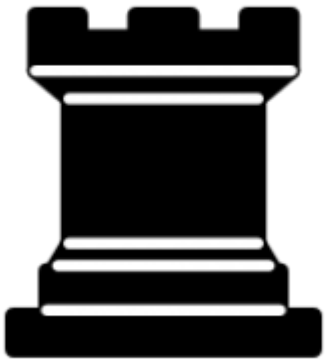


foto 3: toren

De torens kunnen verticaal en horizontaal bewegen over het bord, maar niet diagonaal. De torens kunnen bewegen tot ze een ander stuk tegenkomen.

De witte torens starten op A1 en H1 en de zwarten starten op A8 en H8.

2.1.2.3 Paarden

foto 4: paard

De paarden bewegen telkens in een L-vorm, dus telkens één vak verticaal of horizontaal en één vak diagonaal in dezelfde richting.

De witte paarden starten op B1 en G1 en de zwarte paarden op B8 en G8.

2.1.2.4 Lopers

foto 5: loper

De lopers kunnen enkel diagonaal bewegen. De lopers kunnen bewegen tot ze een ander stuk tegen komen.

De witte lopers starten op C1 en E1 en de zwarte lopers op C8 en E8.

2.1.2.5 Koningin

foto 6: koningin

De beweging van de koningin is een combinatie van een de beweging van een loper en een toren. Zij kan zowel diagonaal, horizontaal als verticaal bewegen en kan bewegen tot zij een ander stuk tegen komt.

De witte koningin zal starten op D1 en de zwarte koningin op D8.

2.1.2.6 Koning

foto 7: koning

De koning kan 1 vakje bewegen in elke richting. Maar de koning kan zichzelf nooit schaak zetten.

3 Realisatie

3.1 Probleemstelling

Een schaakcomputer in het bord implementeren is gemakkelijker gezegd dan gedaan. Om de posities te weten van de spelers stukken zal onder elk vakje een soort detector geplaatst moeten worden diekan doorgeven of er een stuk op het vakje staat of niet.

Hiervoor zijn er 64 (8x8) detectoren nodig. Als deze allemaal individueel gecontroleerd moeten worden, zijn er 64 in-/uitgangen nodig op de chip van het schaakbord.

Dan heb ik ook nog een manier nodig om te tonen welke zet de computer maakt. Hiervoor dacht ik om, zoals in schaaknotatie (bv: E2 E4), het vakje waarop het stuk staat en het vakje waar het stuk naar toe gaat op te lichten.

Daarvoor zou ik ook onder elk vakje een LED nodig hebben, dus 64 LEDs in totaal. Als ik alle LEDs en schakelaars apart wil bedienen en controleren zou ik 128 in-/uitgangen nodig hebben op de chip.

Dat is zeer onrealistisch aangezien de meeste Arduino's of Raspberry Pi's ongeveer 40 pinnen hebben. Dus moest ik een andere manier vinden.

3.2 Oplossing

Het schaakbord zal een chip bevatten die alle in- en uitgangen zal lezen. Deze gegevens zullen dan via wifi verstuurd worden naar een PC om zo de juiste zetten te kunnen berekenen. Immers die berekening is veel te moeilijk voor een kleine chip.

3.2.1 Raspberry Pi Pico W

foto 8: Raspberry Pi Pico W

De chip die ik zal gebruiken zal de Raspberry Pi Pico W zijn. Ik heb gekozen voor deze chip omdat ik hierop in de programmeertaal Python kan programmeren in tegenstelling tot Arduino, die op een eigen taal werkt die gebaseerd is op C. Deze chip kan bovendien ook verbinding maken met wifi.

De Raspberry Pi Pico W heeft 40 pinnen waarvan 26 in-en uitgangen.

De voedingsspanning is 5V.

De in-en uitgangen worden bestuurd van 0V-3,3V.

3.2.2 In-en uitgangen

De in- en uitgangen zullen gestuurd worden met een coördinatensysteem waar ik 16 (8, 8) mosfets aan en uit kan zetten om een vakje te kunnen lezen of sturen.

De ingangen zullen bestuurd worden met magnetische schakelaars en de schaakstukken zullen allemaal een ingebouwde magneet hebben. Zo kan ik zien welk stuk waar staat en kan ik alle stukken ook volgen.

Op de uitgangen zullen LEDs staan die zullen branden als het stuk op dat vakje moet bewegen en waar naar het zal bewegen. Er zullen dus altijd maar twee LEDs tegelijk branden.

De in-en uitgangen zullen worden gestuurd met een scan-systeem waarbij ik alle vakjes rij per rij uitlees en uitstuur zodanig dat er geen signalen door elkaar worden gehaald.

3.2.3 Wat bij promoveren?

Bij het promoveren van een pion kan je normaal kiezen tussen elk ander stuk, behalve de koning. Maar aangezien dit bord niet zelf zijn pionnen kan verschuiven heb ik ervoor gekozen om te promoveren altijd te laten gebeuren naar een koningin, de populairste zet.

3.3 Componenten

3.3.1 Schakelaars

foto 9: reed switch

Mijn keuze voor de schakelaars is reed-switches omdat die kunnen bediend worden door een magneet onderaan de schaakstukken. Doordat de magneet onderaan het schaakstuk zit, is het niet zichtbaar tijdens

het spelen van het spel.
In de reed-switches zitten twee kleine metalen plaatjes. Door daar een magneet op te leggen worden die metalen plaatjes op elkaar geduwd en zo krijg je een contact.

3.3.2 LEDs

foto 10: LED

De LEDs die ik zal gebruiken onder elk vakje zullen groene 'superbright' LEDs zijn. Deze waren voorradig op school.

3.3.3 Mosfets

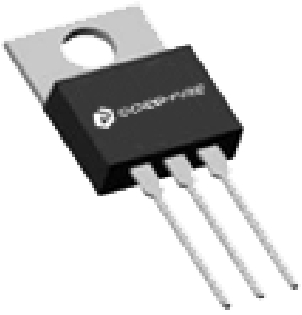


foto 11: mosfet

Ik heb mosfets gekozen als elektrische schakelaars omdat er, in tegenstelling tot transistors, bij mosfets quasi geen spanning over de source en de drain pinnen staat. Dus mijn LEDs zullen maximaal branden.

3.3.4 Diodes

foto 12: diode

Diodes zorgen ervoor dat de stroom maar in één richting kan vloeien.

Om te voorkomen dat de schakelaars invloed zouden hebben op de LEDs heb ik een diode na elke schakelaar geplaatst.

3.3.5 Weerstanden

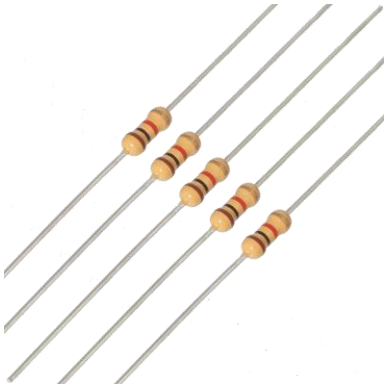


foto 13: weerstanden

Voor elke LED heb ik een weerstand van 220Ω geplaatst zodanig dat de stroom door de LED maximaal 22mA kan zijn. Op die manier zullen de LEDs zo sterk mogelijk branden en een heel lange levensspan hebben.

Na elke kolom van schakelaars heb ik ook telkens een pull-down weerstand van 4,7KΩ gezet om zo ruis op mijn ingangen te voorkomen.

3.3.6 Powerbank

foto 14: powerbank

Deze powerbank heeft een capaciteit van 5000 mAh. Dat is evenveel als de gemiddelde GSM momenteel dus meer dan genoeg om een paar uren schaakgenot te hebben.

Ik heb deze verkozen omwille van zijn compacte grootte en redelijke prijs.

4 Schema's

4.1 Verduidelijking blokschema

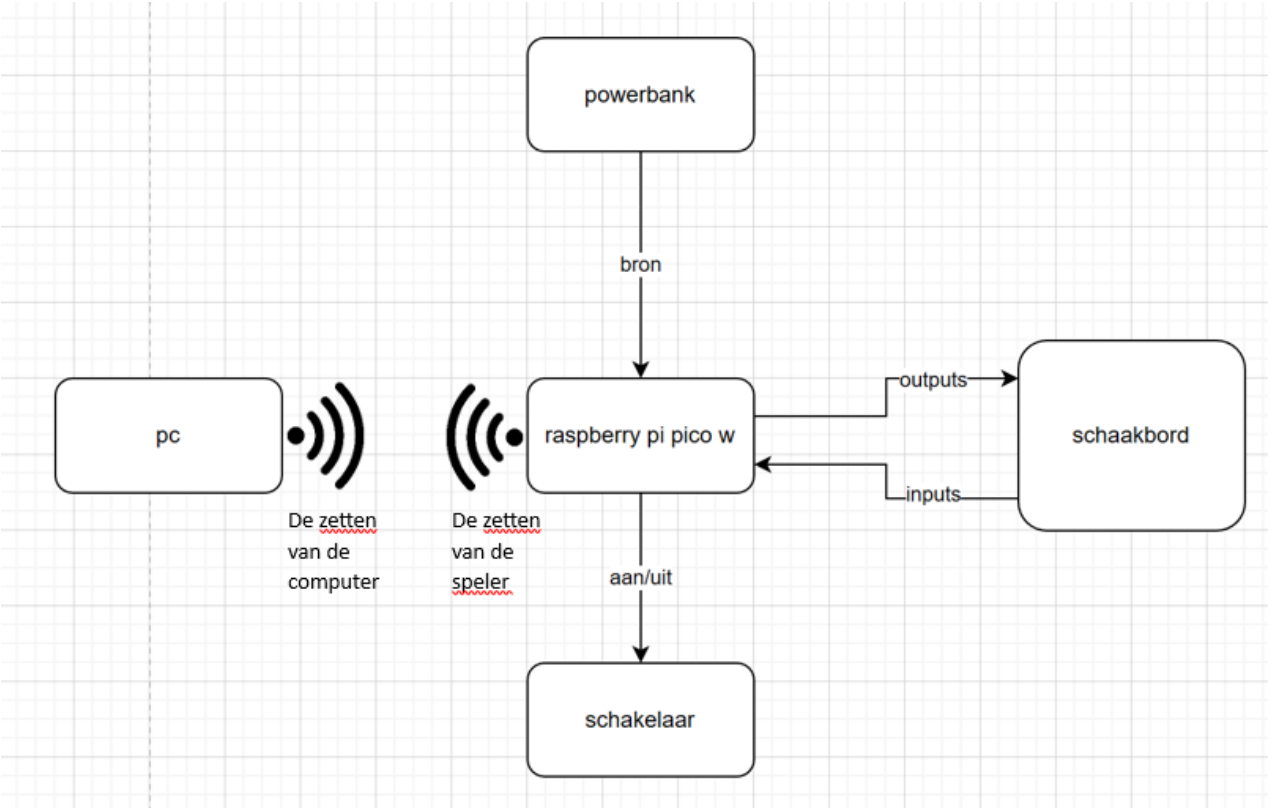


Foto 15: blokschema

4.1.1 Bron

Dit blokschema toont dat er gekozen is voor een powerbank als voedingsbron. De uitgangsspanning van de powerbank is al 5V.

Een andere optie was gebruik maken van een stopcontact, maar dat heeft het nadeel dat er een stopcontact in de buurt moet zijn, en dat er nog een omzetting moet gebeuren van 230VAC naar 5VDC.

Het alternatief om met batterijen te werken, werd verworpen omdat het vervangen van batterijen minder milieuvriendelijk en duurzaam is dan het hergebruik van een powerbank door opnieuw op te laden

4.1.2 PC

Om de zetten te berekenen die de computer moet zetten zal de Raspberry Pi Pico W de zet van de speler doorsturen naar de PC. Die zal dan op basis van een schaakengine zoals Stockfish de beste zet berekenen en dan terugsturen naar de Raspberry Pi Pico W.

Deze berekeningen kunnen niet lokaal worden gedaan aangezien de Raspberry Pi Pico W maar een werkgeheugen van 848 kB heeft.

4.2 Bespreking elektrisch schema

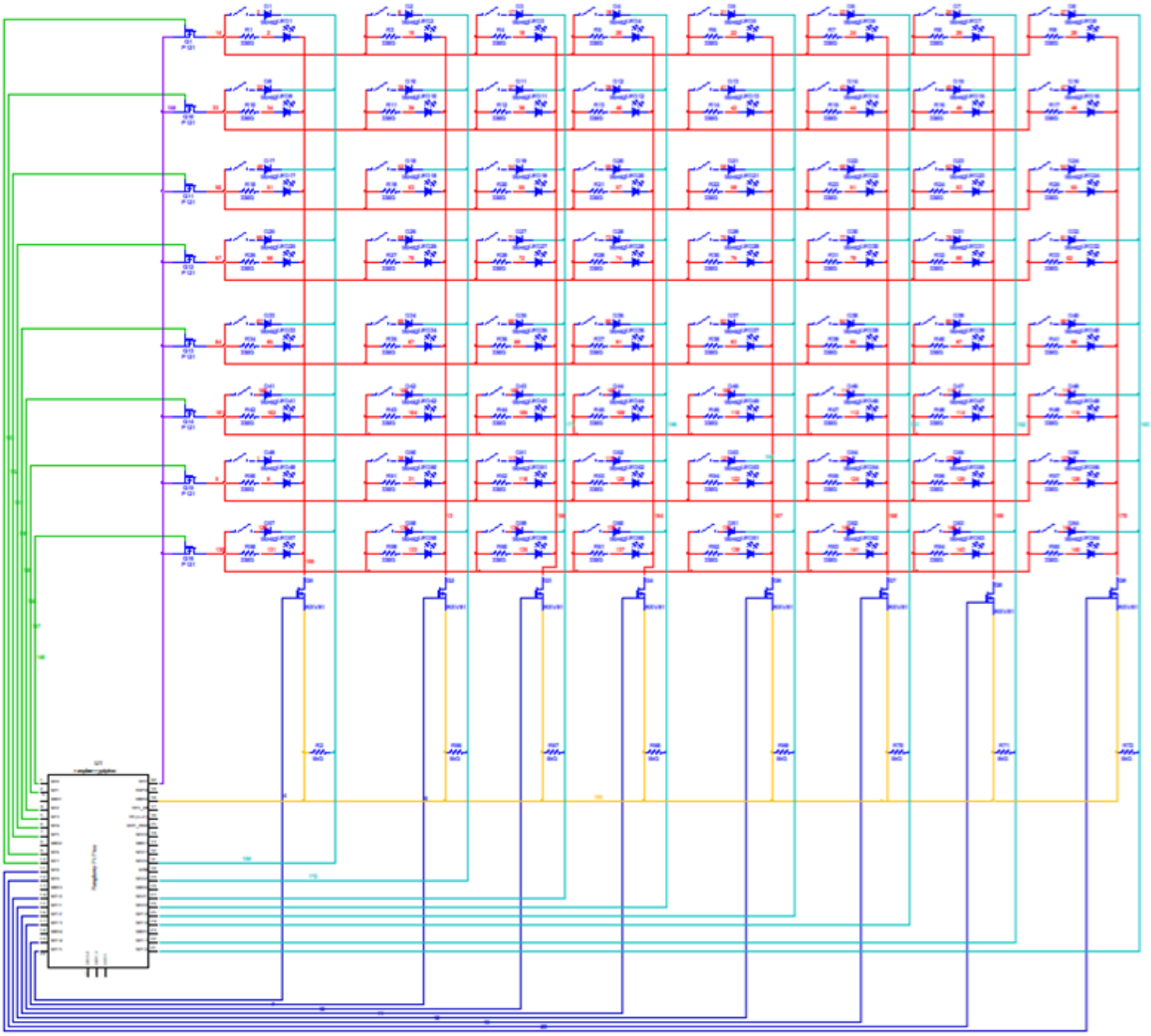


foto 16: volledig elektrisch schema

4.2.1 LEDs

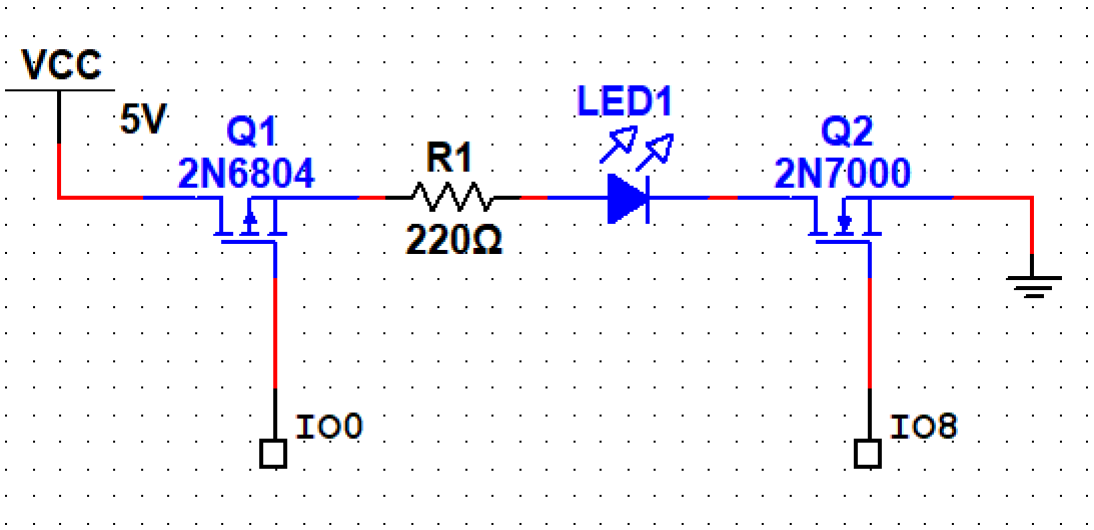


foto 17: elektrisch schema, LEDs

In dit schema zien we hoe de LEDs individueel worden geschakeld. Beide transistors zullen in geleiding moeten zijn om de LED te laten branden. De LED zal alleen branden als Q1 de waarde 0 heeft en Q2 de waarde 1(3,3V) heeft.

Q1	Q2	LED
0	0	0
0	1	1
1	0	0
1	1	0

4.2.2 Schakelaars

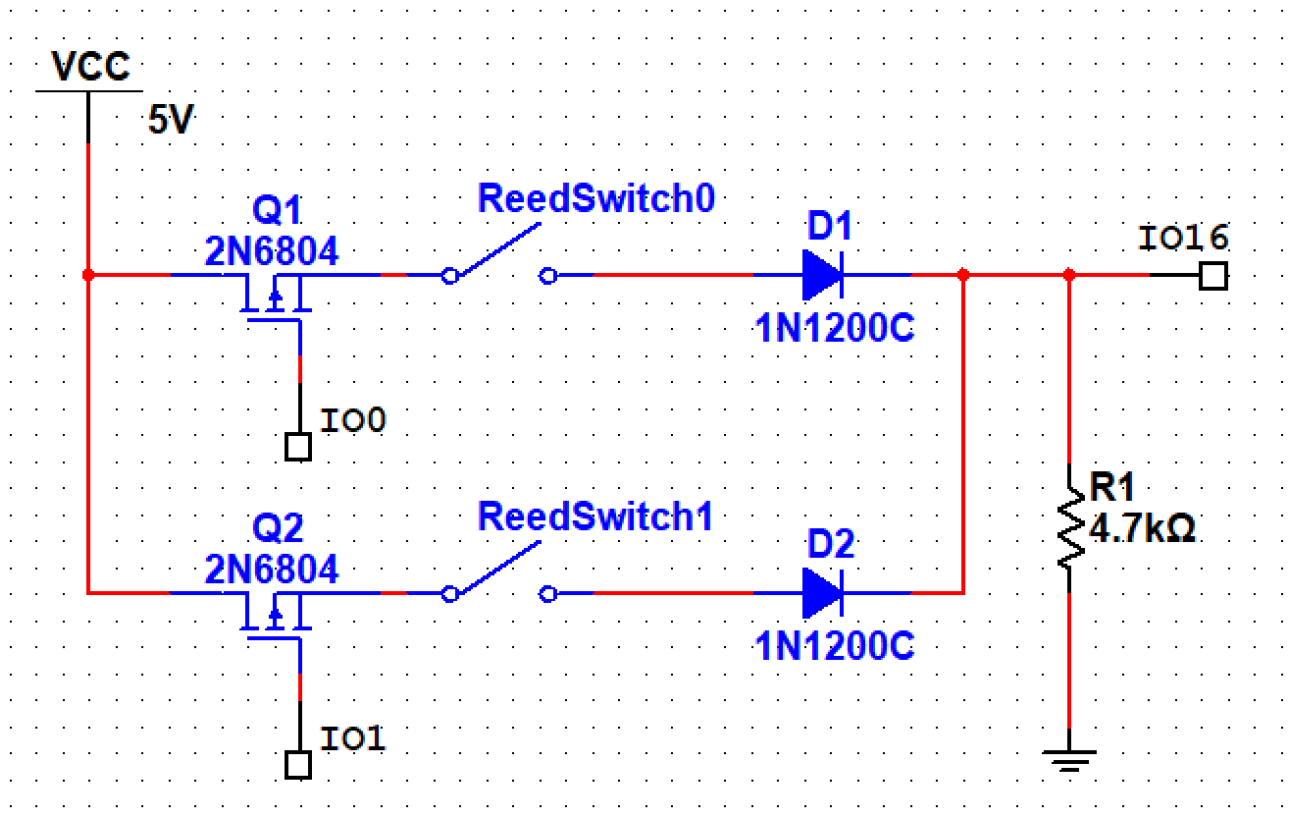


foto 18: elektrisch schema, schakelaar

De ReedSwitch0 wordt enkel uitgelezen als er een 0 komt op Q1. De schakelaars worden per acht op één mosfet parallel geschakeld zodanig dat deze per acht parallel kan worden inlezen.

De scenario's waar beide Q2 en Q1 0 zijn, zijn onmogelijke scenario's omdat er altijd maar 1 per keer geleid.

Q2	ReedSwitch1	Q1	ReedSwitch0	Waarde op I016
0	**0**	**0**	0	X
0	**0**	**0**	1	X
0	**0**	**1**	0	0
0	**0**	**1**	1	0
0	**1**	**0**	0	X
0	**1**	**0**	1	X

0	**1**	**1**	0	1
0	**1**	**1**	1	1
1	**0**	**0**	0	0
1	**0**	**0**	1	1
1	**0**	**1**	0	0
1	**0**	**1**	1	0
1	**1**	**0**	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

4.2.3 Samenwerking van de LEDs en schakelaars

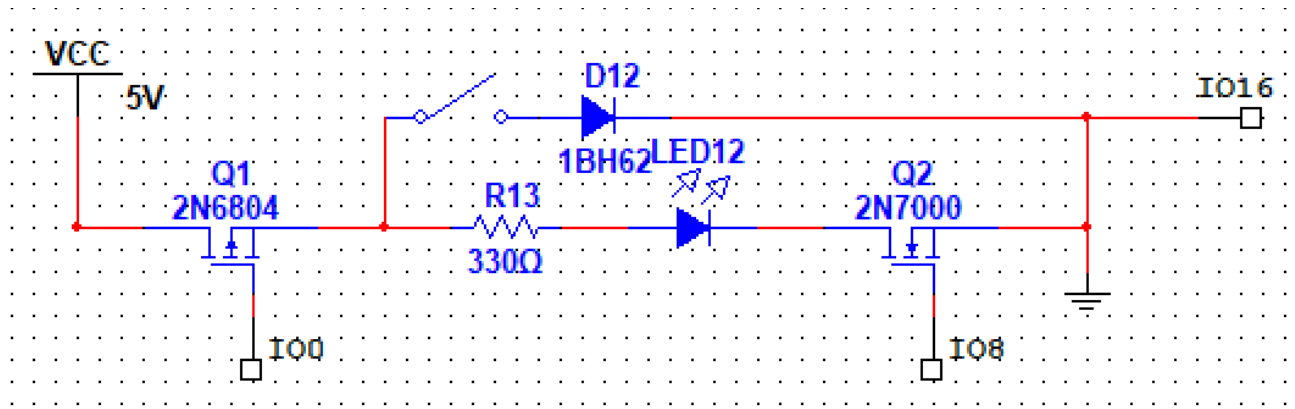


foto 19: elektrisch schema, schakelaar en LED

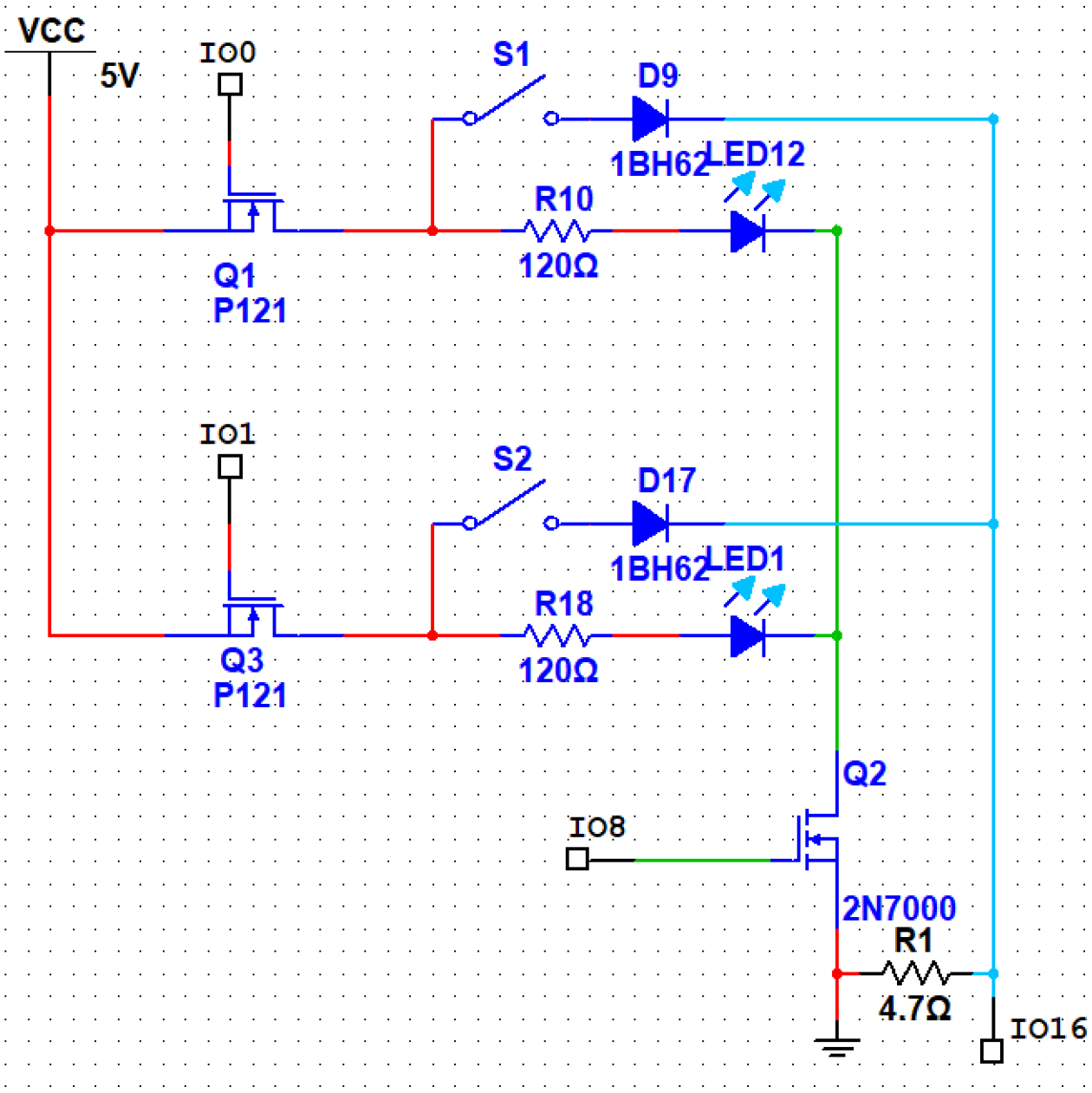


foto 20: elektrisch schema, schakelaar en LED, 2 rijen

Hier kunnen we zien dat er een diode nodig is om ervoor te zorgen dat de verkeerde LEDs niet aangaan als er een schakelaar ingedrukt is.

Als er geen diode is geplaatst, kan het volgende gebeuren:

S1 en S2 zijn beide gesloten.

We willen LED1 inschakelen (Q3 = 0 en Q2 = 1).

De stroom van Q3 kan via S2 naar S1 lopen.

Als de stroom de schakelaar S1 bereikt, kan deze door de gesloten schakelaar S2 gaan en de weg vinden naar LED12.

LED12 zal oplichten, ook al wilden we alleen LED1 inschakelen.

Om dit te voorkomen, moeten we een diode toevoegen in de juiste richting om de stroom in de gewenste richting te geleiden en te voorkomen dat deze via de verkeerde weg stroomt.

4.3 Ultiboard

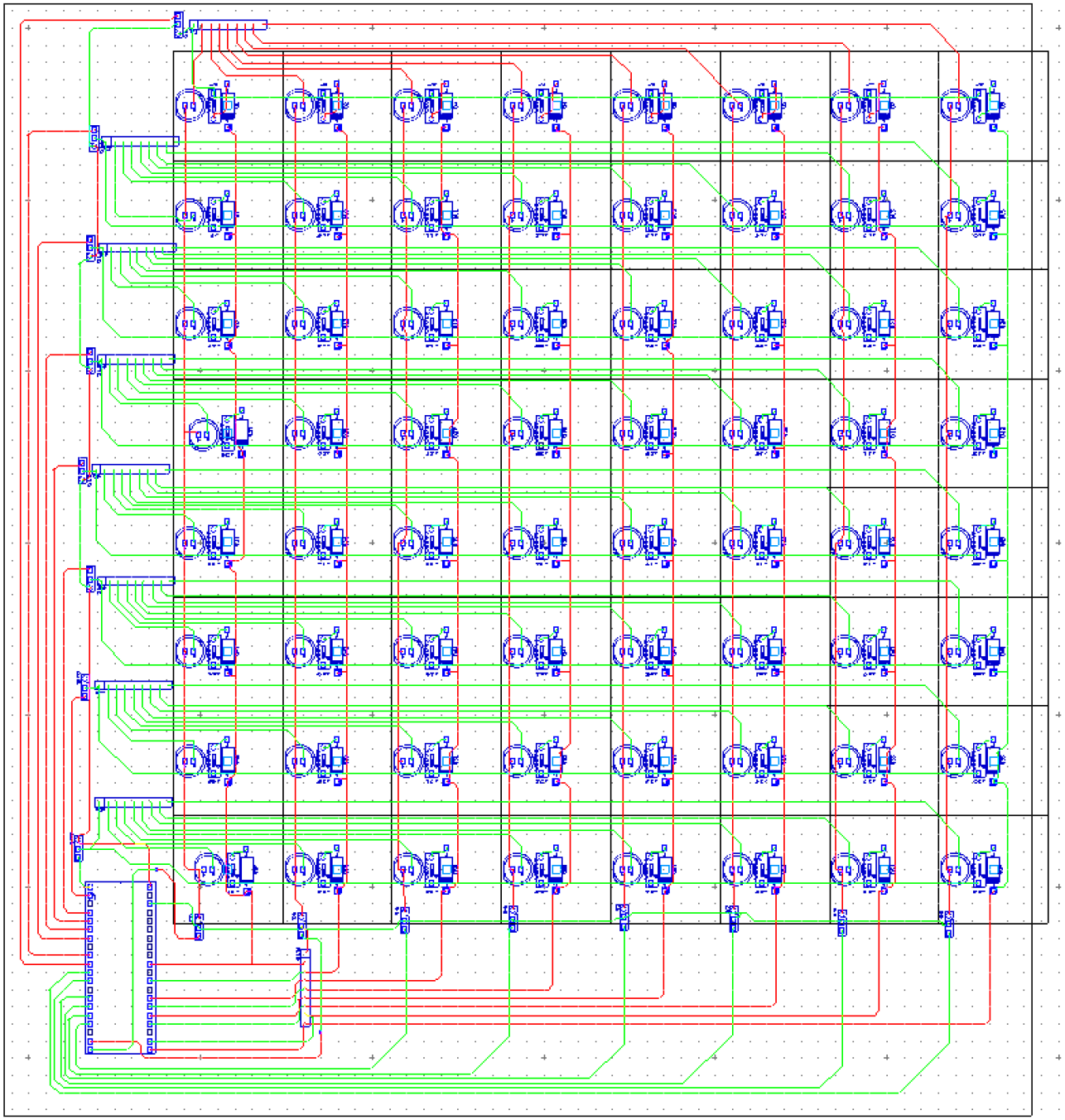


foto 21: ultiboard schema

Dit schema stelt voor hoe de printplaat eruit zal zien. Alle rode lijnen stellen de onderste koperlaag voor. De groene lijnen stellen de bovenste koperlaag voor. Ik heb voor een printplaat gekozen omdat het duidelijker en overzichtelijker zou zijn. Zo is het ook makkelijker om eventuele fouten te zoeken.

4.4 BOM

aantal	naam	ordernummer	prijs per stuk	leverancier
8	n channel mosfet	STU6N65M2	€ 0.29	https://be.farnell.com/stmicroelectronics/stu6n65m2/mosfet-n-channel-650v-4a-to-251/dp/2460384
8	p channel mosfet	FQU8P10TU	€ 0.88	https://gr.mouser.com/ProductDetail/onsemi-Fairchild/FQU8P10TU?qs=mdiO5HdF0Kj2UXljY%2FMuDw%3D%3D
64	reed switch	ORD 324/20-30 AT	€ 0.55	https://gr.mouser.com/ProductDetail/MEDER-electronic-Standex/ORD-324-20-30-AT?qs=WZYOKEqGBpJdxKZwoWl9iw%3D%3D
1	Raspberry Pi Pico W		€ 7.95	https://www.elektor.nl/raspberry-pi-pico-rp2040-w
64	led		€ 0.10	school
64	weerstand	330 ohm	€ 0.05	school
1	printplaat	E1496046	€ 258.54	Eurocircuits N.V.
16	magneten	Kubus NM-5-5-5	€ 0.47	https://www.magnetenkopen.nl/winkel/neodymium-kubusmagneet-5-mm-houdkracht-11-kg/
1	powerbank		€14.99	mediamarkt
1	schakelaar		€4	school
totaal:			€ 337.31	

De grootste kost is de printplaat. Deze printplaat kost zo veel omdat die redelijk groot is. Ik heb gekozen voor zo'n grote printplaat omdat alles dan beter zou vastzitten. Zo moet ik ook geen extra draden trekken om mijn LEDs te verbinden. Een alternatieve printplaat met vergelijkbare grootte, maar iets goedkoper, werd niet gekozen omdat deze minder overzichtelijk was.

4.5 Behuizing

Hieronder zullen al de tekeningen van mijn behuizing staan. Met behulp van deze tekeningen heb ik samen met het bedrijf Publipunt, dat gevestigd is in Evergem, het fysieke schaakbord gerealiseerd.

4.5.1 Pvc plaat

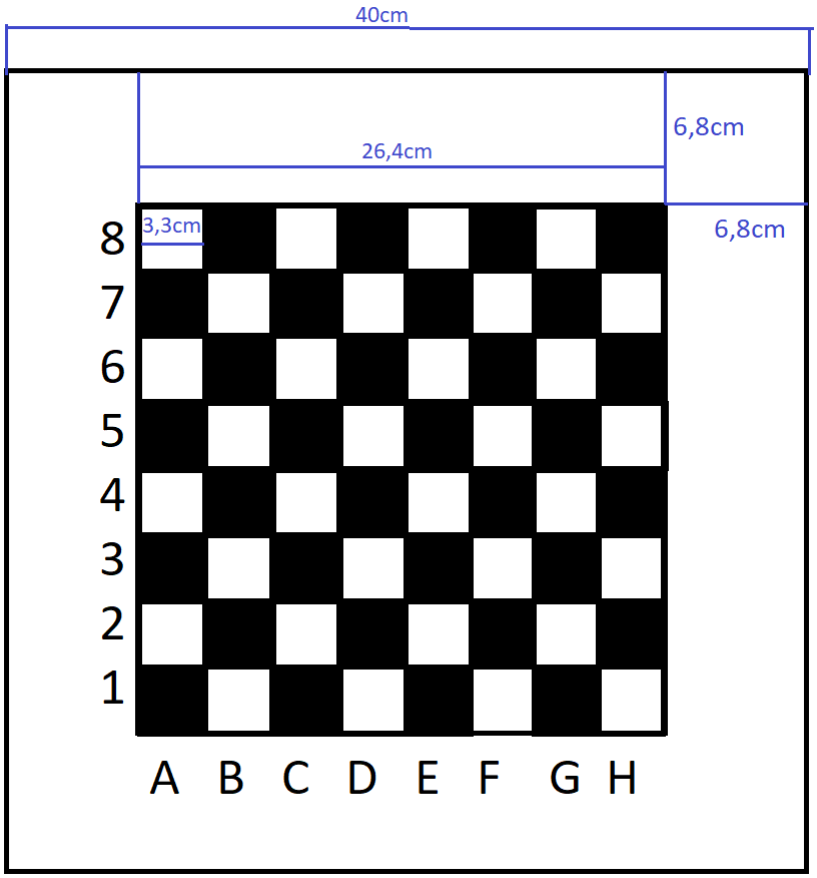


foto 22: pvc plaat

4.5.2 Doos

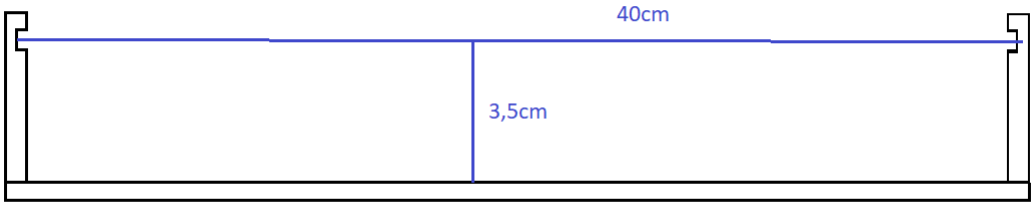


foto 23: doos

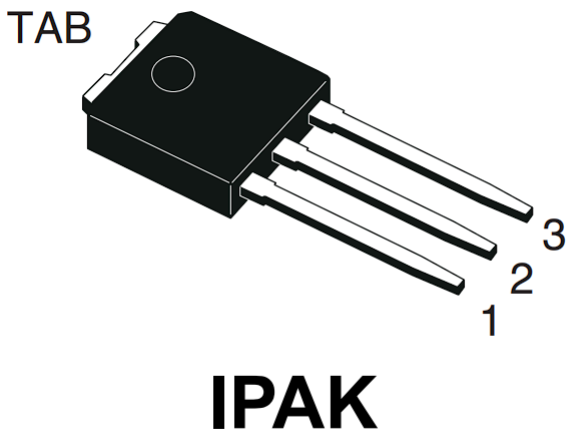
4.5.3 Licht scheider

foto 24: licht scheider

Dit onderdeelfiguur werd zestien keer gemaakt door een 3D-printer. Door de weggelaten stukjes in deze rechthoek pasten die allemaal in elkaar. De licht scheider werd dan tussen de LEDs geplaatst zodat het licht van het ene vakje niet ook het andere vakje kan oplichten.

5 Algemene bespreking mosfets

5.1 N-channel mosfets



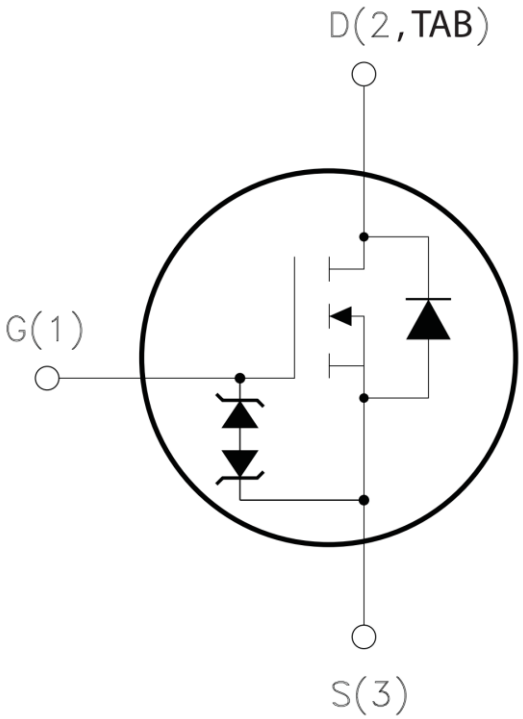
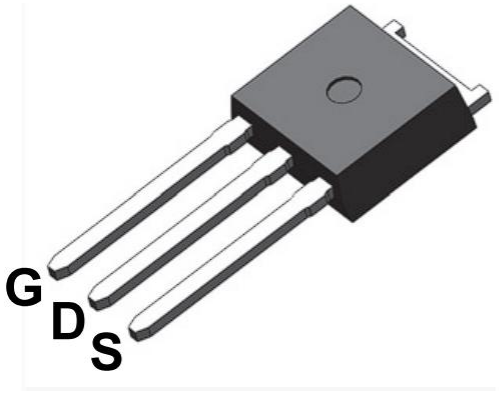


foto 25: N-channel mosfet

foto 26: N-channel mosfet schema

De STU6N65M2 MOSFET is gekozen vanwege de Gate-drempelspanning van 2-4 volt, wat perfect is om te schakelen met de 3,3V-uitgangen van de Raspberry Pi Pico W. De schakeltijd van de MOSFET is 26 ns om volledig geleidend te worden en 26,5 ns om van geleidend naar volledig sper over te gaan.

5.2 P-channel mosfets



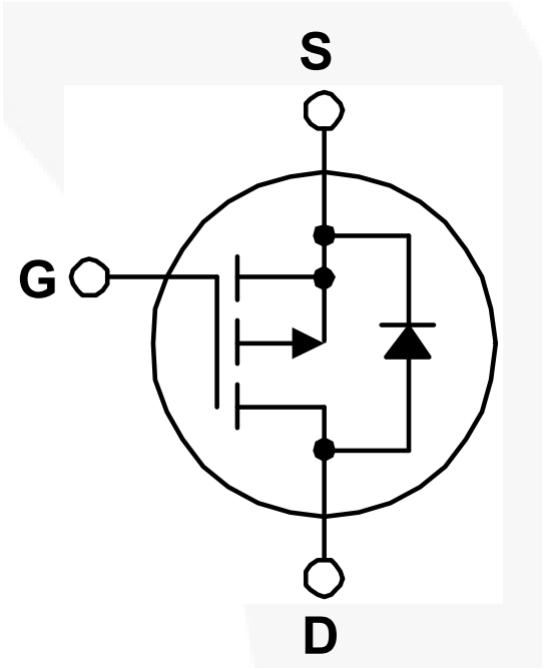


foto 27: P-channel mosfet

foto 28: P-channel mosfet schema

De FQU8P10TU MOSFET is gekozen vanwege de Gate-drempelspanning van -2 tot -4 volt, wat perfect is om te schakelen met de 3,3V-uitgangen van de Raspberry Pi Pico W. De schakeltijd van de MOSFET is 121 ns om volledig geleidend te worden en 55 ns om van geleidend naar volledig sper over te gaan.

6 LEDs aansturen

6.1 Probleemstelling

Als ik de LEDs allemaal tegelijk zou aansturen, zouden er een paar problemen komen door het gebruik van het coördinatensysteem. Zoals te zien op de volgende tabel zullen geen diagonale LEDs aangezet kunnen worden.

mosfet X / mosfet Y	1	1
0	1	1
0	1	1

Hier wou ik alle groene vakjes laten oplichten. Maar als ik dat tegelijk zou laten doen zouden de vakjes in het lichtblauw ook mee oplichten.

6.2 Oplossing

Om dit probleem op te lossen zal ik “scannen”. Dit betekent dat ik de mosfets op de Y-as één per één zal aansturen, waar ik dan alle mosfets op de X-as per rij zal aansturen. Dat zal er zo uitzien:

T1:

mosfet X / mosfet Y	1	0
1	0	0

0	1	0

T2:

mosfet X / mosfet Y	0	1
0	0	1
1	0	0

Dit wordt continu gedaan waardoor de LEDs maar de helft van de tijd aan zullen staan. Maar aangezien deze LEDs zodanig snel aan en uit worden geschakeld kan dit niet gezien worden met het blote oog.

Dit principe wordt toegepast op grotere schaal. Op het schaakbord zullen dan in plaats van twee rijen, acht rijen gescand worden.

7 Software

7.1 Stockfish

7.1.1 Chess engine

Een chess engine is een soort database die de populairste en de beste zetten bijhoudt in elk mogelijk schaak scenario. Zelf een chess engine schrijven is onbegonnen werk, aangezien er 318.979.564.000 stellingen zijn na de eerste vier zetten. En maar liefst 169.518.829.100.544.000.000.000.000.000 na het spelen van tien zetten.

Daarom heb ik besloten om alle zetten te sturen naar zo'n database, en op die manier de beste zet terug te krijgen voor de computer in het bord.

7.1.2 Stockfish

foto 29: Stockfish logo

Stockfish is de populairste chess engine. Deze is open-source, wat betekent dat iedereen de code van de chess engine kan consulteren. Ik heb voor deze chess engine gekozen omdat deze het best gedocumenteerd is.

De grondlegger van Stockfish is Marco Costalba. Hij startte met Stockfish in november 2008 door een andere open-source chess engine, Glaurung 2.1 (geschreven door dhr. Tord Romstad) te gebruiken als basis voor Stockfish. Deze code heeft hij gekopieerd en aangepast, en zo was de eerste versie van Stockfish een realiteit.

Nu wordt Stockfish wekelijks geüpdatet. Zo blijft Stockfish de beste en de populairste chess engine. Veel populaire schaak-websites zoals chess.com of lichess.org gebruiken Stockfish om hun zetten te berekenen.

(Yang, n.d.)

7.2 Flowchart

7.2.1 Raspberry Pi Pico W

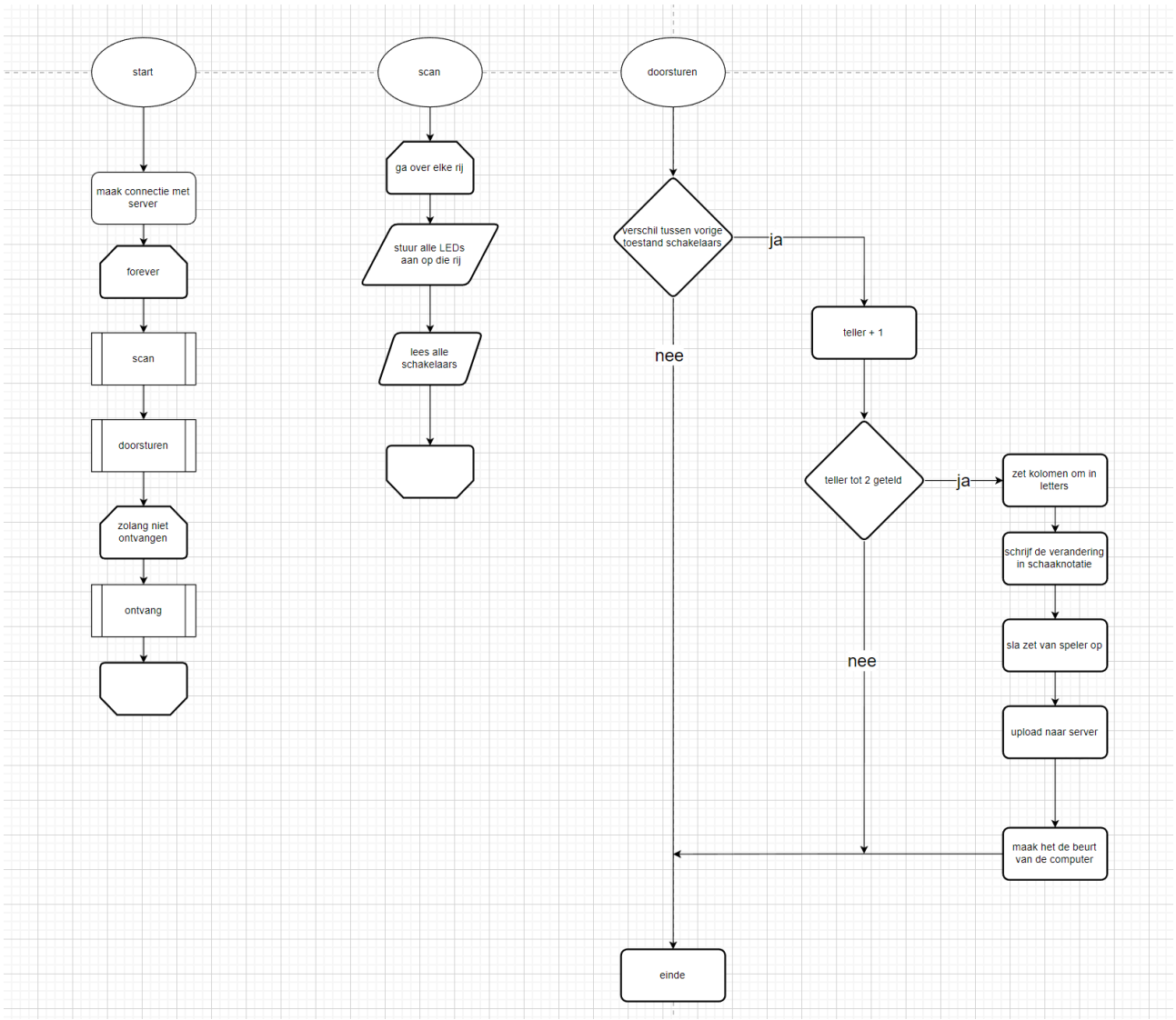


foto 30: flowchart Raspberry Pi Pico W, 1

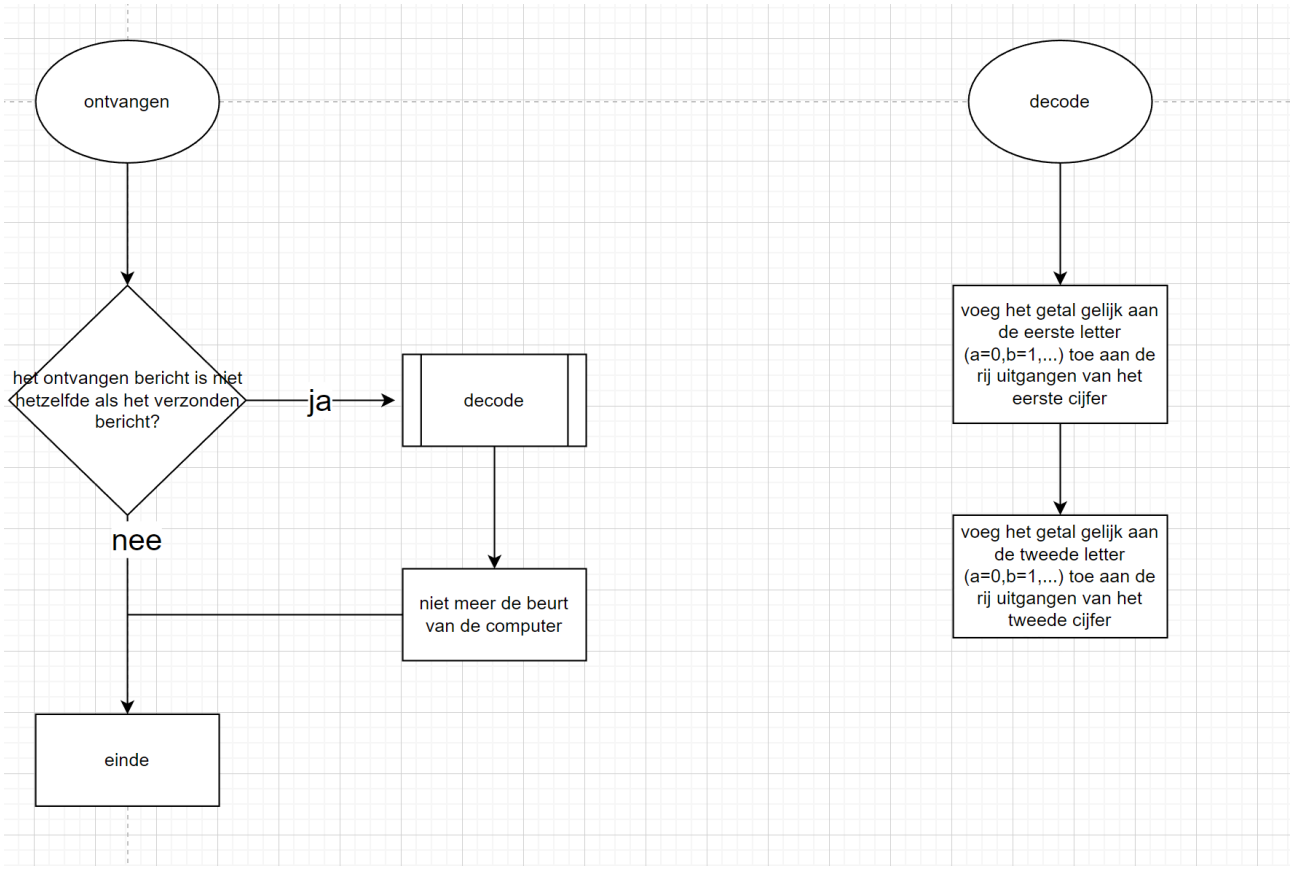


foto 31: flowchart Raspberry Pi Pico W, 2

7.2.2 PC

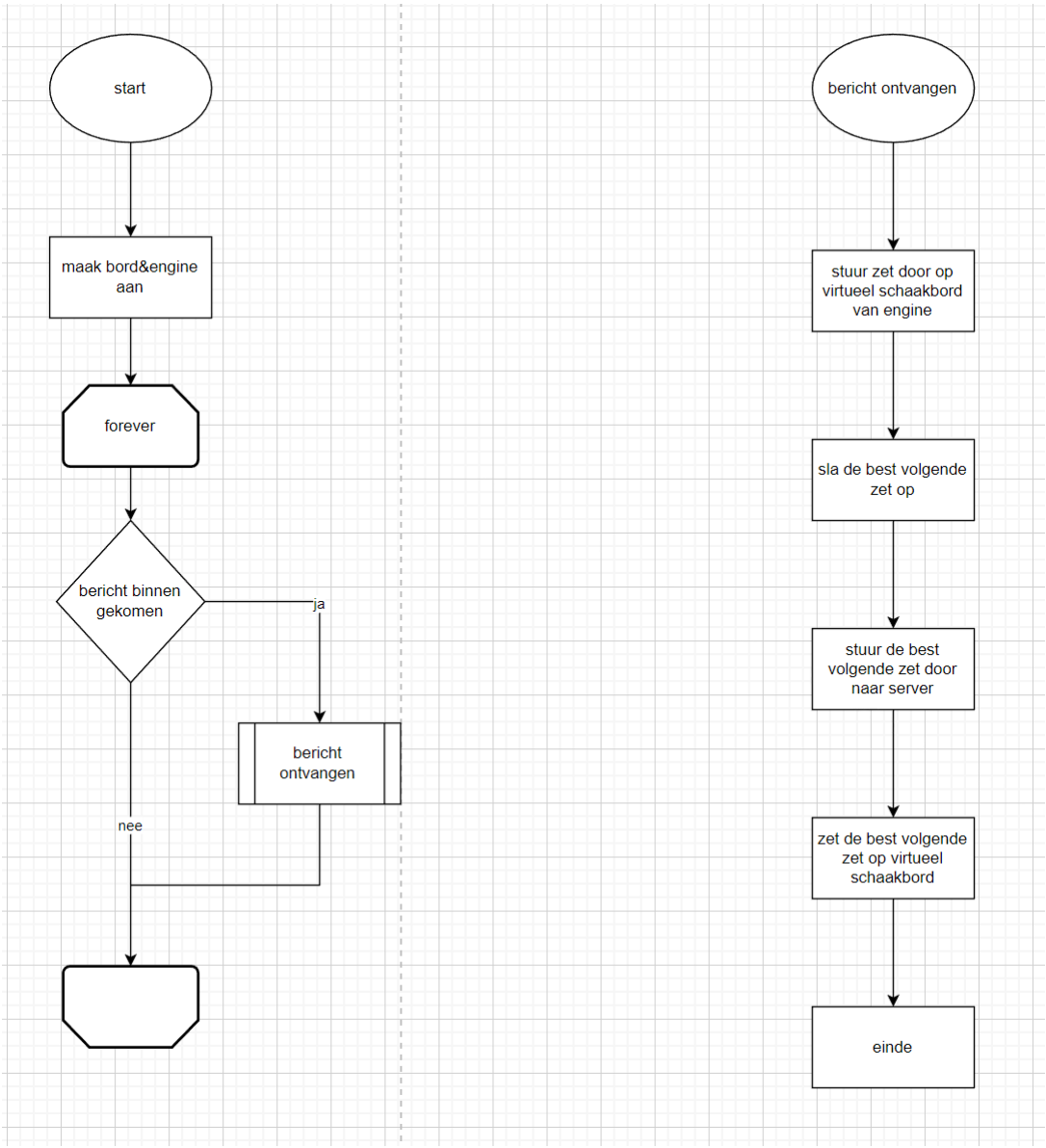


foto 32: flowchart pc

7.3 Code

7.3.1 Raspberry Pi Pico W

```
#voeg alle bibliotheken toe
from machine import Pin
from time import sleep
import network
import time
```

```
from umqtt.simple import MQTTClient


def reset():    #zet alle transistors in sper dus alle LEDs uit
    for i in range(8):
        nummers[i].value(1)
        letters[i].value(0)

reset()


#maak connectie met het internet
wlan = network.WLAN(network.STA_IF)
wlan.active(True)
wlan.connect("OnePlus","123456789") #naam en ww van hotspot
time.sleep(5)

mqtt_server = 'broker.hivemq.com' #website die ik gebruik om data op te slaan
client_id = 'art'   #wordt niet gebruikt maar moet toch megegeven worden
topic_pub = b'E-chess' #onderwerp waar ik mijn zet naar toe stuur
topic_sub = b'E-chess' #onderwerp waar de zet van de computer op komt


#declare alle pinnen als ingangen of uitgangen
nummers = [Pin(0, Pin.OUT), Pin(1, Pin.OUT), Pin(2, Pin.OUT), Pin(3, Pin.OUT),
Pin(4, Pin.OUT), Pin(5, Pin.OUT), Pin(6, Pin.OUT), Pin(7, Pin.OUT),]
letters = [Pin(15, Pin.OUT), Pin(14, Pin.OUT), Pin(13, Pin.OUT), Pin(12, Pin.OUT),
Pin(11, Pin.OUT), Pin(10, Pin.OUT), Pin(9, Pin.OUT), Pin(8, Pin.OUT)]
ingangen = [Pin(26, Pin.IN), Pin(22, Pin.IN), Pin(21, Pin.IN), Pin(20, Pin.IN),
Pin(19, Pin.IN), Pin(18, Pin.IN), Pin(17, Pin.IN), Pin(16, Pin.IN)]

#variable
counter = 0 #telt veranderingen in bord
zetSpeler =''#houdt zet van de speler bij
lastSend = ''#laatst gezonden zet
computerTurn = False #wie zijn beurt


out = [[],[],[],[],[],[],[],[]]     #houdt bij welke LEDs aan moeten

oldSwitches = [     #de positie van alle stukken na het meten
    [1,1,1,1,1,1,1,1],
    [1,1,1,1,1,1,1,1],
    [0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0],
```

```
    [0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0]
    ]
switches = [        #de positie van alle stukken na het meten
    [0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0]
    ]



def decode(msg):#ontcijfert de schaaknotatie (bv: e2e4)
    global out      #gebruik de varaiable out die al bestaat
    reset()             #zet alle LEDs uit
    out = [[],[],[],[],[],[],[],[]]

    #voegt het nummer van de led aan de juiste rij toe
    #het nummer wordt -1 gedaan omdat lijsten starten op 0 en schaaknotatie op 1
    if msg[0] == 'a':
        out[int(msg[1])-1].append(0)
    if msg[0] == 'b':
        out[int(msg[1])-1].append(1)
    if msg[0] == 'c':
        out[int(msg[1])-1].append(2)
    if msg[0] == 'd':
        out[int(msg[1])-1].append(3)
    if msg[0] == 'e':
        out[int(msg[1])-1].append(4)
    if msg[0] == 'f':
        out[int(msg[1])-1].append(5)
    if msg[0] == 'g':
        out[int(msg[1])-1].append(6)
    if msg[0] == 'h':
        out[int(msg[1])-1].append(7)

    if msg[2] == 'a':
        out[int(msg[3])-1].append(0)
    if msg[2] == 'b':
        out[int(msg[3])-1].append(1)
    if msg[2] == 'c':
```

```
            out[int(msg[3])-1].append(2)
        if msg[2] == 'd':
            out[int(msg[3])-1].append(3)
        if msg[2] == 'e':
            out[int(msg[3])-1].append(4)
        if msg[2] == 'f':
            out[int(msg[3])-1].append(5)
        if msg[2] == 'g':
            out[int(msg[3])-1].append(6)
        if msg[2] == 'h':
            out[int(msg[3])-1].append(7)


def sub_cb(topic, msg): #ontvangt de zet van de computer
    global computerTurn #gebruikt de bestaande variable
    global lastSend     #gebruikt de bestaande variable
    msg = msg.decode('utf-8')   #maakt van een binary string een gewone string
    if msg != lastSend: #kijkt ofdat het niet de zet is van de speler

        decode(msg)
        computerTurn = False #maakt een einde aan de beurt van de pc


def mqtt_connect(): #connecteerd met de server waar data op komt
    client = MQTTClient(client_id, mqtt_server, keepalive=60)
    client.set_callback(sub_cb)
    client.connect()
    print('Connected to %s MQTT Broker'%(mqtt_server))
    return client

def reconnect():    #herconnect met de server als het mislukt is
    print('Failed to connect to the MQTT Broker. Reconnecting...')
    machine.reset()

try:        #blijf proberen connecteren
    client = mqtt_connect()
except OSError as e:
    reconnect()




def scan():         #lees alle ingangen en stuur alle uitgangen
    for i in range(8):
        nummers[i].value(0) #ga over elke schakelaar op de Y-as
        for j in range(8):
```

```
            if j in range(len(out[i])): #als er een led aan moet op die rij, steek
die aan voor 5ms
                letters[out[i][j]].value(1)
                sleep(0.005)
                letters[out[i][j]].value(0)

            switches[i][j] = ingangen[j].value()    #lees alle ingangen van die rij
en sla ze op

        nummers[i].value(1) #zet schakelaar van de rij weer uit



def sendMove(): #stuur de zet van de speler naar de server
    global counter  #gebruikt de bestaande variable
    global zetSpeler #gebruikt de bestaande variable
    global computerTurn #gebruikt de bestaande variable
    global lastSend  #gebruikt de bestaande variable
    for i in range(8):
            for j in range(8):

                if(oldSwitches[i][j] != switches[i][j]):    #ga over elke
schakelaar en kijk ofdat er verandering is
                    counter +=1     #een stuk is opgehoffen
                    if counter == 1:#zet eerste deel om in schaaknotatie
                        if j == 0:
                            zetSpeler += 'a'
                        if j == 1:
                            zetSpeler += 'b'
                        if j == 2:
                            zetSpeler += 'c'
                        if j == 3:
                            zetSpeler += 'd'
                        if j == 4:
                            zetSpeler += 'e'
                        if j == 5:
                            zetSpeler += 'f'
                        if j == 6:
                            zetSpeler += 'g'
                        if j == 7:
                            zetSpeler += 'h'

                        zetSpeler += str(i+1)

                    if counter == 2:#zet tweede deel om in schaaknotatie
                        if j == 0:
```

```
                        zetSpeler += 'a'
                    if j == 1:
                        zetSpeler += 'b'
                    if j == 2:
                        zetSpeler += 'c'
                    if j == 3:
                        zetSpeler += 'd'
                    if j == 4:
                        zetSpeler += 'e'
                    if j == 5:
                        zetSpeler += 'f'
                    if j == 6:
                        zetSpeler += 'g'
                    if j == 7:
                        zetSpeler += 'h'
                    zetSpeler += str(i+1)

                    #maak het de beurt van de pc, lees geen schakelaars meer en
stuur geen LEDs meer
                    computerTurn = True

                    topic_msg = str(zetSpeler)  sla het bericht op
                    lastSend = topic_msg    #onthoud het verzonden bericht

                    #zet het bericht om naar een binairy string want data
sturen moet binair
                    topic_msg = topic_msg.encode('ascii')
                    client.publish(topic_pub, topic_msg)    #stuur het bericht


                    counter = 0     #reset de teller
                    zetSpeler = ''  #reset de zet


                oldSwitches[i][j] = switches[i][j]


while 1:
    scan()          #lees ingangen en stuur LEDs
    sendMove()      #stuur de zet naar pc
    while computerTurn:
        client.subscribe(topic_sub) #wacht tot er een bericht terug komt
```

7.3.2 Pc

```
#alle bibliotheken importeren
```

```
import paho.mqtt.client as paho
import chess
import chess.engine

#variable aanmaken
lastVal = '' #laatst gestuurde waarde
board = chess.Board() # maak een bord aan

#kies welke chess-engine je gaat gebruiken
engine =
chess.engine.SimpleEngine.popen_uci(r"C:\Users\artva\Downloads\stockfish_15.1_win_x
64_avx2\stockfish_15.1_win_x64_avx2\stockfish-windows-2022-x86-64-avx2.exe")

#hiermee lezen en zenden we data naar de server
client = paho.Client()



def on_message(client, userdata, msg): #voert dit uit als er een bericht wordt
ontvangen

    global lastVal
    msg = msg.payload.decode('utf-8') #zet het bericht om van een binairy string
naar een string


    if msg != str(lastVal): #kijkt ofdat het bericht van de raspberry pi pico W
komt
        print(msg)
        board.push_san(msg) #stuurt zet van speler naar de engine

        result = engine.play(board, chess.engine.Limit(time=0.1))
        board.push(result.move) #neem de beste zet en zet hem op het bord van de
engine

        lastVal = result.move #sla het bericht op
        (rc, mid) = client.publish('E-chess', str(result.move), qos=2) #stuur zet
van pc naar schaakbord
        print(lastVal)




client.on_message = on_message #voer de functie on_message uit als er een bericht
binnen komt
```

```
client.connect('broker.mqttdashboard.com', 1883) #connecteer met de server
client.subscribe('E-chess', qos=2) #luister naar het onderwerp E-chess
client.loop_start() #blijf luisteren
```

8 Berekeningen

De stroom door de LED mag nominaal maar 20mA zijn. Er zullen telkens maximaal 2 LEDs tegelijk aanstaan. Elke keer dat een LED aanstaat, wacht mijn programma 5ms waardoor de LED zo goed als half de tijd aanstaat.

$$I_{\text{led}} = \frac{U_{\text{bron}} - U_{\text{led}}}{R} = \frac{5V - 3V}{120\Omega} = 16mA$$

9 Fouten

Als lerende student heb ik doorheen dit leerproces natuurlijk ook fouten gemaakt. Deze heb ik er natuurlijk zo goed mogelijk proberen uithalen. Hierna bespreek ik welke fouten ik gemaakt heb en welke oplossing ik ervoor gevonden heb.

9.1 Schema tekenen

Bij het tekenen van mijn schema ondervond ik een paar problemen:

- Ik wou eerst alleen maar P-channel mosfets gebruiken, maar dat bleek uiteindelijk niet te werken toen ik het testte.
- Toen ik een P-channel en een N-channel mosfet apart gebruikte was er geen probleem, maar toen ik die in serie probeerde te plaatsen lukte dat niet. Uiteindelijk heb ik geleerd dat:
    - ik de source van de P-channel mosfet aan de +5V moest hangen
    - ik de source van de N-channel mosfet aan de ‘ground’ moest hangen

9.2 Stick verloren

Dit is misschien niet per se een “fout” maar nadat ik mijn printplaat volledig getekend had, ben ik de stick waarop al mijn documenten stonden verloren. Gelukkig had ik nog een back-up van een paar weken terug en zat alles nog fris in mijn geheugen. Dit heeft me toch twee weken extra gekost.

9.3 Mosfet footprint

Dit heb ik pas later ontdekt. Nadat mijn printplaat gesoldeerd was, begon ik aan het schrijven van de software. Ik ontdekte dat ik verkeerde footprints voor mijn mosfets had gebruikt bij het tekenen van de printplaat waardoor de pinnen op mijn bord niet overeenkwamen met de pinnen van mijn mosfets.

Dit is dan na een paar dagen zoeken toch opgelost door alle koperbanen die naar de poten van de mosfets liepen te onderbreken en opnieuw te bekabelen.

9.4 Raspberry Pi Pico W opslagruimte

Toen alle mosfets werkten, wou ik onmiddellijk beginnen programmeren. Het originele plan was om alle berekeningen lokaal te laten gebeuren. Maar ik liep vast aangezien ik voor de zetten van de computer een schaak engine nodig zou hebben, en die was te groot voor de Raspberry Pi Pico W. Hierdoor moest een oplossing gezocht worden.

Gelukkig heeft de Raspberry Pi Pico W een wifi-module heeft. Deze module zou ik dan kunnen gebruiken om al de zetten van de computer door een externe pc te laten berekenen.

10 Handleiding

Als speler zal jij altijd wit zijn.

Belangrijk is ervoor te zorgen dat het spel voldoende is opgeladen. Dit doe je door de powerbank in het bord op te laden.

Nadat de powerbank volledig opgeladen is, kan je beginnen spelen.

Start met het bord aan te schakelen. Nu kan je de eerste zet zetten.

Nadat je jouw zet hebt gezet, zal het bord een zet zetten. De zet die het bord maakt, zal worden getoond door middel van twee LEDs.

Als je de twee LEDs ziet, moet je het stuk dat op één van de LEDs staat verplaatsen naar de andere LED.

Speel het spel verder tot iemand (speler of computer van het bord) schaakmat staat.

Als het spel gedaan is, schakel het bord uit en zet alle stukken terug op de juiste plaatsen.

Besluit

Wat heb ik geleerd?

Ik heb geleerd dat ik goed moet letten op de footprints als ik een printplaat teken. Ik heb geleerd hoe ik footprints moet vinden en hoe ik ze zelf teken.

Ik heb ook geleerd dat ik beter eerst de berekeningen doe om latere fouten te vermijden.

Ook heb ik geleerd dat ik beter moet plannen voor ik dingen koop zoals bekijken hoe groot mijn programma zal zijn en zeker zijn dat het op de gekozen chip past.

Als ik meer tijd zou hebben

Als ik meer tijd zou hebben dan zou ik de laatste fouten eruit halen, en misschien een scherm en wat extra knoppen toevoegen zodat promoveren beter lukt.

Als ik **veel** meer tijd zou hebben dan zou ik de zetten van de computer laten gebeuren met een magneet, die de stukken vanzelf verplaatst, i.p.v. LEDs te gebruiken, maar dat is onmiddellijk veel moeilijker.

Ervaring

Ik vond het leuk om te ervaren wat het is om van niets, iets helemaal zelf te maken. Ik heb veel geleerd uit deze ervaring, vooral ook wat het is om zelfstandig te werken. Natuurlijk had ik altijd een vangnet van leerkrachten die klaar stonden om mij te helpen, maar het was toch de bedoeling zo zelfstandig mogelijk te werken.

Wat zou ik anders gedaan hebben

Ik zou een normale Raspberry Pi gebruikt hebben i.p.v. een Raspberry Pi Pico W. omdat ik dan niet zou moeten connecteren met een pc om de zetten te berekenen.

Bibliografie

Yang, D. (sd). *About*. Opgehaald van Stockfish: https://stockfishchess.org/about/

Bijlage

FAIRCHILD SEMICONDUCTOR®

November 2013

FQD8P10 / FQU8P10

P-Channel QFET® MOSFET

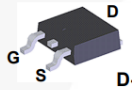
-100 V, -6.6 A, 530 mΩ

Description

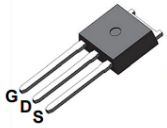
This P-Channel enhancement mode power MOSFET is produced using Fairchild Semiconductor's proprietary planar stripe and DMOS technology. This advanced MOSFET technology has been especially tailored to reduce on-state resistance, and to provide superior switching performance and high avalanche energy strength. These devices are suitable for switched mode power supplies, audio amplifier, DC motor control, and variable switching power applications.

Features

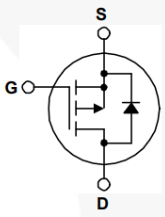
- -6.6 A, -100 V, $R_{DS(on)}$ = 530 mΩ (Max) @ V_{GS} = -10 V, I_D = -3.3 A
- Low Gate Charge (Typ. 12 nC)
- Low Crss (Typ. 30 pF)
- 100% Avalanche Tested



D-PAK



I-PAK



Absolute Maximum Ratings T_C = 25°C unless otherwise noted.

Symbol	Parameter			FQD8P10TM / FQU8P10TU	Unit
V_{DSS}	Drain-Source Voltage			-100	V
I_D	Drain Current	- Continuous (T_C = 25°C)		-6.6	A
		- Continuous (T_C = 100°C)		-4.2	A
I_{DM}	Drain Current	- Pulsed	(Note 1)	-26.4	A
V_{GSS}	Gate-Source Voltage			± 30	V
E_{AS}	Single Pulsed Avalanche Energy		(Note 2)	150	mJ
I_{AR}	Avalanche Current		(Note 1)	-6.6	A
E_{AR}	Repetitive Avalanche Energy		(Note 1)	4.4	mJ
dv/dt	Peak Diode Recovery dv/dt		(Note 3)	-6.0	V/ns
P_D	Power Dissipation (T_A = 25°C) *			2.5	W
	Power Dissipation (T_C = 25°C)			44	W
		- Derate above 25°C		0.35	W/°C
T_J, T_{STG}	Operating and Storage Temperature Range			-55 to +150	°C
T_L	Maximum lead temperature for soldering purposes, 1/8" from case for 5 seconds			300	°C

Thermal Characteristics

Symbol	Parameter	FQD8P10TM FQU8P10TU	Unit
$R_{\theta JC}$	Thermal Resistance, Junction to Case, Max.	2.84	°C/W
$R_{\theta JA}$	Thermal Resistance, Junction to Ambient (Minimum Pad of 2-oz Copper), Max.	110	
	Thermal Resistance, Junction to Ambient (*1 in^2 Pad of 2-oz Copper), Max.	50	

foto 33: P-channel mosfet, datasheet 1

Package Marking and Ordering Information

Part Number	Top Mark	Package	Packing Method	Reel Size	Tape Width	Quantity
FQD8P10TM	FQD8P10	D-PAK	Tape and Reel	330 mm	16 mm	2500 units
FQU8P10TU	FQU8P10	I-PAK	Tube	N/A	N/A	70 units

Electrical Characteristics T_C = 25°C unless otherwise noted.

Symbol	Parameter	Test Conditions	Min	Typ	Max	Unit
Off Characteristics						
BV_{DSS}	Drain-Source Breakdown Voltage	V_{GS} = 0 V, I_D = -250 μA	-100	--	--	V
ΔBV_{DSS} / ΔT_J	Breakdown Voltage Temperature Coefficient	I_D = -250 μA, Referenced to 25°C	--	-0.1	--	V/°C
I_{DSS}	Zero Gate Voltage Drain Current	V_{DS} = -100 V, V_{GS} = 0 V	--	--	-1	μA
		V_{DS} = -80 V, T_C = 125°C	--	--	-10	μA
I_{GSSF}	Gate-Body Leakage Current, Forward	V_{GS} = -30 V, V_{DS} = 0 V	--	--	-100	nA
I_{GSSR}	Gate-Body Leakage Current, Reverse	V_{GS} = 30 V, V_{DS} = 0 V	--	--	100	nA
On Characteristics						
$V_{GS(th)}$	Gate Threshold Voltage	V_{DS} = V_{GS}, I_D = -250 μA	-2.0	--	-4.0	V
$R_{DS(on)}$	Static Drain-Source On-Resistance	V_{GS} = -10 V, I_D = -3.3 A	--	0.41	0.53	Ω
g_{FS}	Forward Transconductance	V_{DS} = -40 V, I_D = -3.3 A	--	4.1	--	S
Dynamic Characteristics						
C_{iss}	Input Capacitance	V_{DS} = -25 V, V_{GS} = 0 V, f = 1.0 MHz	--	360	470	pF
C_{oss}	Output Capacitance		--	120	155	pF
C_{rss}	Reverse Transfer Capacitance		--	30	40	pF
Switching Characteristics						
$t_{d(on)}$	Turn-On Delay Time	V_{DD} = -50 V, I_D = -8.0 A, R_G = 25 Ω (Note 4)	--	11	30	ns
t_r	Turn-On Rise Time		--	110	230	ns
$t_{d(off)}$	Turn-Off Delay Time		--	20	50	ns
t_f	Turn-Off Fall Time		--	35	80	ns
Q_g	Total Gate Charge	V_{DS} = -80 V, I_D = -8.0 A, V_{GS} = -10 V (Note 4)	--	12	15	nC
Q_{gs}	Gate-Source Charge		--	3.0	--	nC
Q_{gd}	Gate-Drain Charge		--	6.4	--	nC
Drain-Source Diode Characteristics and Maximum Ratings						
I_S	Maximum Continuous Drain-Source Diode Forward Current		--	--	-6.6	A
I_{SM}	Maximum Pulsed Drain-Source Diode Forward Current		--	--	-26.4	A
V_{SD}	Drain-Source Diode Forward Voltage	V_{GS} = 0 V, I_S = -6.6 A	--	--	-4.0	V
t_{rr}	Reverse Recovery Time	V_{GS} = 0 V, I_S = -8.0 A, dI_F / dt = 100 A/μs	--	98	--	ns
Q_{rr}	Reverse Recovery Charge		--	0.35	--	μC

Notes:
1. Repetitive rating : pulse-width limited by maximum junction temperature.
2. L = 5.2 mH, I_{AS} = -6.6 A, V_{DD} = -25 V, R_G = 25 Ω, starting T_J = 25°C.
3. I_{SD} ≤ -8.0 A, di/dt ≤ 300 A/μs, V_{DD} ≤ BV_{DSS}, starting T_J = 25°C.
4. Essentially independent of operating temperature.

foto 34: P-channel mosfet, datasheet 2

life.augmented

STF6N65M2, STP6N65M2, STU6N65M2

N-channel 650 V, 1.2 Ω typ., 4 A MDmesh™ M2 Power MOSFETs in TO-220FP, TO-220 and IPAK packages

Datasheet - preliminary data

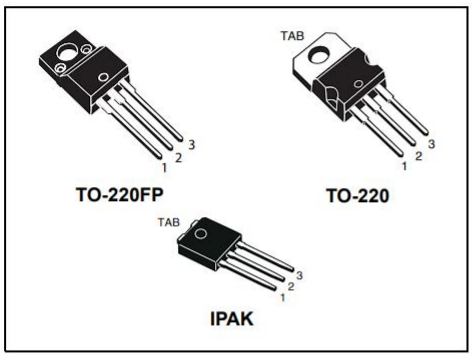
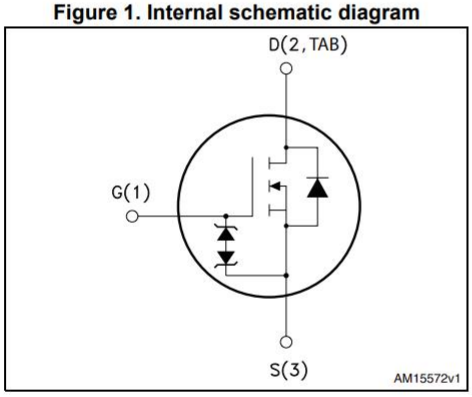


Figure 1. Internal schematic diagram



Features

Order codes	V_{DS}	$R_{DS(on)}$ max	I_D
STF6N65M2	650 V	1.35 Ω	4 A
STP6N65M2			
STU6N65M2			

- Extremely low gate charge
- Excellent output capacitance (C_{OSS}) profile
- 100% avalanche tested
- Zener-protected

Applications

- Switching applications

Description

These devices are N-channel Power MOSFETs developed using MDmesh™ M2 technology. Thanks to their strip layout and improved vertical structure, the devices exhibit low on-resistance and optimized switching characteristics, rendering them suitable for the most demanding high efficiency converters.

Table 1. Device summary

Order codes	Marking	Package	Packaging
STF6N65M2	6N65M2	TO-220FP	Tube
STP6N65M2		TO-220	
STU6N65M2		IPAK	

foto 35: N-channel mosfet, datasheet 1

2 Electrical characteristics

(T_C = 25 °C unless otherwise specified)

Table 5. On /off states

Symbol	Parameter	Test conditions	Min.	Typ.	Max.	Unit
$V_{(BR)DSS}$	Drain-source breakdown voltage	V_{GS} = 0, I_D = 1 mA	650			V
I_{DSS}	Zero gate voltage drain current	V_{GS} = 0, V_{DS} = 650 V			1	µA
		V_{GS} = 0, V_{DS} = 650 V, T_C=125 °C			100	µA
I_{GSS}	Gate-body leakage current	V_{DS} = 0, V_{GS} = ± 25 V			±10	µA
$V_{GS(th)}$	Gate threshold voltage	V_{DS} = V_{GS}, I_D = 250 µA	2	3	4	V
$R_{DS(on)}$	Static drain-source on-resistance	V_{GS} = 10 V, I_D = 2 A		1.2	1.35	Ω

Table 6. Dynamic

Symbol	Parameter	Test conditions	Min.	Typ.	Max.	Unit
C_{iss}	Input capacitance	V_{GS} = 0, V_{DS} = 100 V, f = 1 MHz	-	226	-	pF
C_{oss}	Output capacitance		-	12.8	-	pF
C_{rss}	Reverse transfer capacitance		-	0.65	-	pF
$C_{oss\ eq.}$ (1)	Equivalent output capacitance	V_{GS} = 0, V_{DS} = 0 to 520 V	-	114	-	pF
R_G	Intrinsic gate resistance	f = 1 MHz open drain	-	6.5	-	Ω
Q_g	Total gate charge	V_{DD} = 520 V, I_D = 4 A, V_{GS} = 10 V (see *Figure 8*)	-	9.8	-	nC
Q_{gs}	Gate-source charge		-	1.7	-	nC
Q_{gd}	Gate-drain charge		-	4	-	nC

1. $C_{oss\ eq.}$ is defined as a constant equivalent capacitance giving the same charging time as C_{oss} when V_{DS} increases from 0 to 80% V_{DSS}

Table 7. Switching times

Symbol	Parameter	Test conditions	Min.	Typ.	Max.	Unit
$t_{d(on)}$	Turn-on delay time	V_{DD} = 325 V, I_D = 2 A, R_G = 4.7 Ω, V_{GS} = 10 V (see *Figure 15* and *Figure 20*)	-	19	-	ns
t_r	Rise time		-	7	-	ns
$t_{d(off)}$	Turn-off delay time		-	6.5	-	ns
t_f	Fall time		-	20	-	ns

foto 36: N-channel mosfet, datasheet 2

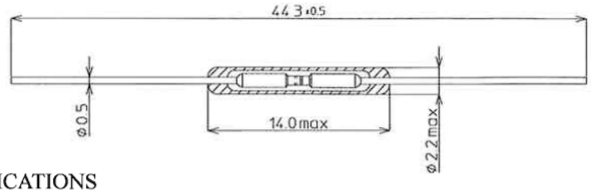
Table 8. Source drain diode

Symbol	Parameter	Test conditions	Min.	Typ.	Max.	Unit
I_{SD}	Source-drain current		-		4	A
I_{SDM} (1)	Source-drain current (pulsed)		-		16	A
V_{SD} (2)	Forward on voltage	I_{SD} = 4 A, V_{GS} = 0	-		1.6	V
t_{rr}	Reverse recovery time	I_{SD} = 4 A, di/dt = 100 A/µs V_{DD} = 60 V (see *Figure 17*)	-	260		ns
Q_{rr}	Reverse recovery charge		-	1.2		µC
I_{RRM}	Reverse recovery current		-	9.2		A
t_{rr}	Reverse recovery time	I_{SD} = 4 A, di/dt = 100 A/µs V_{DD} = 60 V, T_j = 150 °C (see *Figure 17*)	-	400		ns
Q_{rr}	Reverse recovery charge		-	1.84		µC
I_{RRM}	Reverse recovery current		-	9.1		A

1. Pulse width limited by safe operating area.
2. Pulsed: pulse duration = 300 µs, duty cycle 1.5%

foto 37: N-channel mosfet, datasheet 3

ORD324 REED SWITCH



I . SPECIFICATIONS

1-1. Electrical characteristics

1. Pull-in	10～50	AT	
2. Drop-out	5	AT	min
3. Contact resistance	100	mΩ	max
4. Breakdown voltage	250	DCV	min
5. Insulation resistance	10^{10}	Ω	min
6. Electrostatic capacitance	0.3	pF	max
7. Contact rating	10	W, VA	
8. Maximum switching voltage	200	DCV	
9. Maximum switching current	0.5	A	
10. Maximum carry current	1.0	A	

1-2. Operating characteristics

1. Operate time	0.4	ms	max
2. Bounce time	0.3	ms	max
3. Release time	0.05	ms	max
4. Resonant frequency	$5000^{\pm 400}$	Hz	
5. Maximum operating frequency	500	Hz	

1-3. Environmental characteristics

1. Shock	980(100G)	m/s^2	(6ms)
2. Vibration	490(50G)	m/s^2	(10～2000Hz)
3. Temperature range	-40～+125	℃	
4. Lead tensile strength	29.4(3kg)	N	

1-4. Life expectancy

1. 5Vdc-1mA; Resistance load	$1x10^8$		

II . STRUCTURE AND DIMENSIONS

1. Contact arrangement	Form A		
2. Contact material	Iridium		
3. Glass diameter	2.20 φ	mm	max
4. Glass length	14.0	mm	max
5. Overall length	$44.3^{\pm 0.5}$	mm	

DATE	REMARKS ON REVSES	DESIGNED	OKI SENSOR DEVICE CORPORATION	
		DRAWN	NOMENCLATURE	
		CHECKED	ORD324 REED SWITCH	
		APPROVED	REFERENCE	DRAWING NO.
		DATE OF ISSUE		

55138

foto 38: reed-switch datasheet