Motorslot

Door:
Lauren Debie
&
Toon Jansen

voorblad

Inhoudsopgave

1 Auteurs

Lauren

Afbeelding 1

Mijn naam is Lauren Debie. Ik ben afgestudeerd in 2019 in elektriciteit elektronica op het Technisch Instituut Sint-Paulus.

2de fase Bacheloropleiding Elektronica-ICT met als keuze optie Embedded electronics.

In mijn vrije tijd maak ik voor mezelf kleine elektronica projecten zoals een geluidsysteem van een oude radio set te maken en aan mijn brommer werken.

Toon

Afbeelding 2

Mijn naam is Toon Jansen, klasgenoot en full time vertaler voor alles wat Lauren schrijft. Ik ben ook afgestudeerd in elektriciteit elektronica aan het technisch instituut sint-Paulus "TISP" voor de vrienden. We zijn dus allebei van hetzelfde dorp en dezelfde school.

Als achtergrond kennis hebben we dus een goede 6 jaar aan middelbare opleiding gehad in elektronica. Een basiskennis aan programmeren door een GIP te maken met behulp van Arduino en in mijn vrije tijd houd ik me bezig met kleinere projecten te programmeren, zwemmen en een gezonde hoeveelheid uitgaan.

2 Inleiding

Voor mensen die met een brommer rijden doet er zich vaak een probleem voor. Namelijk dat vervelende gevoel dat je hebt als je niet zeker bent dat je brommer wel veilig op slot is. Dieven hebben slimmere trucjes bedacht om een brommer te stelen zoals het gewoonweg inladen in een aanhangwagen of vervoer bus. Een oplossing hier voor is echter een slot met een alarm. Maar met deze oplossing zijn er enkele praktische problemen. Het eerste en grootste probleem is dat er momenteel geen alarm op de markt is dat zowel een slot heeft dat het wiel blokkeert en een alarm systeem heeft. Verder heeft men bij een alarm ook het probleem dat men er altijd moet aan denken om dit op te zetten wat de meeste mensen wel eens durven vergeten. Het is hiervoor dat wij met ons project een oplossing wilden bieden. Dat deden we dan dus echter aan de hand van het maken van een wiel slot dat op de brommer vast gemonteerd staat waar ook een alarm op zit dat afgaat als een ongewenste persoon de brommer probeert te bewegen. Verder zit er ook nog een beveiliging in die er voor zorgt dat de brommer onmogelijk kan opgezet worden als het alarm opstaat of verwijdert/vernield word door de dief. Het op en af zetten van dit alarm zal gebeuren aan de hand van een key-card die enkel in de broekzak van de gebruiker zit of door de brommersleutel zelf.

3 Doelstelling

Het maken van een brommer alarm met volgende specificaties:

- Een wiel lock dat het achterwiel kan blokkeren
- Een alarm dat afgaan wanneer een dief de brommer probeert te stelen

- Een systeem dat ervoor zorgt dat de brommer onmogelijk te starten is wanneer het alarm op staat.
- Een key-card die het alarm op en af kan zette
- Het alarm moet ook met de brommer sleutels zelf op en af gezet moeten worden.
- Een indicator dat laat zien dat het alarm op of af staat.

4 Resultaat

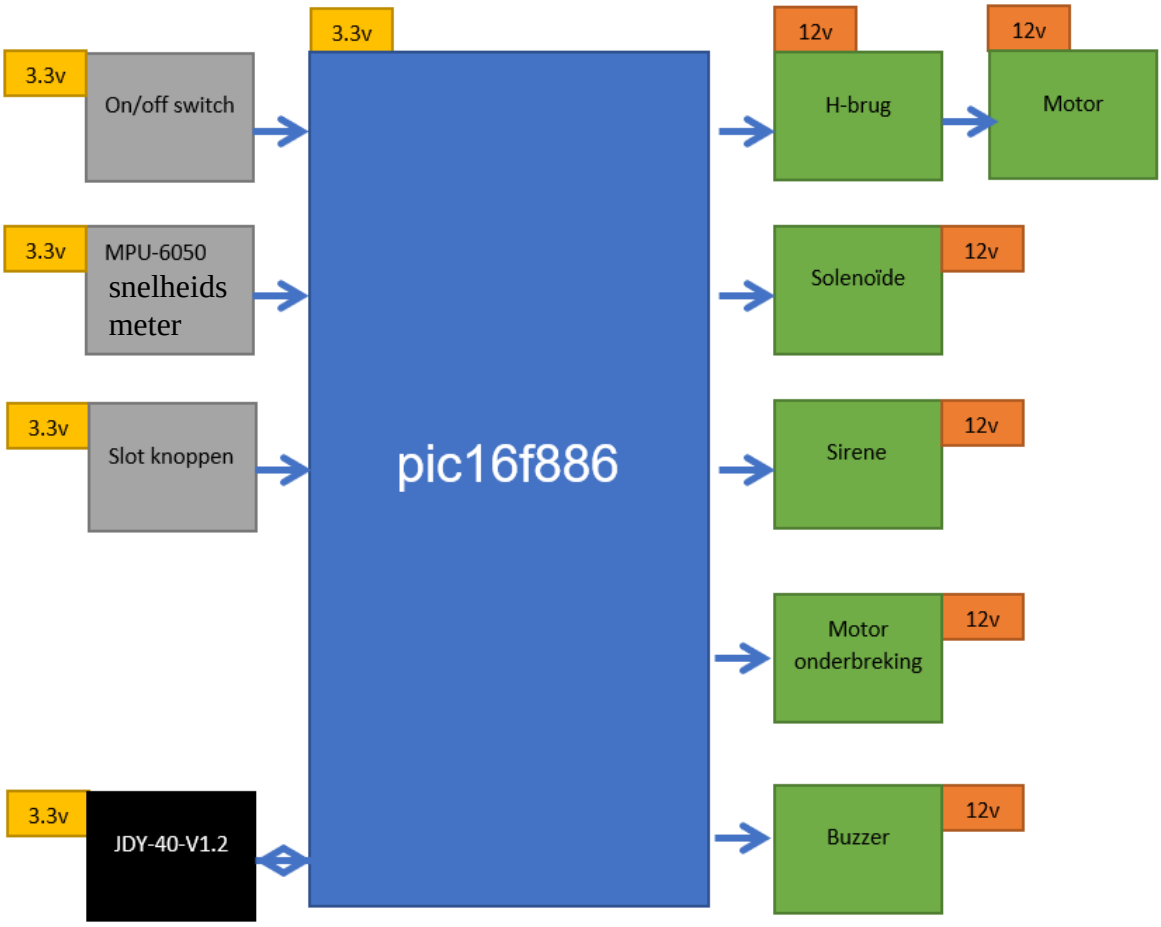
Bij het schrijven van dit onderdeel op 21/05, de laatste dag dat er nog aan het verslag kan gewerkt worden, zijn we tot het volgende resultaat gekomen:

Alle componenten zijn getest en werken. Na 3 hertekeningen en bestellingen komen we tot de conclusie dat alle schakelingen werken zoals gewenst. Als ook alle functies en testprogramma's werken zoals gewenst. Het enige voorval wat ons tegen houd is een onbekend probleem aan poort B waardoor deze niet genoeg spanning kan leveren voor een logisch hoog signaal te maken als men 2 pinnen hoog wilt maken. Dit is niet het geval bij de testkits en laat ons dan ook met verwondering voor wat hier de oorzaak van kan zijn. We hebben dit probleem kunnen omzeilen door steeds slechts 1 pin hoog te houden terwijl de anderen laag blijven. Het toeval wil dat we nooit meer dan 1 pin tegelijk een logisch hoog signaal moeten laten uitsturen. (Eindelijk eens een tegenslag die wel te omzeilen valt)

Het resultaat is dus een werkend draadloos motorslot dat alarm geeft iedere keer dat de key-card niet in de buurt is om te signaleren dat de motor gestolen wordt.

5 Blokschema

5.1 Motor moederbord



Figuur 1

Beschrijving:

- **PIC16F886:**

  Dit is de microcontroller die we in C programmeren en als brein van ons project gebruiken.

- **On/off switch:**

  Dit is een schakelaar die de voeding van het alarm kan op en af zetten. Deze schakelaar mag alleen uitgelezen worden wanneer de brommer afstaat en niet op slot is.

- **MPU-6050:**

  Dit is een accelerometer deze dient om te kijken dat de brommer in beweging is.

- **Slot knoppen:**

  Deze dienen om te kijken dat het wielslot open of dicht is.

- **JDY-40-V1.2**

  RF transceiver wordt gebruikt om te communiceren met de keycard voor het slot op of af te zetten.

- **H-brug:**

  Deze stuurt de motor van het slot aan.

- **DC-Motor:**

  Deze dient om het wiel slot open en dicht te doen.

- **Solenoïde:**

  Dit is een bescherming zodat het wielslot niet onverwacht open of dicht kan gaan.

- **Sirene**

  Deze maakt lawaai wanneer een ongewenste persoon de brommer wil verplaatsen.

- **Motor onderbreking:**

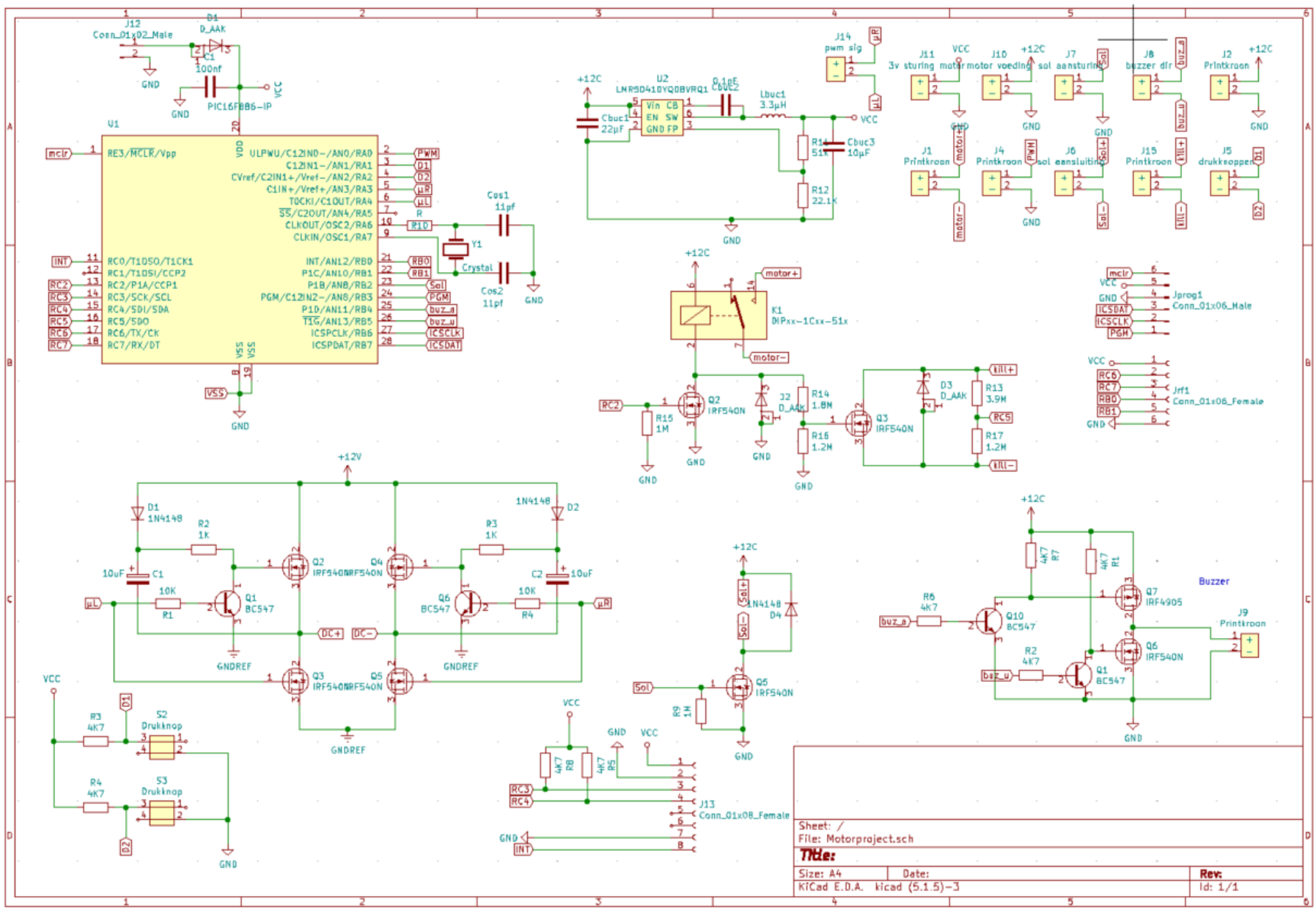
  Deze is nodig om de brommer te kunnen starten zonder de sleutel als men het alarm unlocked heeft.

5.2 Key-card

Figuur 2

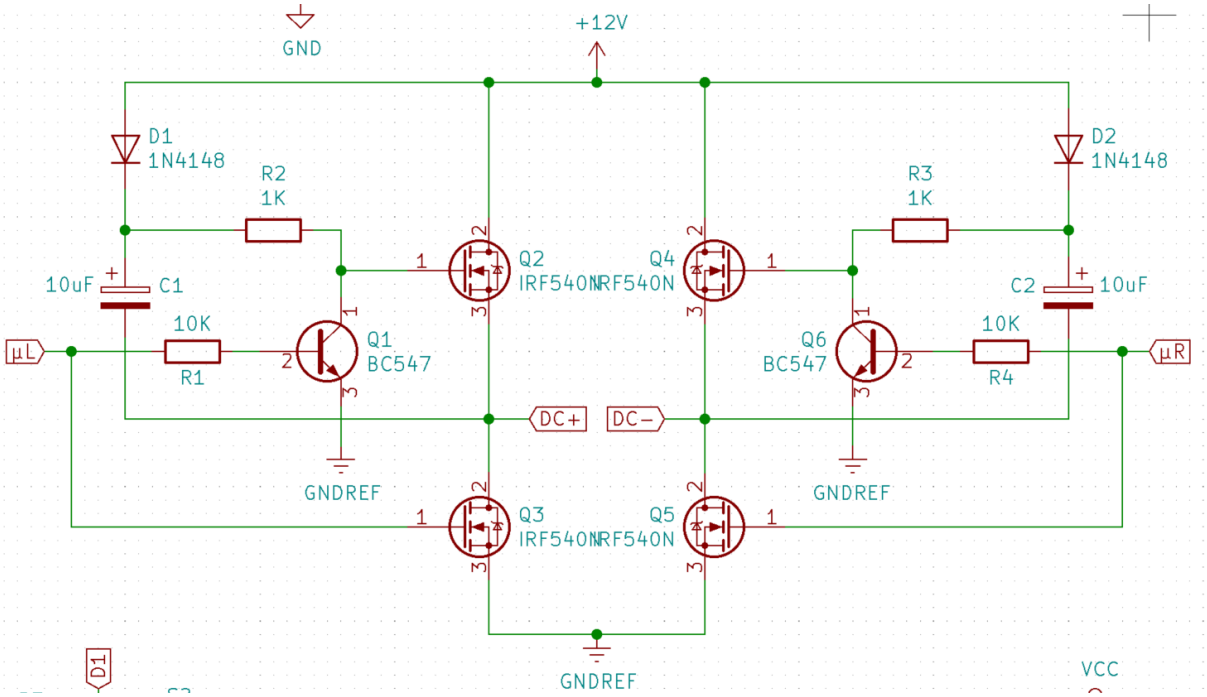
6 Hardware

6.1 Schema:



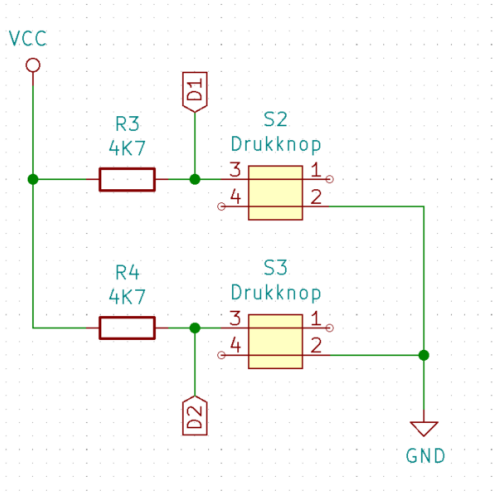
Figuur 3

Verschillende delen:



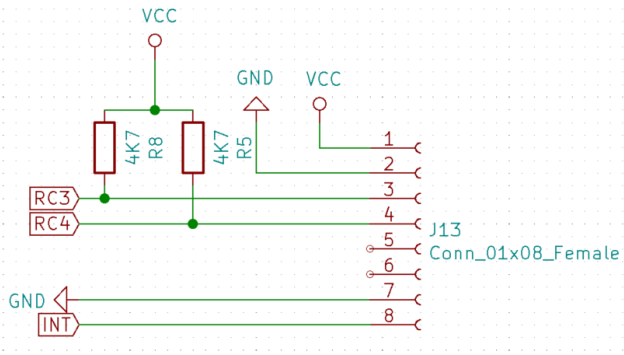
Figuur 4

H-brug met bootstrap



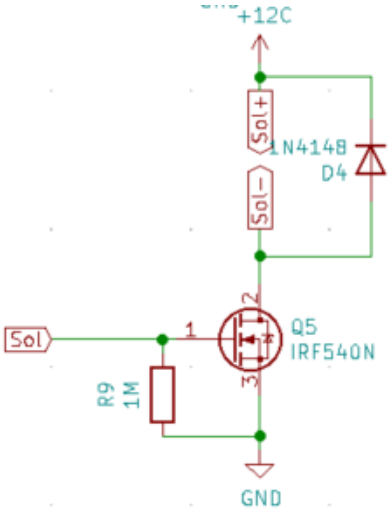
Figuur 5

Drukknoppen

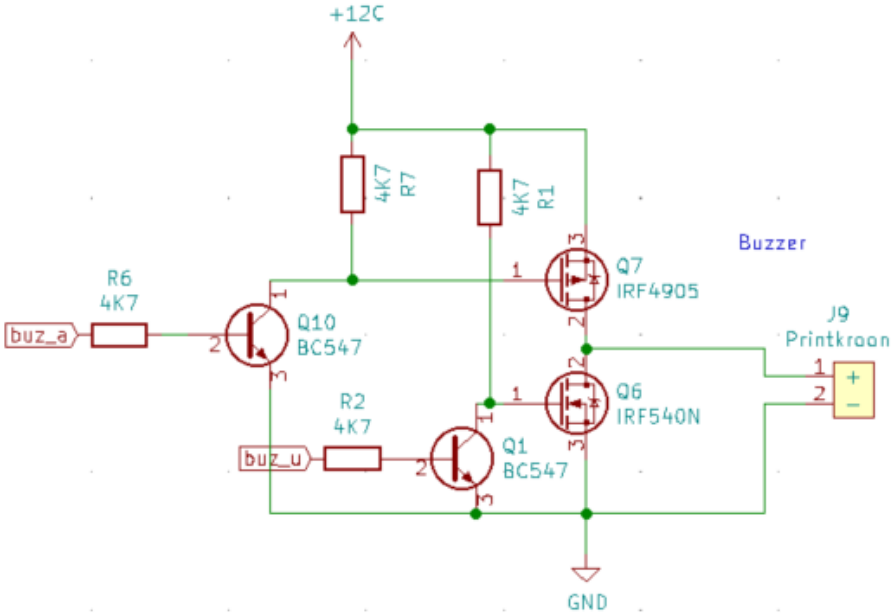


Aansluiting accelerometer

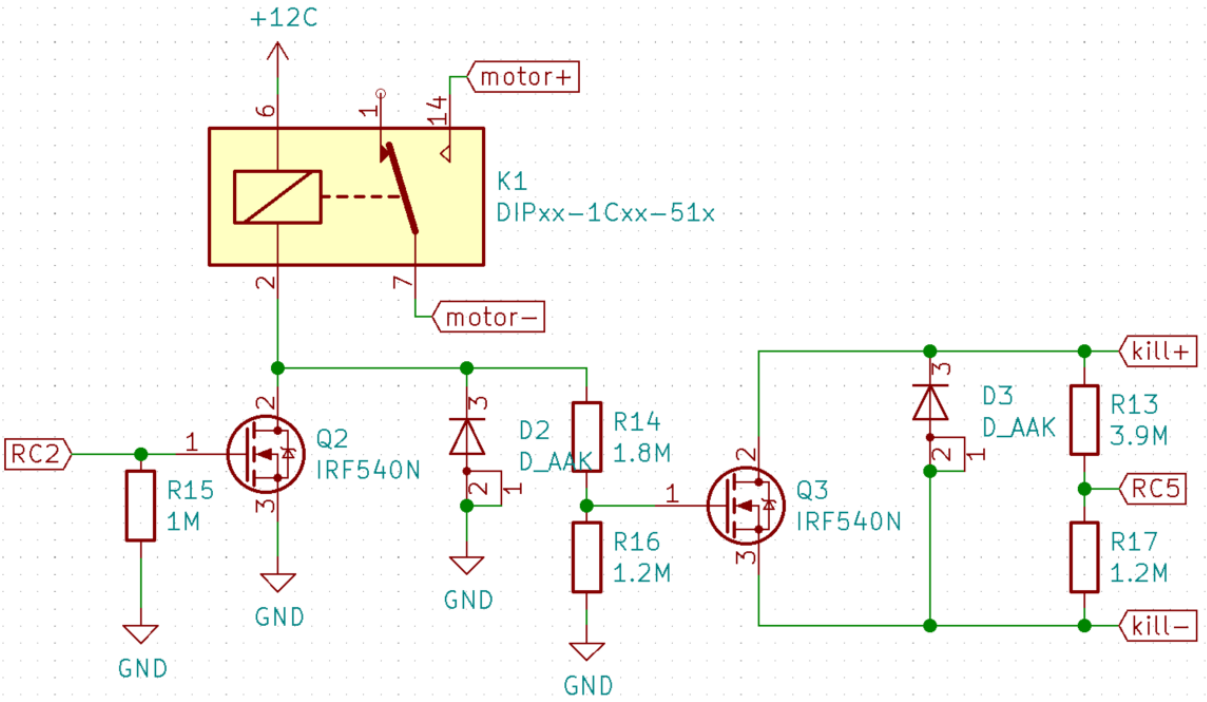
Figuur 6



Figuur 7
Solenoïde aansturing

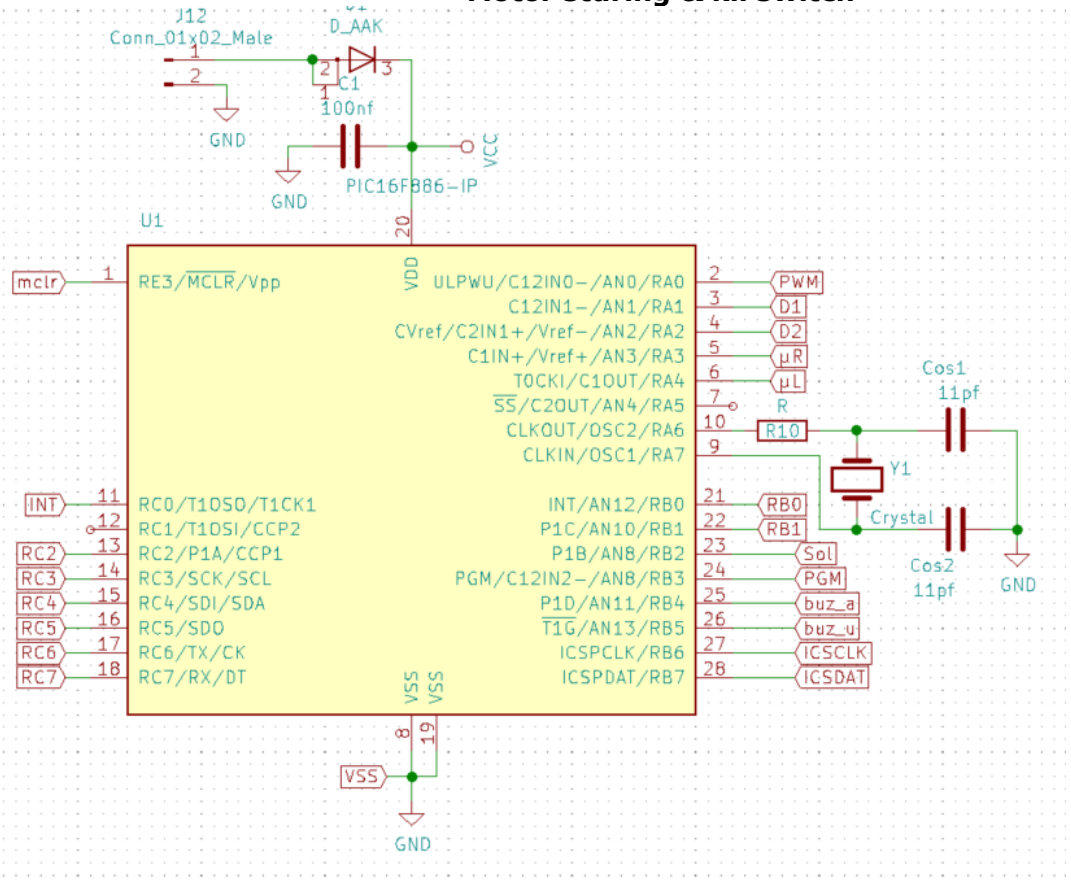


Figuur 8
Buzzer schema



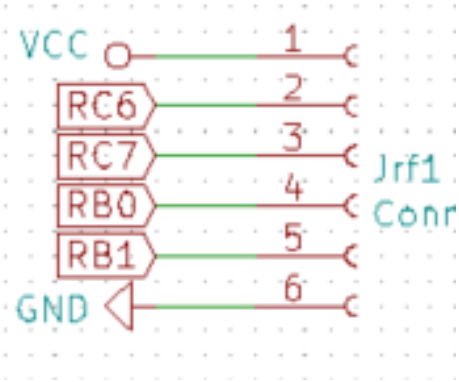
Figuur 9

Motor sturing & kil switch

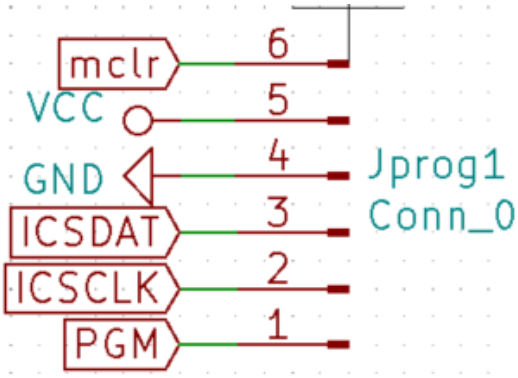


Figuur 10

Voeding en aansluiting pic16f886



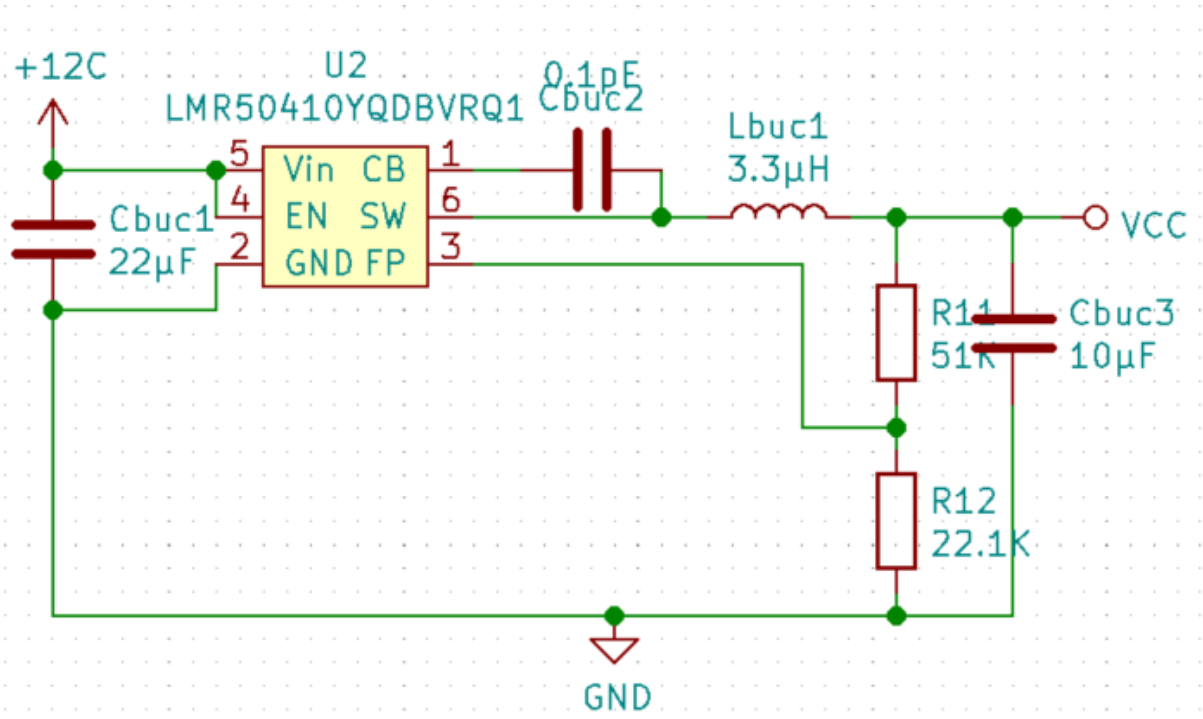
Figuur 11
RF-module connector



Figuur 12
Programmeer connector

6.2 Voeding

Om het pcb bordje te voeden maken we gebruik van een Buck converter die de 12 volt voedingspanning om zet naar een 3,3volt. Deze 12 volt wordt ook gebruikt voor het voeden van de 12 volt DC motor.



Figuur 13
Buck converter schakeling

Dit is het gebruikte schema voor de voeding. Terug te vinden in de datasheet in de bibliografie.

6.3 Microcontroller

Intro

Voor ons project maken wij gebruik van een microchip PIC16F886 als brein voor het alarm en de key-card. De reden dat we met de PIC reeks van microchip zijn gegaan is omdat zowel de controllers als de ontwikkeling omgeving low kost waren en aan de eisen van het project voldeden. Verder hebben wij dan voor de PIC16F886 gekozen doordat wij hier al van microchip zelf een tekstbord konden kopen zodanig dat wij al enkelen componenten konden testen als de printen gemaakt werden.

Keuze microcontroller

De eisen waar onze microcontroller aan moest voldoen, en ook dat ook doen, zijn:

- low kost
- een duidelijke en goed gedocumenteerde ontwikkelomgeving
- Low power zodat de brommer nog kan opstarten na langere termijn stil staat
- mogelijkheid om via seriële communicatie zo als i2c en UART te kunnen communiceren met andere modules.

Ontwikkelomgeving

Voor deze PIC te kunnen programmeren hebben wij gebruikgemaakt van de pickit 3 als programeer dongle. Het programma dat we schreven is gemaakt in MPlab X omdat deze was aangeraden door de verkoper van onze dongle en omdat er veel documentatie over terug te vinden was. Verder maakte wij gebruik van het 28 pin pickit demobord van microchip om stukken software te kunnen testen.

Specificaties PIC

De belangrijkste van deze specificaties zijn dat de PIC een inwendige klok heeft. Deze is instelbaar tussen 31k en 8 M Hz. De werkspanning ligt tussen de 2 en 5 Volt. De werk stroom kan tussen de 11 en 220µA liggen en als laatste dat er met deze IC UART en I2C mogelijk is die we nodig hebben voor onze externe modules.

28/40/44-Pin Flash-Based, 8-Bit CMOS Microcontrollers with nanoWatt Technology

High-Performance RISC CPU:

- Only 35 instructions to learn:
  - All single-cycle instructions except branches
- Operating speed:
  - DC – 20 MHz oscillator/clock input
  - DC – 200 ns instruction cycle
- Interrupt capability
- 8-level deep hardware stack
- Direct, Indirect and Relative Addressing modes

Special Microcontroller Features:

- Precision Internal Oscillator:
  - Factory calibrated to ±1%
  - Software selectable frequency range of 8 MHz to 31 kHz
  - Software tunable
  - Two-Speed Start-up mode
  - Crystal fail detect for critical applications
  - Clock mode switching during operation for power savings
- Power-Saving Sleep mode
- Wide operating voltage range (2.0V-5.5V)
- Industrial and Extended Temperature range
- Power-on Reset (POR)
- Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Brown-out Reset (BOR) with software control option
- Enhanced low-current Watchdog Timer (WDT) with on-chip oscillator (software selectable nominal 268 seconds with full prescaler) with software enable
- Multiplexed Master Clear with pull-up/input pin
- Programmable code protection
- High Endurance Flash/EEPROM cell:
  - 100,000 write Flash endurance
  - 1,000,000 write EEPROM endurance
  - Flash/Data EEPROM retention: > 40 years
- Program memory Read/Write during run time
- In-Circuit Debugger (on board)

Low-Power Features:

- Standby Current:
  - 50 nA @ 2.0V, typical
- Operating Current:
  - 11 µA @ 32 kHz, 2.0V, typical
  - 220 µA @ 4 MHz, 2.0V, typical
- Watchdog Timer Current:
  - 1 µA @ 2.0V, typical

Peripheral Features:

- 24/35 I/O pins with individual direction control:
  - High current source/sink for direct LED drive
  - Interrupt-on-Change pin
  - Individually programmable weak pull-ups
  - Ultra Low-Power Wake-up (ULPWU)
- Analog Comparator module with:
  - Two analog comparators
  - Programmable on-chip voltage reference (CV_{REF}) module (% of V_{DD})
  - Fixed voltage reference (0.6V)
  - Comparator inputs and outputs externally accessible
  - SR Latch mode
  - External Timer1 Gate (count enable)
- A/D Converter:
  - 10-bit resolution and 11/14 channels
- Timer0: 8-bit timer/counter with 8-bit programmable prescaler
- Enhanced Timer1:
  - 16-bit timer/counter with prescaler
  - External Gate Input mode
  - Dedicated low-power 32 kHz oscillator
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Enhanced Capture, Compare, PWM+ module:
  - 16-bit Capture, max. resolution 12.5 ns
  - Compare, max. resolution 200 ns
  - 10-bit PWM with 1, 2 or 4 output channels, programmable "dead time", max. frequency 20 kHz
  - PWM output steering control
- Capture, Compare, PWM module:
  - 16-bit Capture, max. resolution 12.5 ns
  - 16-bit Compare, max. resolution 200 ns
  - 10-bit PWM, max. frequency 20 kHz
- Enhanced USART module:
  - Supports RS-485, RS-232, and LIN 2.0
  - Auto-Baud Detect
  - Auto-Wake-Up on Start bit
- In-Circuit Serial Programming™ (ICSP™) via two pins
- Master Synchronous Serial Port (MSSP) module supporting 3-wire SPI (all 4 modes) and I^2C™ Master and Slave Modes with I^2C address mask

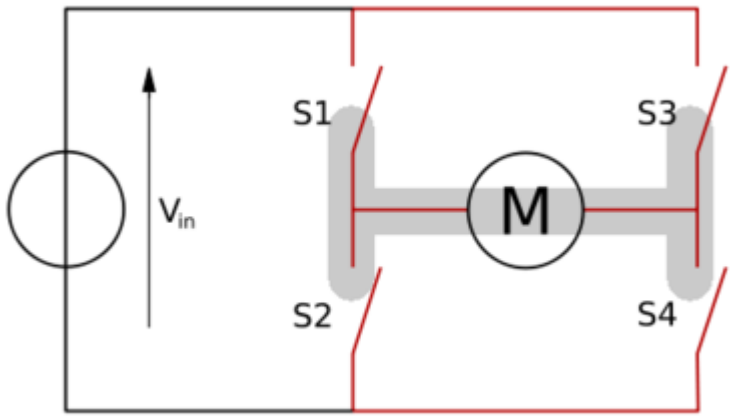
Figuur 14

6.4 H-brug

Het slot wordt bestuurd door een solenoïde die het klemstuk vast zet en een DC-motor die het klemstuk tussen het wiel draait en er terug uit draait.

Om een DC-motor links en rechts te laten draaien moeten we de stroom door die motor respectievelijk in verschillende richtingen laten vloeien. Om dit te realiseren zouden we steeds opnieuw de draden van de motor omdraaien of een relay te gebruiken, maar de meest gebruikte en praktische manier is dan toch de H-burg.

Een H-brug heeft zijn naam te dankten aan de vorm van het principeschema. Dit schema heeft de vorm van de hoofdletter H.



Figuur 15

Principe H-brug

Door schakelaars 1 en 4 samen dicht te laten gaan zal er stroom van links naar rechts door de motor vloeien. Als men deze terug opendoet en schakelaars 2 en 3 dichtdoet zal de motor in de andere richting beginnen te draaien. Men moet wel goed opletten dat hardware- en softwarematig de schakelaars 1 & 2 en 3 & 4 nooit samen kunnen aangaan. Hiermee maakt men namelijk kortsluiting.

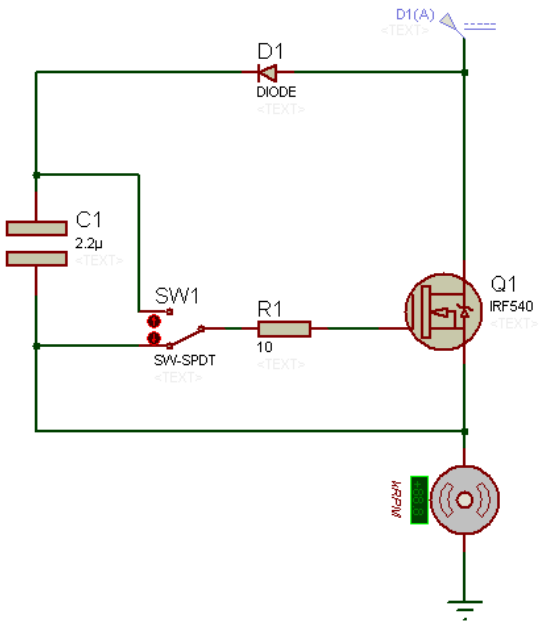
Bootstrapping

Als schakelaars van de H-brug wordt algemeen P-kanaal MOSFET's voor de positieve kant van de motor gekozen en N-kanaal fets voor de negatieve kant. Aangezien de nieuwste generatie P-kanaal fets nog steeds niet zo goed zijn als de eerste generatie N-kanalen hebben we besloten om enkel N-kanaal MOSFET's te gebruiken in mijn H-burg.

$R_{DS(on)} = 44m\Omega$

Figuur 16

Het grote probleem met de N-kanaal MOSFET is dat deze een positieve spanning nodig heeft aan de gate ten opzichte van de source. Omdat de motor op 12 volt werkt moeten we dus de gatespanning hoger zien te krijgen dan 12 volt. Dit doen we door te bootstrappen.



Figuur 17

Bootstrap principe

Door een condensator op te laden aan 12 volt via een diode zal de condensator de voedingsspanning vasthouden. Als we de schakelaar van de MOSFET naar boven zetten zal er een verschil van +12 volt aan de gate van de MOSFET staan in vergelijking met de source. Hierdoor gaat de MOSFET in geleiding en zal de verbruiker, in dit geval een DC-motor, 12

volt opnemen. Omdat er nog steeds een verschil van 12 volt over de condensator staat zal er een verschil van +12volt aan de gate van de MOSFET blijven staan. Nu staat er 24 volt aan de gate en 12 volt aan de source. Omdat de gate en source van elkaar gescheiden zijn zal de condensator niet kunnen ontladen via deze weg. De diode zorgt er dan weer voor dat er geen ontlading is van condensator naar bron. Door het omlaag halen van de schakelaar zal het verschil in spanning tussen gate en source terug 0 volt worden waardoor de MOSFET spert.

MOSFET's

De MOSFET's die ik gebruik in mijn schakeling hebben als typenummer: IRF540N. De belangrijkste eigenschappen staan hieronder opgesomd:

- Rds(on)
- Id
- Pd
- Rth j-a
- Ugs

Rds(on) is de weertand die de MOSFET bezit tijdens het doorlaten. De nieuwste generatie P-MOSFET's hebben nog steeds een slechtere Rds(on) als de vroegere N-MOSFET's. Deze waarde laat ons ook berekenen hoeveel warmte de MOSFET produceert en of men er een heatsink aan de MOSFET moet zetten.

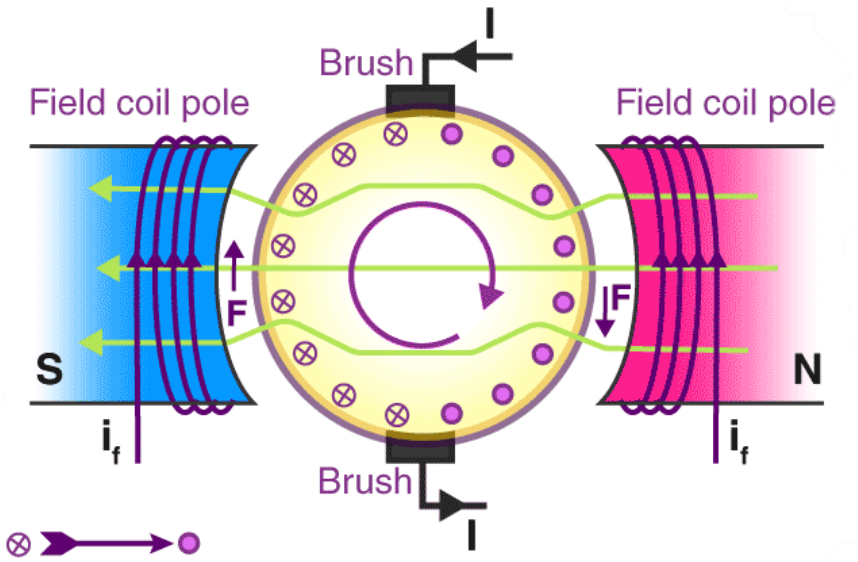
Om te berekenen of je een heatsink nodig hebt of niet moet je eerst weten wat de MOSFET aan warmte opwekt. Om dit te weten moeten we de weerstand Rds(on) vermenigvuldigen met de stroom². Dit geeft ons volgende formule:

- $Pb = Rds.I^2$
- $Pbmax = \frac{maxTJ-TA}{\text{Rth j-a}}$

Omdat het vermogen van de MOSFET de limiet niet overschrijdt moeten we geen heatsink plaatsen.
De spanning die moet staan tussen gate en source is ook belangrijk om te weten. Anders zal de MOSFET nooit doorlaten. Deze spanningswaarde wordt vertegenwoordigd door Ugs.

DC-motor

Een dc-motor is een motor die gelijkstroom gaat omzetten in een draai-beweging. Dit wordt mogelijk gemaakt doordat er een spanning wordt aangesloten op koolstof-borsteltjes. Die geven een contact aan de contact- punten die op de rotor staan. Dit gaat ervoor zorgen dat er een stroom door de spoelen vloeit die op de rotor bevestigd zijn. Deze spoelen gaan dan een magnetisch velt dat tegen gesteld is aan de permanente magneten op de stator waardoor de rotor gaat draaien. Doordat de rotor draait gaan ook de contactpunten op de stator draaien. Hierdoor wordt de spanning omgekeerd waar door het magnetisch veld ook wordt omgedraaid en de rotor blijft draaien.



Figuur 18
Principe DC-motor

6.5 RF-module

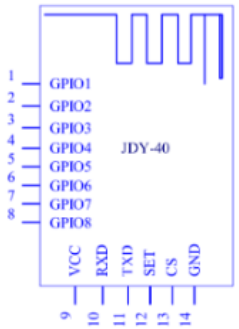
Intro

Wij maken gebruik van de JDY-40-V1.2 als onze RF-module. Deze module gaan we gebruiken voor een draadloze communicatie tussen de key-card en het alarm zelf. Via deze verbinding zullen de key-card en het alarm communiceren dat het alarm op of af moet staan.

Specificaties

De belangrijkste specificaties zij dat de JDY-40-V1.2 een 2.4GHz RF-module is die UART interface communiceert naar de microcontroller. Verder hebben deze modules een bereik tot 120m en verbruiken: 4mA tijdens het verzenden, 24mA tijdens het ontvangen en 4µA in slaap stand en een werk spanning van 2.2 – 3.6V. Als laatste kan men de parameters van deze module instellen via de AT-instructie set.

Pin out



Pin function description

Pin	Pin Names	Function
1	GPIO1	Input/output IO, which can be configured by AT+CLSS
2	GPIO2	Input/output IO, which can be configured by AT+CLSS
3	GPIO3	Input/output IO, which can be configured by AT+CLSS
4	GPIO4	Input/output IO, which can be configured by AT+CLSS
5	GPIO5	Input/output IO, which can be configured by AT+CLSS
6	GPIO6	Input/output IO, which can be configured by AT+CLSS
7	GPIO7	Input/output IO, which can be configured by AT+CLSS
8	GPIO8	Input/output IO, which can be configured by AT+CLSS
9	VCC	Power Supply (2.2V - 3.6V)
10	RXD	Serial port input pin
11	TXD	Serial port output pin
12	SET	AT command switching pin (low level AT instruction, high level transparent transmission)
13	CS	CS chip select pin (low level of awaken, high level of sleep)
14	GND	Power Ground

Figuur 19

AT-intrusie set

AT instruction set

Sequence	instructions	Effect	Default
1	AT+BAUD	Baud rate	9600
2	AT+RFID	Wireless ID	8899
3	AT+DVID	Device ID	1122
4	AT+RFC	Channel（128 Channels）	001
5	AT+POWE	Transmit power	+10db
6	AT+CLSS	Type	A0

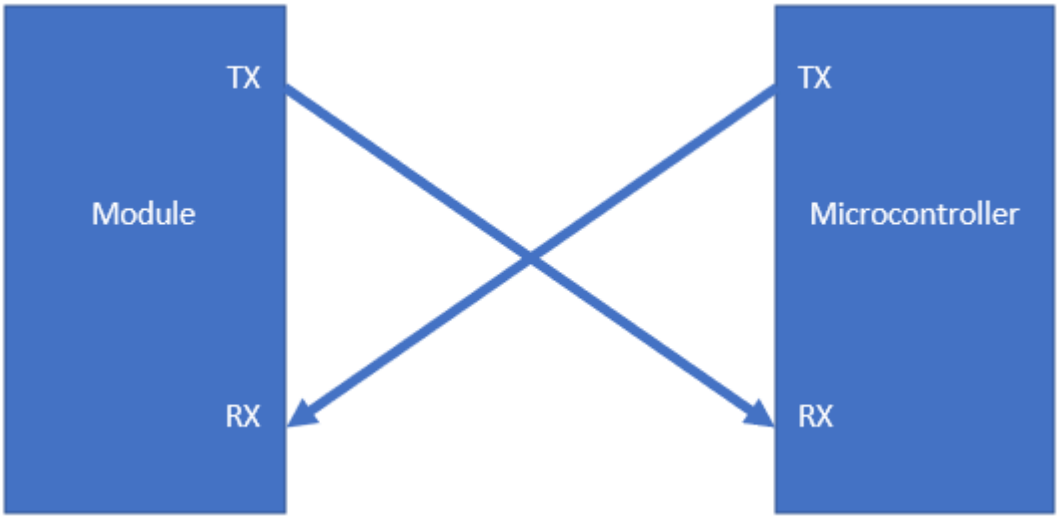
Figuur 20

Werking

Met de CS-pin kan men bepalen dat de module wakker blijft of inslaap moet gaan. Dit door de pin hoog te maken als men de module in sleep wil zetten en laag te maken als men de module waker wil maken voor RF-data te verzenden/ontvangen. De SET pin wort gebruikt voor tegen de modulen te zeggen dat men de UART data moet doorsturen via RF of zich te configureren. Dit wordt gedaan door de SET pin laag te maken voor de module te configureren en hoog te maken als men er RF-communicatie mee wil doen. Via de RX en TX wordt aan de hand van het UART protocol data verstuurt of uitgelezen naar of van de microcontroller.

6.5.1 UART

UART is een asynkroon protocol dat gebruik maakt van 2 pinnen (RX en TX) voor een seriële data overdracht. De RX en TX worden omgewisseld bij de hardware verbinding zoals in onderstaande tekening.



Figuur 21

Protocol

Start bit	5-9 Data bits	0 – 1 Controle bit	Stop bit

Werking

De UART transmissie lijnen worden hooggehouden wanneer er geen data wordt gestuurd. Door deze laag te maken met de start bit laat de zender van de data weten tegen de ontvanger dat hij data gaat sturen hierna zal de zender 5 tot 9 bits data sturen. Hoeveel data bits dat deze stuurt hangt af van de onderlinge afspraken maar in onze toepassing wordt er gebruik gemaakt van 8 bits. Na de data bits wordt er in sommigen gevallen nog een controle bit gestuurd maar dat wordt in onze communicatie niet gebruikt. Als laatste wordt er nog een stop bit gestuurd die de transmissie lijn terug hoog zet om de receiver te laten weten dat de communicatie gedaan is.

UART configuratie op de PIC16F886

Als eerste gaan wij tegen de PIC zeggen dat op de USART poort (pin C6 en C7) op asynchroon zetten. Dit doen we door de SYNC bit op 0 te zetten. Hierna gaan we de seriële poort enablen door de SPEN bit hoog te maken waarna we de RX(C7) en TX(C6) pinnen gaan configureren. Door TRISC6 bit hoog te maken en TRISC7 bit laag te maken zetten wij de TX

pin als een output en de RX pin als een input. Als dit gebeurd is zijn wij klaar op de baud rate op 96000 in te zetten. Dit doen we door het juist instellen van de BRG16 bit, BRGH bit en het register. De instellingen hiervoor vinden wij terug in tabel 12-5 van de datasheet bij 1Mhz omdat dit onze klok frequentie is. Dan gaan we als laatste er voor zorgen dat de USART poort blijft ontvangen en kan blijven sturen. Dit gebeurt door de CREN bit en bit hoog te maken.

UART zenden en ontvangen op de PIC16F886

Voor het versturen van data moeten we eerst kijken dat de PIC momenteel bezig is met het versturen van data dit doen we door te kijken dat de TRMT bit hoog is wanneer deze hoog is kan men in het TXREG register data zetten die men wil versturen. De PIC zal deze dan automatisch op de juiste manier verzenden. Voor het ontvangen van data moet men eerst kijken dat er wel data ontvangen is dit doet men door naar de RCIF bit te kijken of deze hoog is. Als dit het geval is kan men dan de ontvangen data lezen uit het RCREG register.

TABLE 12-5: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

BAUD RATE	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1											
	Fosc = 20.000 MHz			Fosc = 18.432 MHz			Fosc = 11.0592 MHz			Fosc = 8.000 MHz		
	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)
300	300.0	0.00	16665	300.0	0.00	15359	300.0	0.00	9215	300.0	0.00	6666
1200	1200	-0.01	4166	1200	0.00	3839	1200	0.00	2303	1200	-0.02	1666
2400	2400	0.02	2082	2400	0.00	1919	2400	0.00	1151	2401	0.04	832
9600	9597	-0.03	520	9600	0.00	479	9600	0.00	287	9615	0.16	207
10417	10417	0.00	479	10425	0.08	441	10433	0.16	264	10417	0	191
19.2k	19.23k	0.16	259	19.20k	0.00	239	19.20k	0.00	143	19.23k	0.16	103
57.6k	57.47k	-0.22	86	57.60k	0.00	79	57.60k	0.00	47	57.14k	-0.79	34
115.2k	116.3k	0.94	42	115.2k	0.00	39	115.2k	0.00	23	117.6k	2.12	16

BAUD RATE	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1											
	Fosc = 4.000 MHz			Fosc = 3.6864 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)
300	300.0	0.01	3332	300.0	0.00	3071	299.9	-0.02	1666	300.1	0.04	832
1200	1200	0.04	832	1200	0.00	767	1199	-0.08	416	1202	0.16	207
2400	2398	0.08	416	2400	0.00	383	2404	0.16	207	2404	0.16	103
9600	9615	0.16	103	9600	0.00	95	9615	0.16	51	9615	0.16	25
10417	10417	0.00	95	10473	0.53	87	10417	0.00	47	10417	0.00	23
19.2k	19.23k	0.16	51	19.20k	0.00	47	19.23k	0.16	25	19.23k	0.16	12
57.6k	58.82k	2.12	16	57.60k	0.00	15	55.56k	-3.55	8	—	—	—
115.2k	111.1k	-3.55	8	115.2k	0.00	7	—	—	—	—	—	—

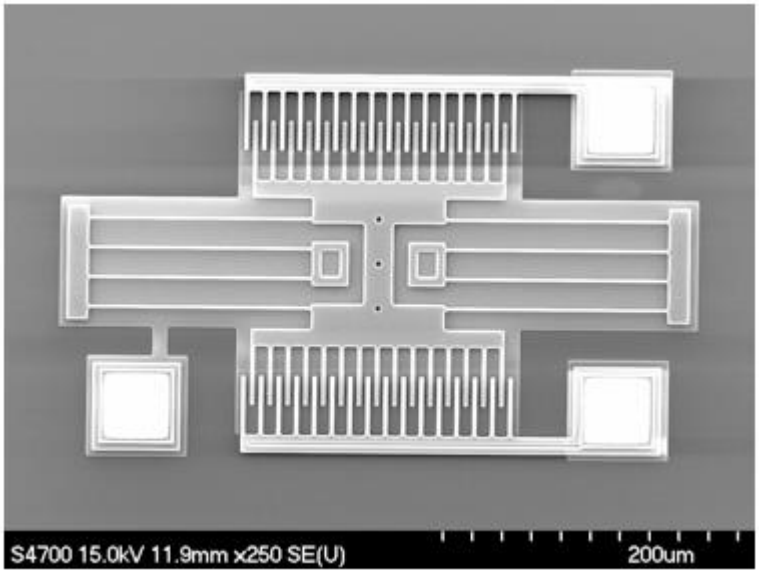
Figuur 22

6.6 MPU6050

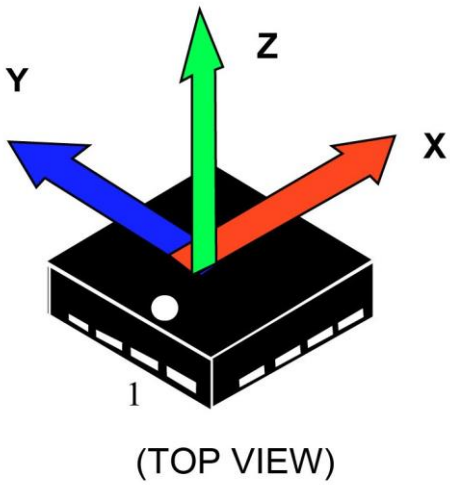
De mpu6050 is een erg complexe component waarvan de datasheet groter is dan dit verslag. Dit komt omdat de MPU6050 3 interne en 5 externe censoren ter beschikking kan hebben. Het ondersteunt ook I2C communicatie & SPI communicatie en heeft een auxiliary I2C poort voor 5 andere slave devices in te lezen zodat communicatie met de master vlotter verloopt. Het is meer een hub voor censoren dan dat het zelf een censor is. Maar door de vereiste van dit project gebruiken we deze hub als Accelerometer in low power mode.

Een accelerometer is een censor die versnelling detecteert. Dit gebeurd door MEMS dat staat voor: Micro-electromechanical systems. Dit zijn zoals

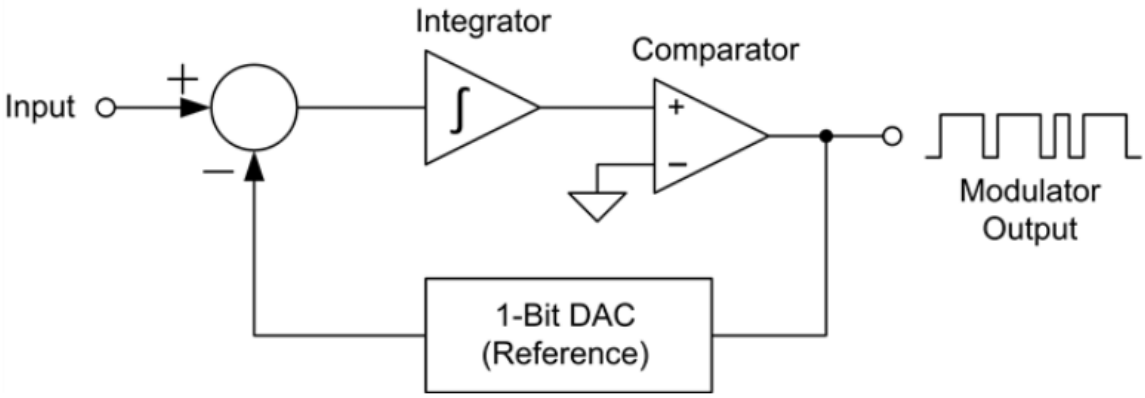
de naam verteld Microscopisch kleine mechanische censoren die bewegen wanneer ze een versnelling ervaren. Deze verplaatsing wordt door een capacitieve censor ingelezen en vervolgens door een sigma-delta ADC verwerkt naar een digitale waarde.



Figuur 24
Interne mems



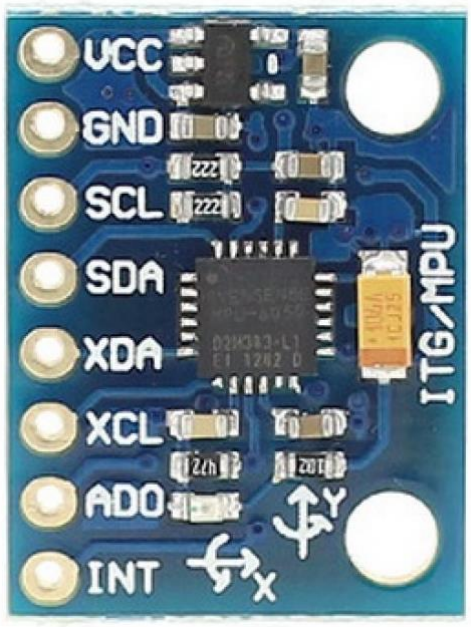
Accelerometer
Figuur 23



Figuur 25
Sigma-delta ADC

De MPU-6050 heeft nog wat extra mogelijkheden buiten de I2C bus. Zo heeft hij een extra "aux" bus die hij gebruikt om zelf master over te spelen. Zo kan je via de interne registers 5 ander slave adressen laten besturen door de MPU en op hun beurt een extra MPU aansluiten. Deze laatste kan gebeuren door AN0 aan VCC te leggen. Deze zorgt ervoor dat het adres van de MPU met 1 bit verhoogd word. Ook heeft het device een

INT pin. Deze kan geconfigureerd worden zodanig dat hij een interrupt signaal genereerd naar de master. Dit kan een signaal zijn dat data klaar staat voor gelezen te worden, de interne buffer overflowt of dat een externe slave data heeft verstuurt op de tweede bus.



Figuur 26
MPU-6050 module

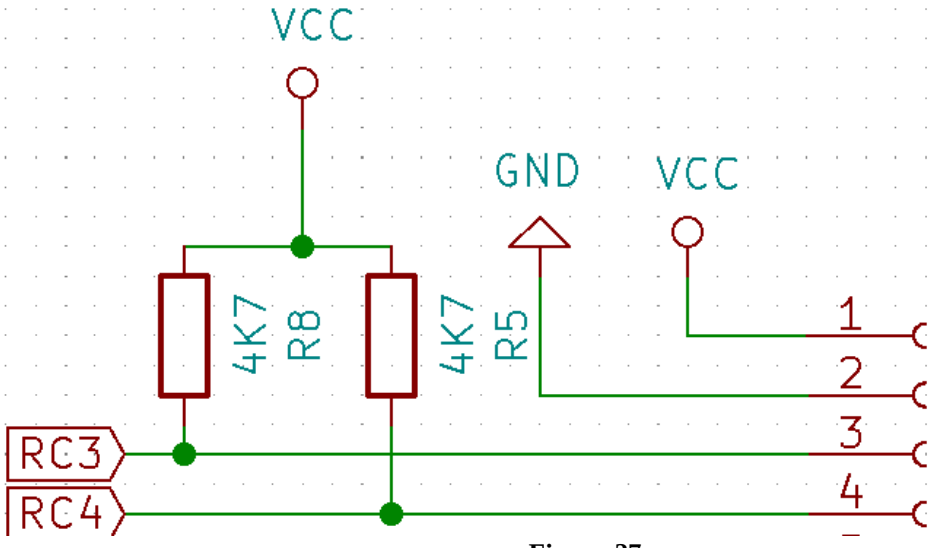
Wat de interne censoren betreft heeft de MPU er 3. Een accelerometer, een gyroscoop en een temperatuur censor. Deze twee eerste worden door MEMS en sigma-delta ADC's verwerkte tot een 2 byte signed waarde. De gyroscoop hadden we mogelijks ook kunnen gebruiken voor beweging te detecteren omdat een gyroscoop de verplaatsing in graden meet. Maar omdat dit veel rekenintensiever is en er al instructies voor low power acceleratie te meten in de datasheet stonden leek het ons beter deze te gebruiken.

6.6.1 I2C

I2C is een vorm van seriële communicatie die werkt aan de hand van 2 draden namelijk een klok-lijn scl en een data-lijn sda. Voor I2C te laten werken wordt er gebruikgemaakt van een master en een slave. Het is zelfs zo dat men gebruik kan maken van twee of meerdere masters op één bus.

De microcontroller ondersteunt I2C communicatie doormiddel van interne registers en configuraties. Deze kunnen start en stop condities genereren door registers te schrijven en data kan verzonden en ontvangen worden doormiddel van een intern register genaamd SSPBUF. SSP staat voor Synchronous Serial Port.

Deze poort heeft ook ingebouwde pull-up weerstanden en maakt zijn aansluitingen dus open-drain. Dit is vereist voor de communicatie door I2C maar omdat deze interne weerstanden niet voor stabiele communicatie kunnen zorgen hebben we externe pull-up weerstanden aangesloten. Te zien op fig.27



Figuur 27

I2C bus

Communicatie

Voor een goede communicatie mogelijk te maken zijn er een paar regels gemaakt. Zo wordt er gebruik gemaakt van een master-slave systeem. Hierbij is de master de leider van de bus en bepaalt over de snelheid waarop er gecommuniceerd wordt. Masters kunnen communiceren met andere masters en slaves. Slaves zijn devices die geen communicatie kunnen initialiseren en kunnen dus enkel antwoorden op berichten van masters. Voor een visualisatie zie fig.28

Een gebruikelijke communicatie gebeurd als volgt:

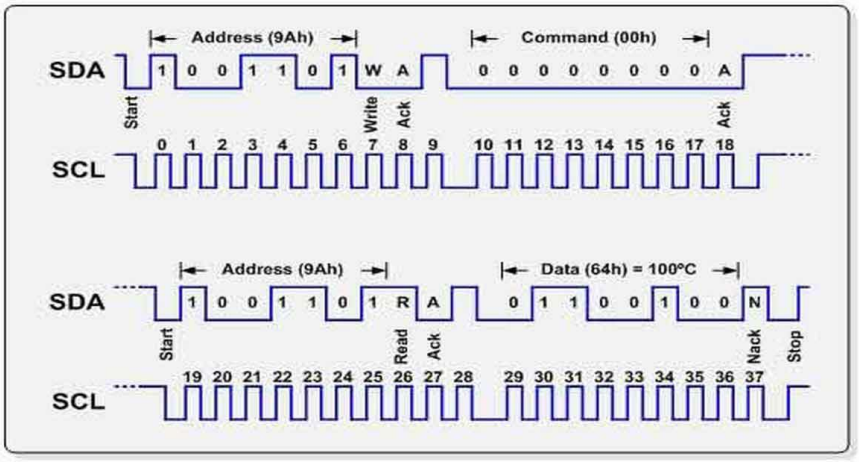
1) De master voert een start conditie uit op de I2C bus die aan alle andere apparaten duidelijk maakt dat de bus in gebruik gaat zijn en hen aangeeft te luisteren naar het aankomende adres. Deze startconditie gebeurd door een wisseling van data, hoog naar laag, terwijl het kloksignaal hoog is. Dit mag dus nooit gebeuren tijdens het sturen van data want anders interpreteert de slave dit als een herhaalde start conditie.

2) Het adres van de gewenste slave of master wordt doorgestuurd door de master die de bus bestuurd. Dit is gebruikelijk een 7 bit adres waarvan de 8ste bit aangeeft of er naar dit adres geschreven zal worden of uitgelezen.

3) De master zal hierna een extra clockpuls genereren op de bus om het aangesproken device een kans te geven een bevestiging te sturen. Als de slave een bevestiging stuurt weet de master dat de slave correct aangesproken is.

4) Nu kan de master, afhankelijk van of er geschreven of gelezen wordt. Data sturen/ontvangen via de bus. Ook hier wordt steeds een bevestiging op gevraagd aan de slave*.

5) Als de master genoeg data heeft ontvangen en de communicatie wilt beëindigen stuurt hij een stop conditie. Dit is een wisseling van de data, laag naar hoog, terwijl het kloksignaal hoog is.

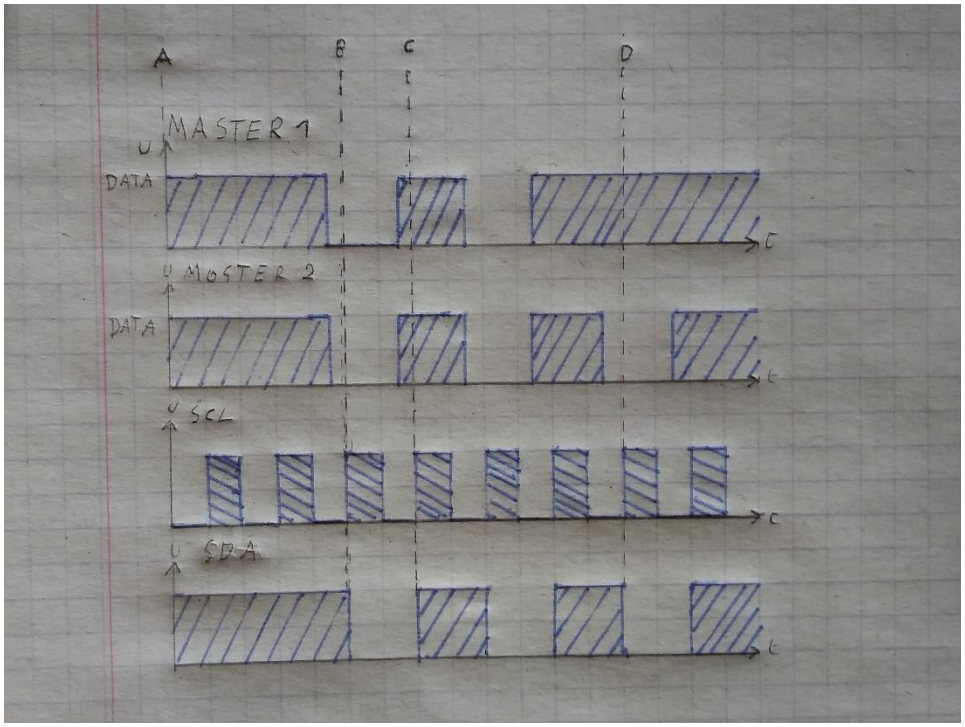


Figuur 28
I2C communicatie

*Bij het ontvangen van data stuurt de master een bevestiging.

Master-master

Als men meerdere masters gebruikt kan het gebeuren dat er twee of meer masters op hetzelfde moment data willen sturen. Dit zorgt ervoor dat er een master-masterconflict ontstaat. Het doorsturen van de juiste data gebeurt op volgende wijze: De SDA (datalijn) is normaal hoog (punt “A” tot “B” in onderstaande tekening). Als er meerdere masters een zelfde signaal sturen gebeurt er niks en zullen ze blijven sturen (punt “B” tot “D” in onderstaande tekening). Wanneer er een verschillend signaal gestuurd wordt dan gaat de master die normaal hoog wordt detecteren dat er niet gereageerd wordt op zijn signaal (na punt “D”). Daarna gaat dezen stoppen met sturen en gaat de andere master voort.



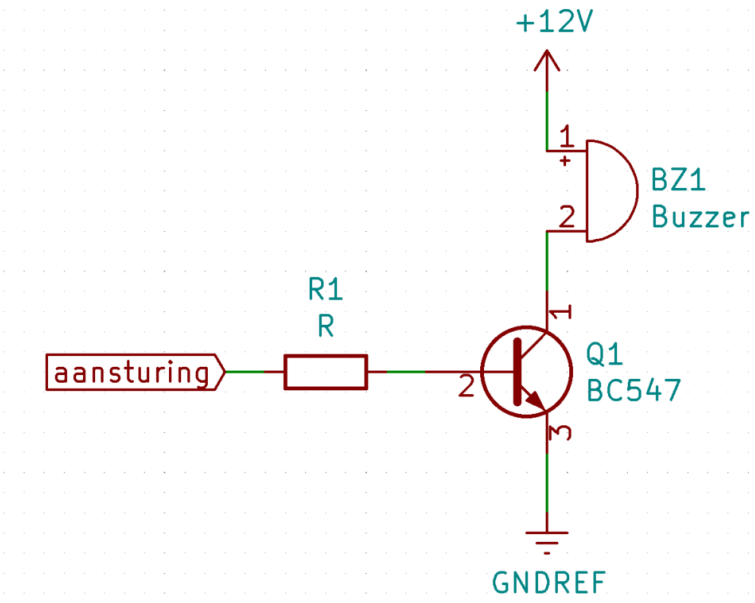
Figuur 29
Multi master collision

Merk op dat de data (sda) veranderd tijdens een flank van de klok. Dit gebeurd in werkelijkheid niet waarbij data enkel veranderd als de klok laag is. (Foutje in de tekening.)

6.7 Buzzer

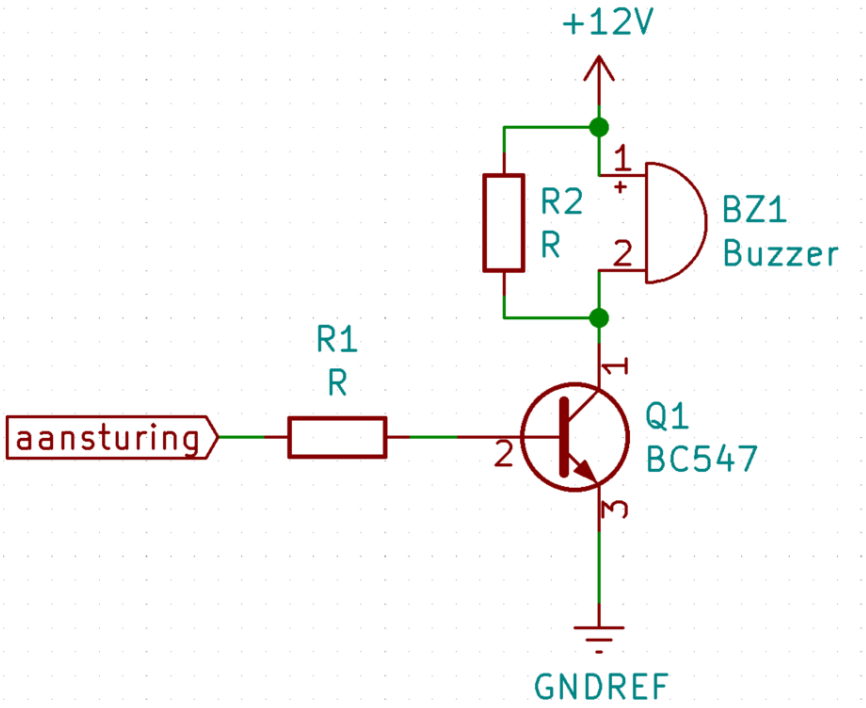
Om dieven bang te maken en de eigenaar te waarschuwen dat de brommer gestolen wordt maken we gebruik van een buzzer. Deze zal een hoge alarmtoon geven wanneer hij merkt dat de key-card niet in de buurt is, het alarm op staat en tegelijk er beweging plaatsvind op de brommer.

Voordat men de schakeling kan begrijpen moet men eerst weten wat een Buzzer precies is. Namelijk een spoel die rond een kern gewikkeld is. Deze zal de kern naar zich toe trekken als er spanning aan de spoel gelegd wordt en als men dat vaak achter elkaar doet genereerd deze beweging een hoorbare toon. Omdat spoelen energie opslaan tijdens het doorlaten van stroom kan men niet zomaar volgende schakeling gebruiken.



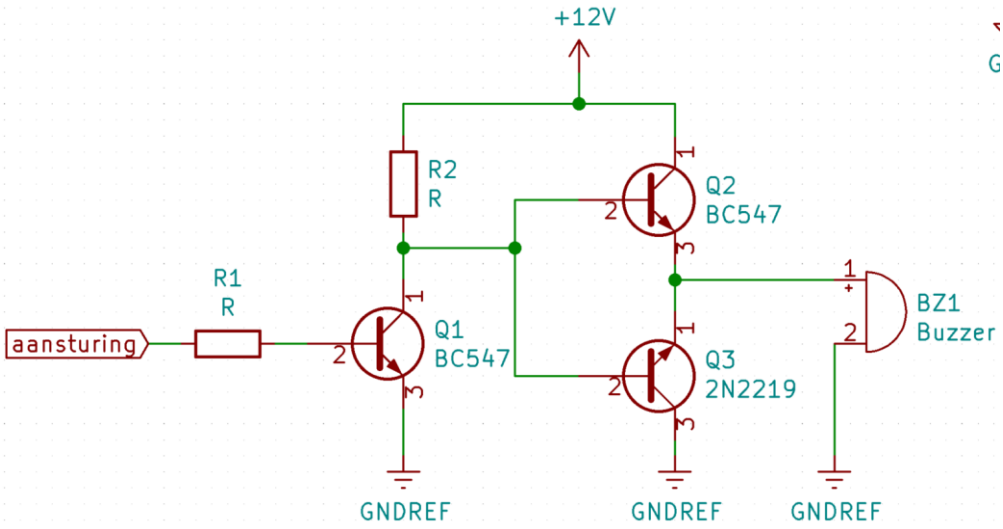
Figuur 30

Het is namelijk zo dat de spoel zichzelf zou moeten ontladen. Dit kan gedaan worden met een weerstand of diode parallel over de buzzer te zetten.



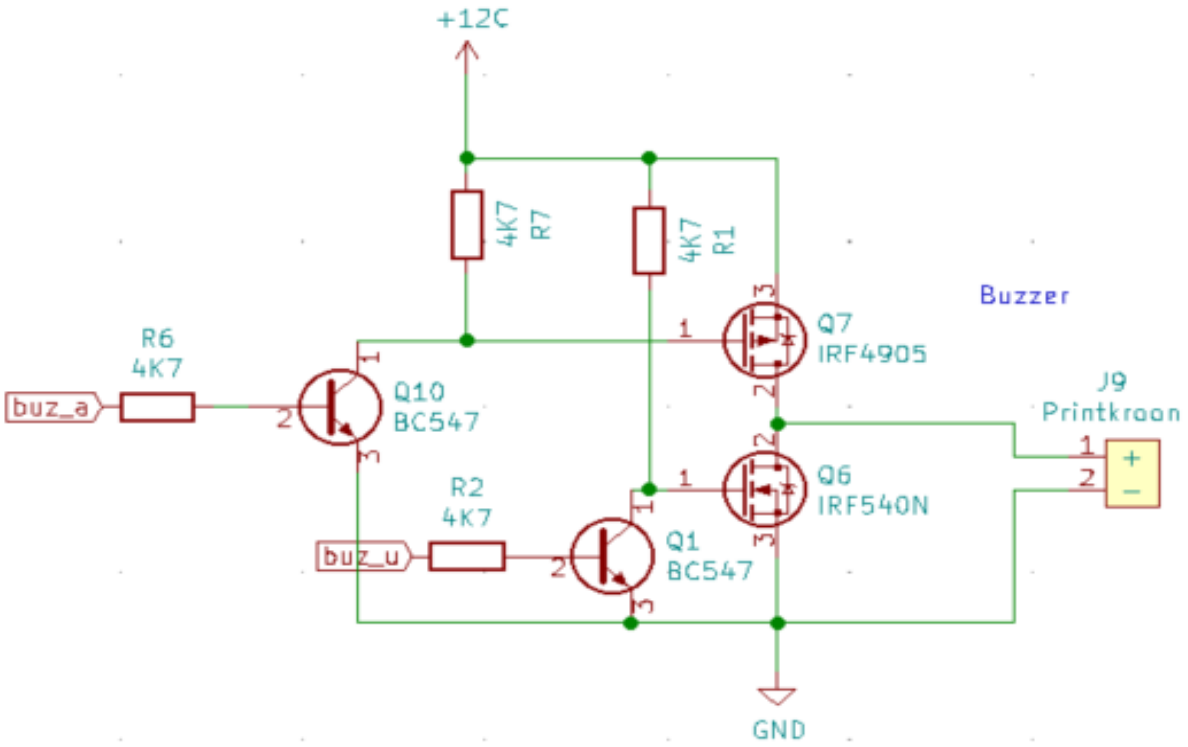
Figuur 31

Het nadeel hiervan is dat de weerstand tijdens het bedrijven van de buzzer ook vermogen opneemt wat een overbodige verbruiker is.



Figuur 32

Het weertand probleem kan opgelost worden door hem te vervangen door 2 transistoren die ervoor zorgen dat de weerstand heel hoog is tijdens het HIGH gedeelte en de buzzer kan ontladen direct daarna. Wat we winnen aan dit sircuit moeten we dan wel terug opofferen door een vermindering van spanning aan de zoemer van ongeveel 1,2 volt. Dit brengt ons op de laatste en gebruikte schakeling.



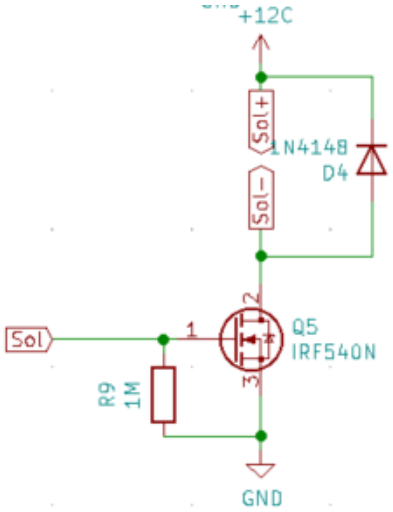
Figuur 33

Hierbij maken we een push-pull trap voor de zoemer die we zelf kunnen besturen zodat beide MOSFET's niet tegelijk gaan doorgeven. Dit zorgt ervoor dat er niet steeds een kortsluiting plaatsvindt tijdens het schakelen.

6.8 Solenoïde

De solenoïde heeft maar 1 enkel doel. Het haakstuk vast zetten voor en na het in of uitschakelen van het slot. Dit is als extra veiligheid dat deze niet tijdens het rijden tussen het wiel terecht komt en tegelijk zodat dieven een moeilijkere tijd hebben met het uittrekken van de haak.

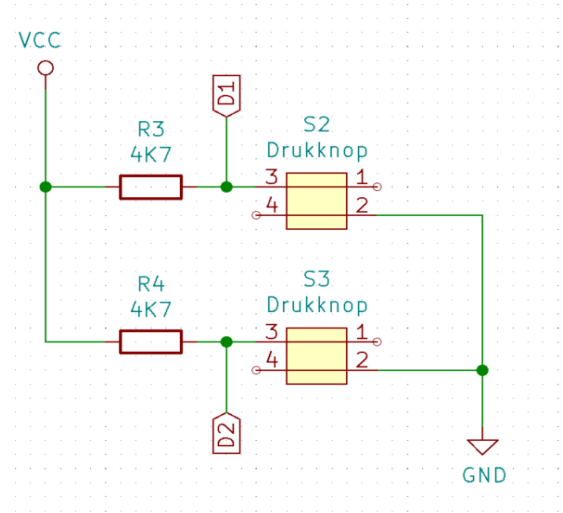
De solenoïde is net zoals de buzzer een inductieve verbruiker. Daarom moeten we ook hier zorgen dat deze ontladen kan worden in de tijd dat we hem niet gebruiken. Hiervoor is gezorgd door een antiparallele diode over de solenoïde te zetten.



Figuur 34

6.9 Drukknoppen

De drukknoppen hebben als functie het aangeven van de toestand van het slot. Als de drukknop een signaal aangeeft dat deze ingedrukt is weten we dat het slot tegen de drukknop zit en vice versa. Een alternatief voor deze primitieve contactdetector was dan ook een infrarood detector. Maar deze was buiten ons budget gerekend.

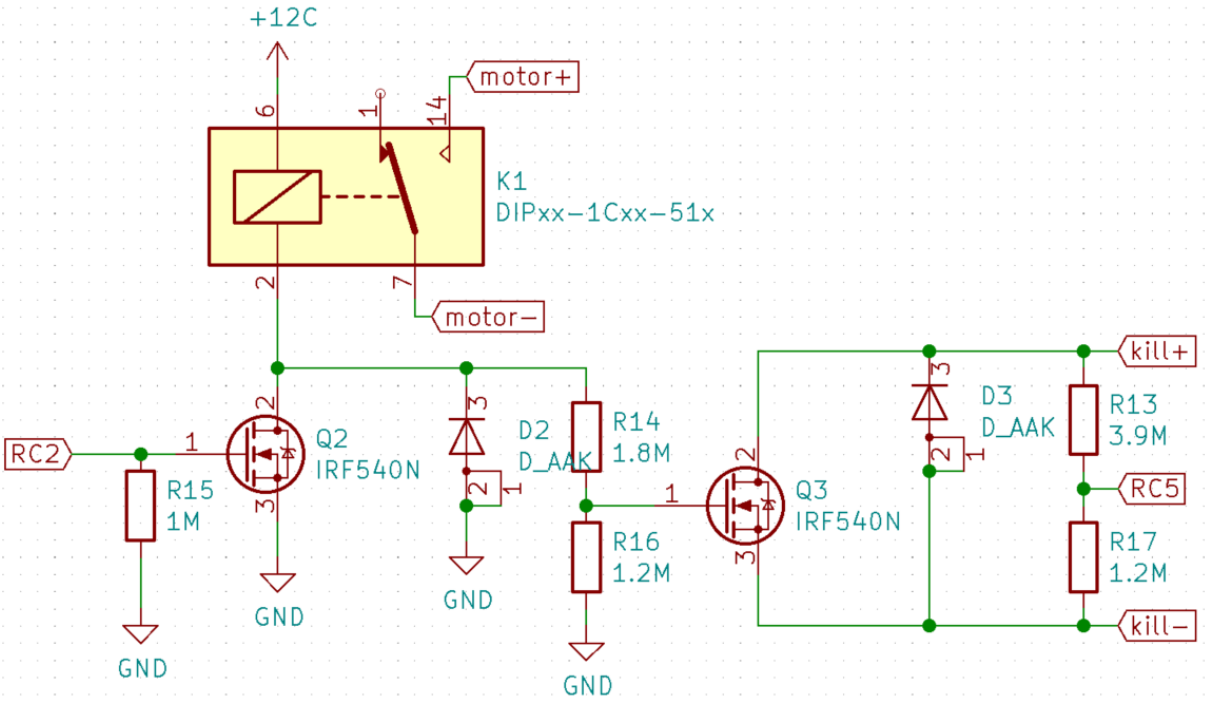


Figuur 35

Hierbij gebruiken we twee drukknoppen die aangesloten zijn aan pull-up weerstanden.

6.10 Motor en kill switch

Om de motor stil te leggen wordt een relay aangestuurd dat de voeding naar de motor toe afschakelt. Ook staat er op het stuur van de brommer een kil switch die als beveiliging dient tijdens het rijden.



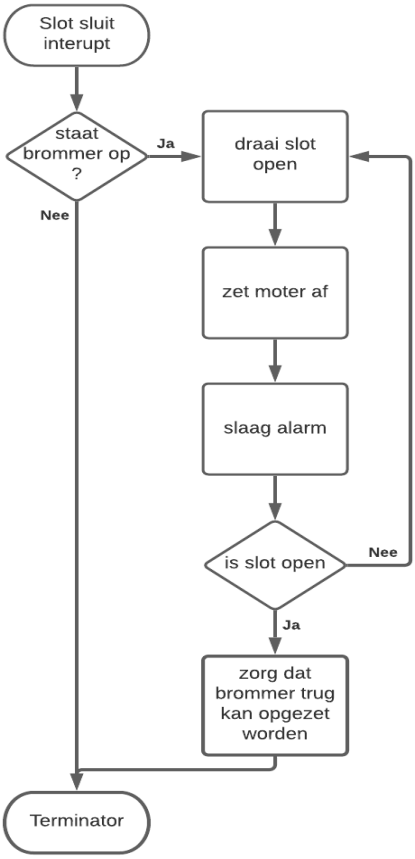
Figuur 36

7 Software

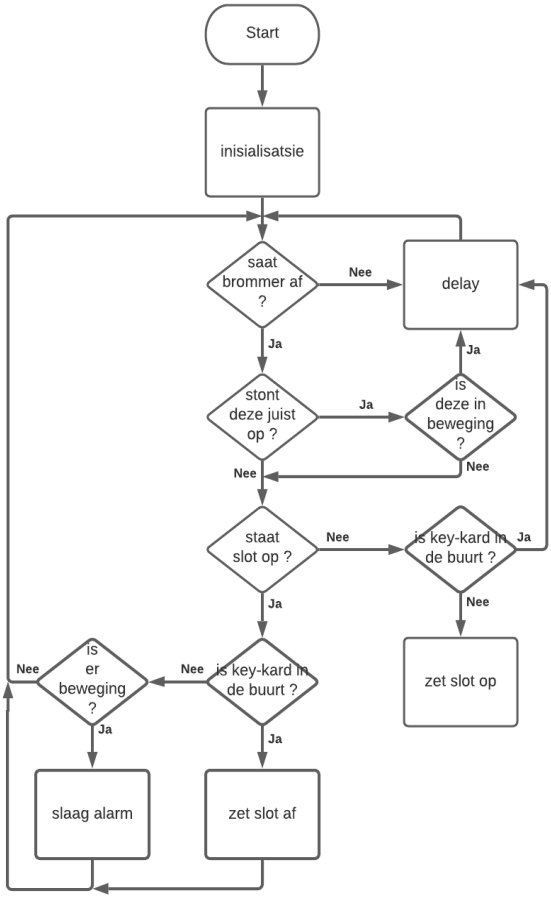
7.1 Flowcharts

Omdat de code tussen de tijd van schrijven van dit verslag en werkelijk presenteren van het project nog sterk kan veranderen hebben we besloten om enkel de flowcharts in dit verslag te zetten.
Mocht u interesse hebben in een kijkje te nemen naar deze code kunt u altijd terecht op onze GITHUB repository via: deze link

Interrupt voor slot

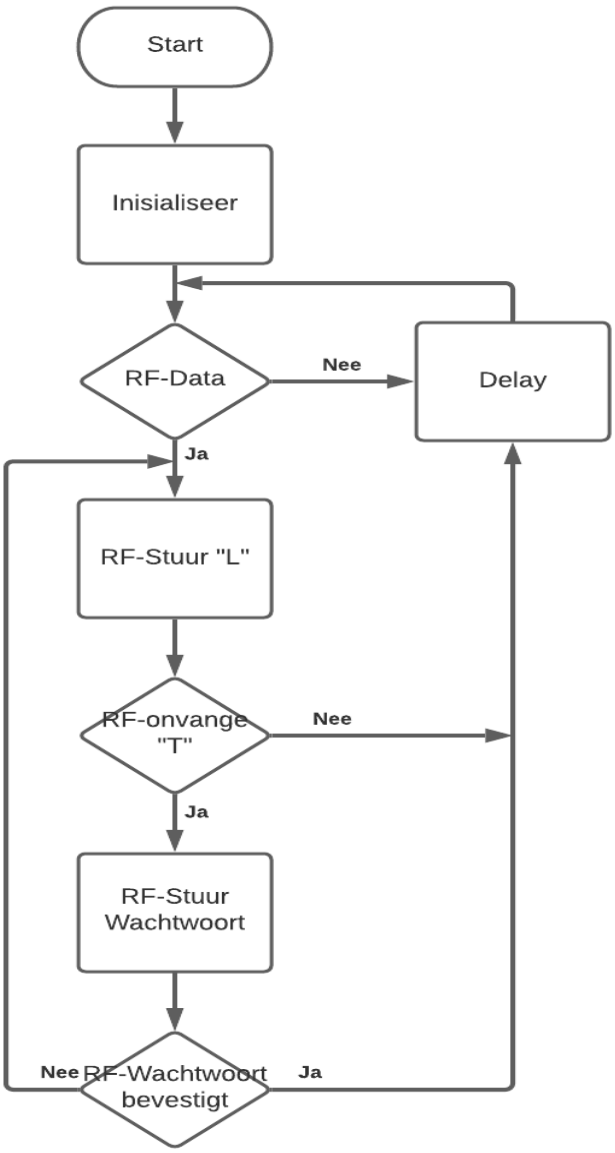


main code moederboard



Figuur 37
Figuur 38

Main code key-card



Figuur 39

8.1 Besluit Lauren

Als ik terugblik op de manier waarop we dit project gemaakt hebben ben ik relatief tevreden. We waren op tijd begonnen en hadden ervoor gezorgd dat als we moesten wachten op bepaalde componenten, we altijd wel een ander deel van ons project hadden waar wij aan konden werken. Het enige wat ik zou willen aanpassen aan onze werkwijze is dat dat we van tevoren een betere planning maken van wanneer welk component af moet zijn.

Wanneer ik dan over onze samenwerking terugkijk ben ik echter heel tevreden. We hebben elk ongeveer evenveel aan dit project gedaan en vulde in mijn ogen elkaars zwaktes op. Verder was het handig dat wij dicht bij elkaar wonen zodat we op een corona veilige manier ook samen aan het project konden werken in het weekeinde.

Dan als ik als laatste terugblik op de resultaten die we geboekt hebben ben ik echter heel tevreden. Voor de rest ben ik persoonlijk zelf van plan om dit op mijn brommer te monteren na dat ik er een stevigere behuizing heb rond gemaakt. Verder denk ik dat dit slot ideaal geschikt zou zijn mits een paar esthetische aanpassingen zo als steviger materiaal en een mooiere behuizing voor de open markt.

8.2 Besluit Toon

Nu het einde van het project in zicht is duiken er toch altijd de gebruikelijke dingen op zoals de tijdsdruk om het verslag af te schrijven en of alle code wel in orde is of hier en daar weer een hardware fout die nergens op slaat. Maar ondanks deze tegenslagen die we dit weekend er nog uit gaan werken ben ik zeer tevreden over wat we dit semester hebben kunnen bereiken.

De samenwerking met Lauren verliep vlot zoals ik verwachte omdat we goed op elkaar zijn ingespeeld. Ik heb geprobeerd hem waar ik kon te helpen en hij mij terwijl we gerust ons eigen ding konden doen en voorbereiden. Het heeft er dan ook voor gezorgd dat we veel tijd hadden om op ons eigen naar de gekozen problemen toe te werken en daarvoor oplossingen te vinden. Daarbij het toepassen van de leerstof van dit jaar uiteraard niet te vergeten. Zeker het verwerken van grote boeken datasheets was voor me echt een ervaring die ik vorig jaar zeker niet zo durfde trotseren.

Ik zou ook graag de leerkrachten willen bedanken voor hun tijd en moeite die ze in ons hebben gestoken voor ons steeds verder te pushen naar oplossingen waar we soms zelf niet op zouden komen.

Samengevat ben ik heel bij met hoe het project is uitgedraaid en wat we er samen van hebben kunnen bereiken. Ik had me geen betere collega kunnen bedenken en hoop dat ik op deze manier nog veel projecten kan verwezenlijken.

Bibliografie

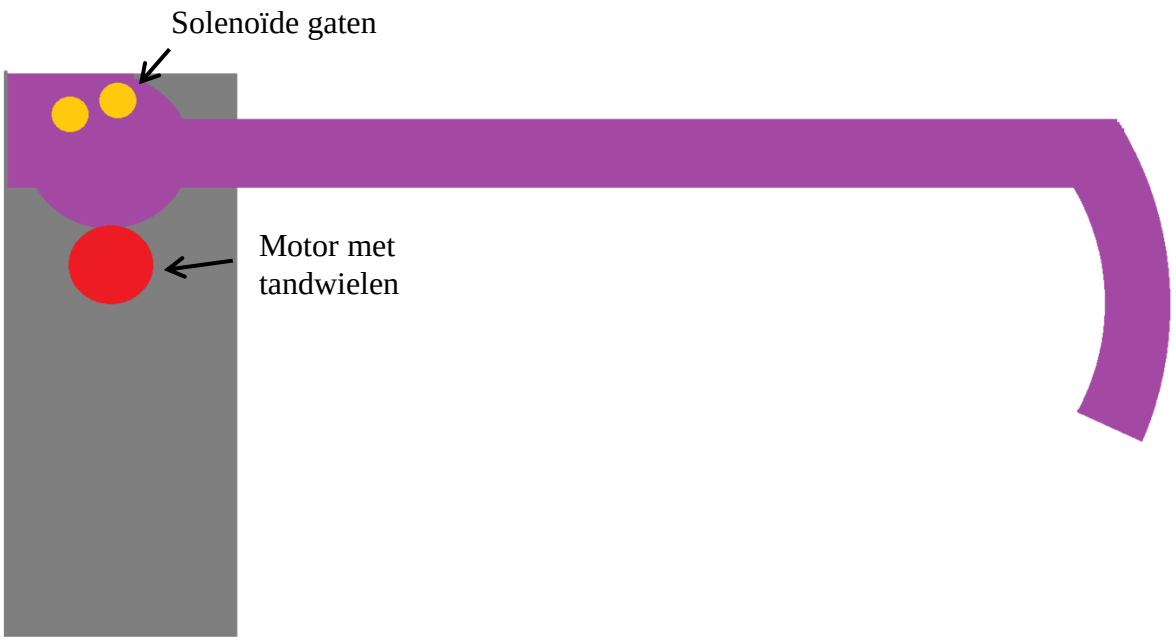
1 Internet links

- **LMR50410 Buck converter**
- **PIC16F886**
- **RF-module**
- **IRF540N(N-chan MOSFET)**
- **IRF4905(P-chan MOSFET)**
- **BC547(NPN transistor)**
- **MPU-6050 product specification**
- **MPU-6050 register map**

Bijlagen

B-1 Wiel lock

In onderstaande figuur kan men het locking mechanisme zien. Als u op deze tekening kijkt ziet u in het paarse-haak deze kan draaien op de base van het slot die in het grijs wordt weergegeven zodat deze tussen de spaken van de brommer kont. Dit zorgt ervoor dat men de brommer niet meer zal kunnen verplaatsen. Voor het in plaats te houden van deze haak zijn hier 2 gaten in voorzien in het geel. In deze gaten komt namelijk de stang van de solenoïde. Wat ervoor zorgt dat het slot in de juiste positie blijft staan. Het in de juiste positie zetten van de haak wort gedaan door een motor die is aangesloten op het wiel dat in het root is ingekleurd.



Figuur 40