SECUNDAIR ONDERWIJS
ZELZATE • WACHTEBEKE • MARIAKERKE
Sint-Laurens

Geïntegreerde Proef

TSO Elektriciteit elektronica

2018 – 2019

Technisch directeur: E. Van Gucht

Mentor school: E. Arckens

Leerling: Robbe Dierickx

Voorwoord

Als laatstejaarsleerlingen van Elektriciteit-Elektronica hebben we de opdracht gekregen om een geïntegreerde proef (GIP) te maken. Deze proef is een belangrijk onderdeel voor het slagen in het laatste jaar van het secundair onderwijs.

Ik heb gekozen om een kruispuntverlichting te maken die gestuurd wordt met een enkele output. Ik heb hiervoor gekozen omdat ik een uitdaging wou in het hardware- en softwaregedeelte van het vak.

Graag zou ik Dhr. Arckens en Dhr. Coppejans willen bedanken, voor de hulp die ze geboden hebben bij het realiseren van mijn GIP en het antwoorden op al mijn vragen.

Ook zou ik mijn medeleerlingen willen bedanken voor het helpen met sommige technische aspecten bij de opdracht, zoals het wikkelen van de spoel.

Ten slotte wil ik mijn ouders bedanken voor hun morele steun.

1 Inhoud

2 Inleiding

Deze bundel is de schriftelijke weergave van mijn geïntegreerde proef. Het doel is het ontwerpen en realiseren van een werkend kruispunt die wordt aangestuurd met één output. Ook wordt er gebruik gemaakt van lusdetecie en een manuele voetgangersknop.

Hierbij komt er hardware en software bij kijken.

Het hardware gedeelte is waarbij ik het grootste aantal van de tijd bij heb besteed is het ontwerpen van de schema's en borden op Eagle. Deze ontwerpen werden dan verzonden naar een firma die deze borden maakten en verstuurden. De borden moesten worden gesoldeerd en gedebugd.

Het software gedeelte was het programmeren van het kruispunt en lusdetectie. Hierbij werd gebruik gemaakt van het programma Arduino. Het programma werd geüpload op een ATTiny85 met behulp van ISP.

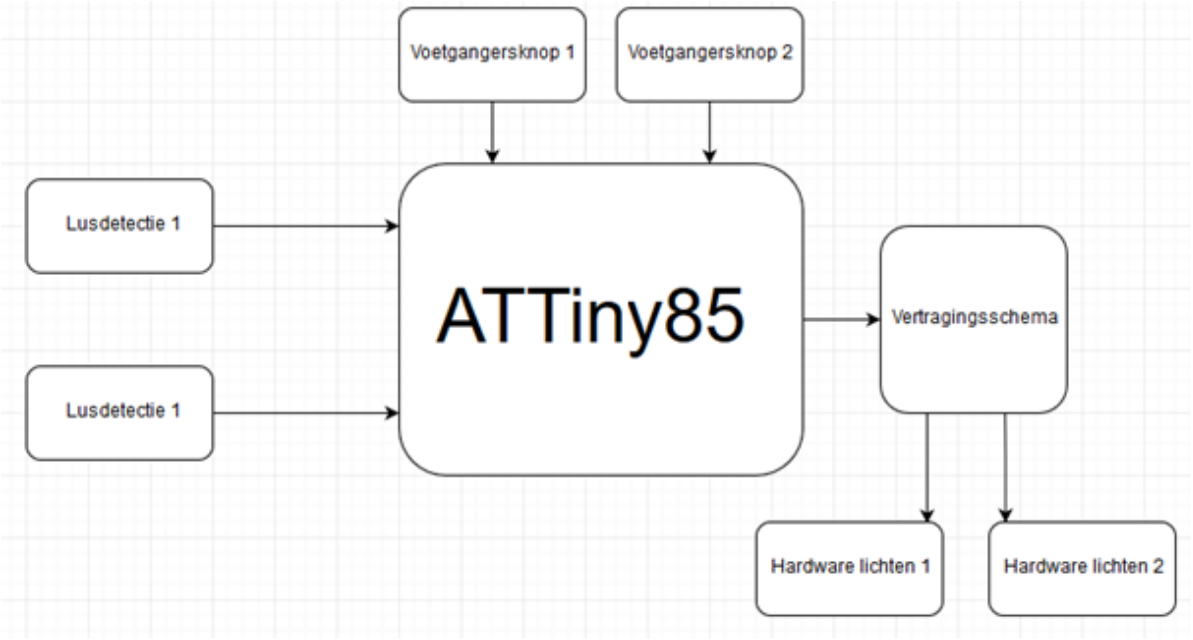
Hierbij werd individueel gewerkt.
Bij het uitwerken van de GIP diende ik rekening te houden met enkele beperkingen, zoals het beschikbare materiaal.

Voor het maken van dit eindwerk heb ik gebruik gemaakt van het internet, bijvoorbeeld de Arduino forum's. Deze bronnen waren makkelijk te bereiken, makkelijk te verstaan en bevatten de nodige informatie.

3 Omschrijving van de opdracht

Het realiseren van een automatische regeling van een kruispunt gestuurd met een enkele output met lusdetectie en een manuele voetgangersknop.

4 Blokschema hardware

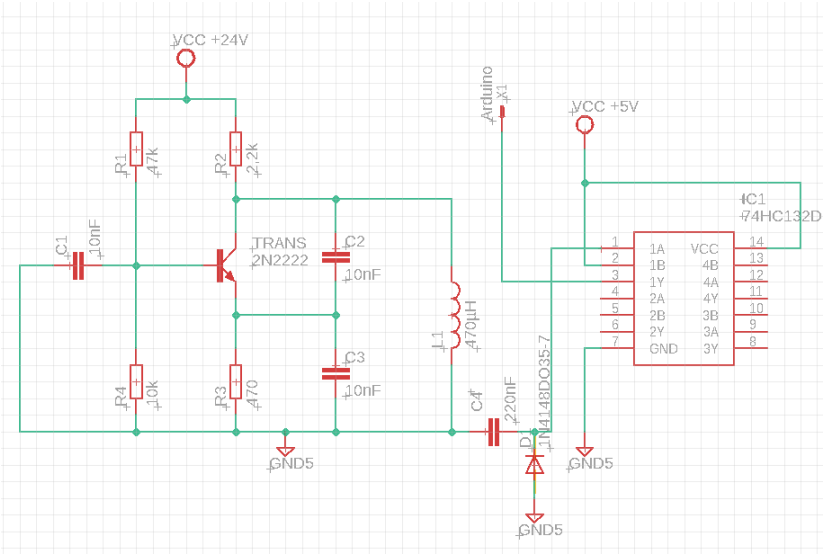


5 Lusdetectie

5.1 Versie 1

Een detectielus is een elektronische schakeling waarbij met behulp van een spoel en de verstoring van het magnetisch veld ervan doormiddel van een metalen plaat. De metalen plaat verstoord het magnetisch veld waardoor de impedantie verhoogt en dus de frequentie ook. We bekomen dit signaal met een colpitts oscillator. Om dit te meten met de Arduino hebben we een blokgolf nodig van maximaal 5V. Daarom gebruiken we een schmitt trigger om dit signaal om te vormen naar een blokgolf. Door de verhoging van de frequentie wanneer de metalen plaat erover komt kunnen we dit meten met een Arduino.

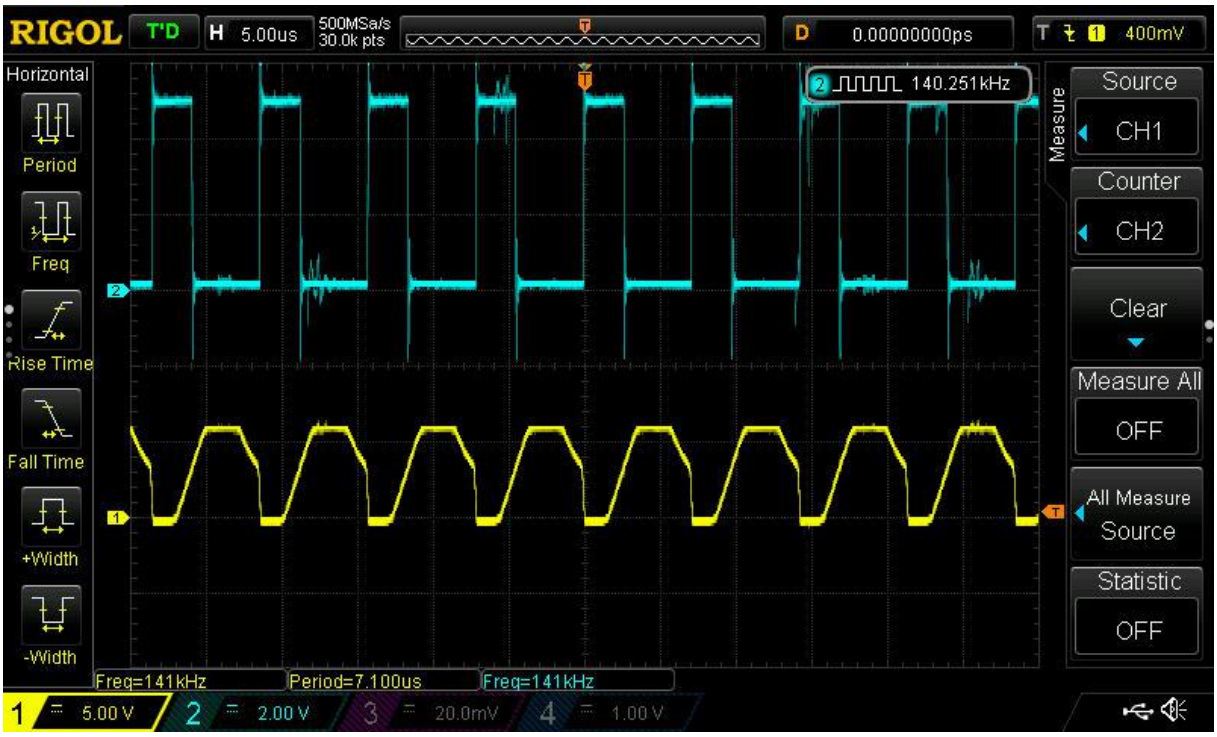
Schema:



Zelfinductie:
Zelfinductie is het verschijnsel dat een veranderende elektrische stroom door een geleider (zoals een spoel van koperdraad) een veranderend magnetisch veld opwekt, en dat veranderende magnetische veld weer een tegenspanning veroorzaakt in dezelfde geleider, die de verandering van die stroom tegengaat.

Inductantie:
Inductantie is de wisselstroomweerstand van een spoel ter grootte van ωL, waar ω de hoekfrequentie is en L de zelfinductie.

Oscilloscoop uitlezing:

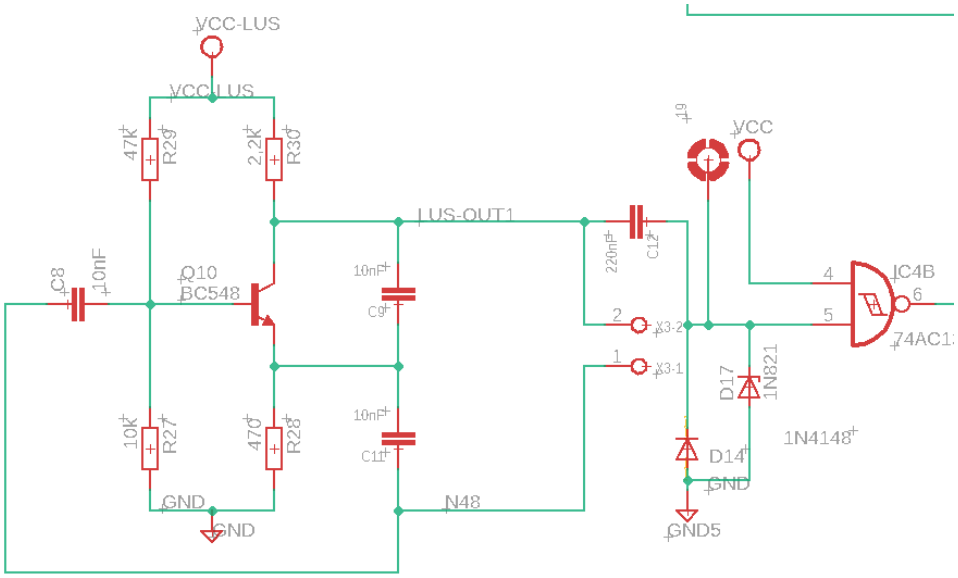


In het geel zie je het signaal van de oscillator voor de schmitt-trigger met een frequentie van 141kHz, een periode van 7,1µs en een peak to peak spanning van ongeveer 5V. In het blauw zie je het signaal van de oscillator na de schmitt-trigger die nu een blokgolf is met dezelfde frequentie, periode en peak to peak spanning als het signaal ervoor.

5.2 Versie 2

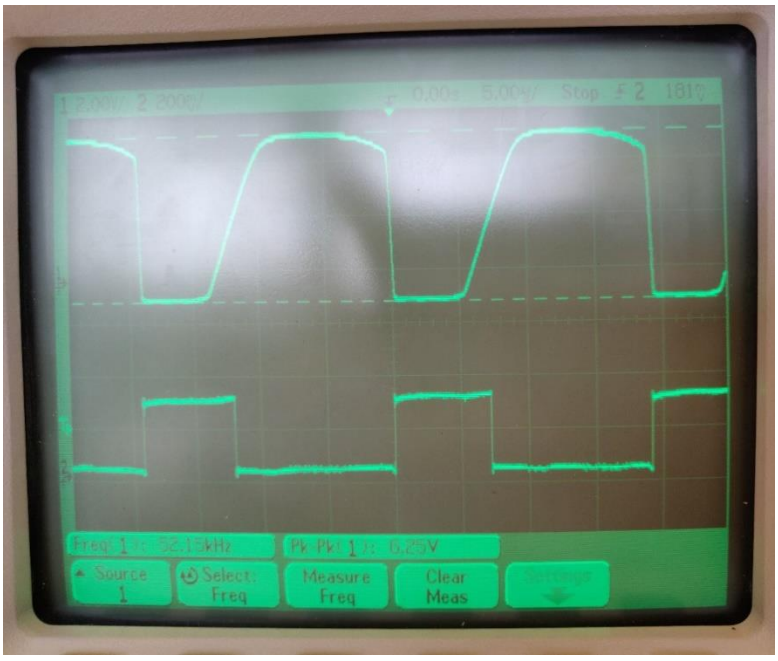
Aangezien er een fout zat in het schema van de vorige oscillator heb ik een paar aanpassingen gemaakt.

Schema:



Aanpassingen:

- De output was aan de grond gepositioneerd en is nu juist gezet.
- Een zenerdiode voor de schmitt-trigger om het spanningsniveau te verlagen.
- De dummyspoel verwijderd.
- Nieuwe spoel.

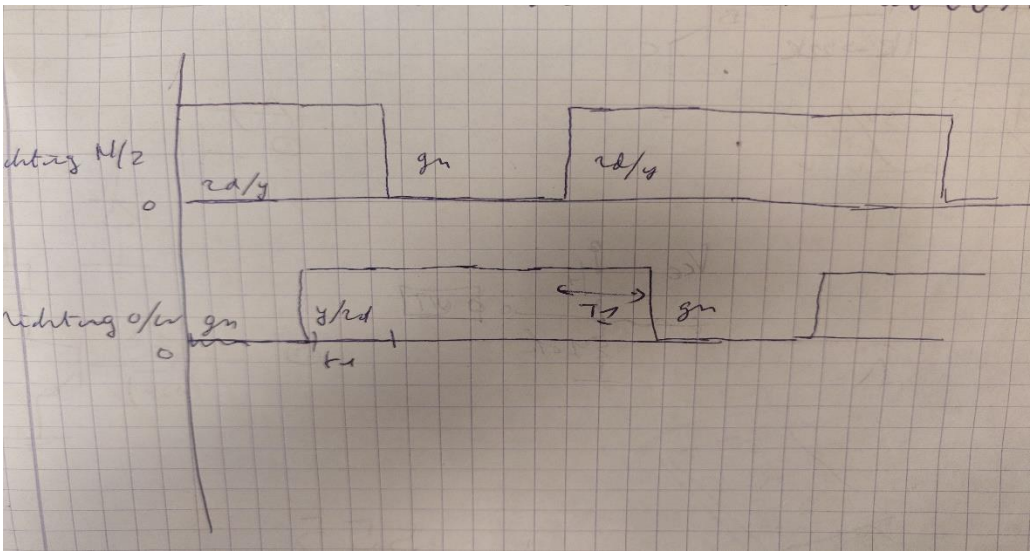


Het bovenste signaal is de output van de oscillator. Deze heeft een frequentie van 52kHz met een peak- to peak spanning van 6,25V.

Het onderste signaal is de output van de schmitt-trigger. Deze is een blokgolf van dezelfde frequentie als de sinus erboven

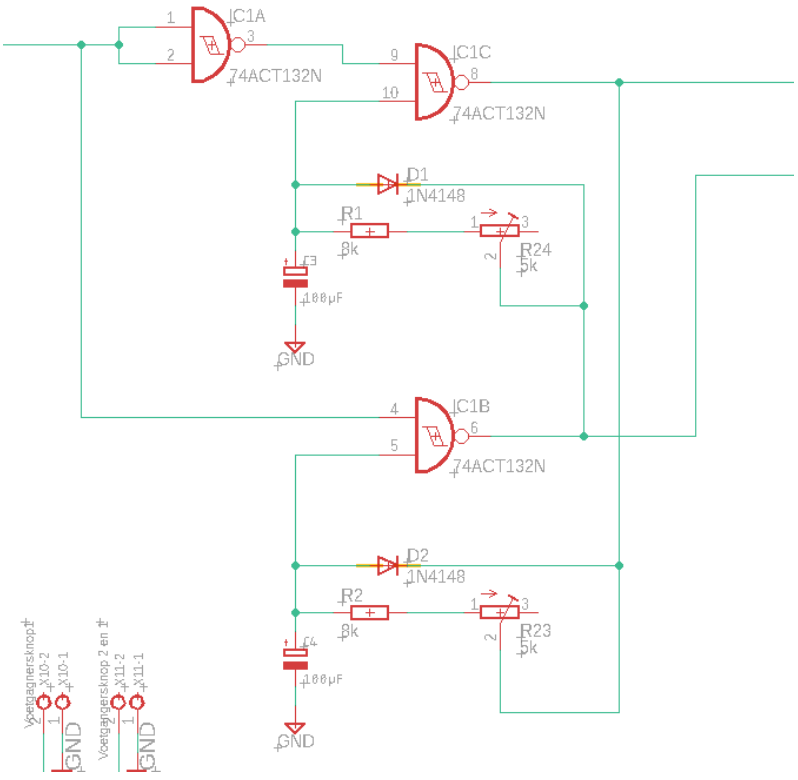
6 Vertragingsschema

Omdat ik het kruispunt met één output wil besturen heb ik een schema nodig waarbij ik dan twee signalen bekom zoals de foto hieronder.



Hiervoor gebruik ik een Shmitt Trigger en een vertraging met een RC-constante waarbij R.C = de tijdconstante in seconde is.

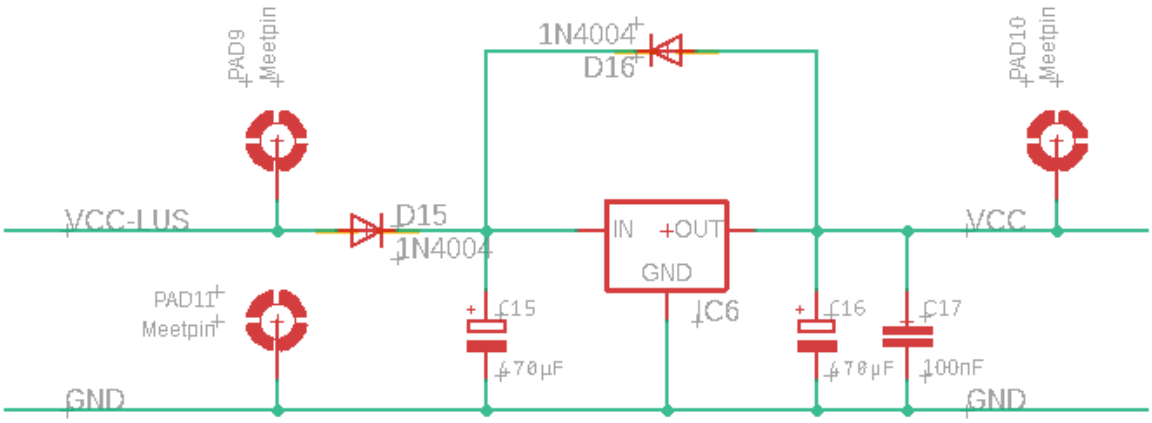
Schema:



7 24V naar 5V converter

Omdat het niet mooi is om twee verschillende bronnen te gebruiken maar ik wel met twee verschillende spanningen heb maak ik gebruik van een spanningsregelaar naar 5V.

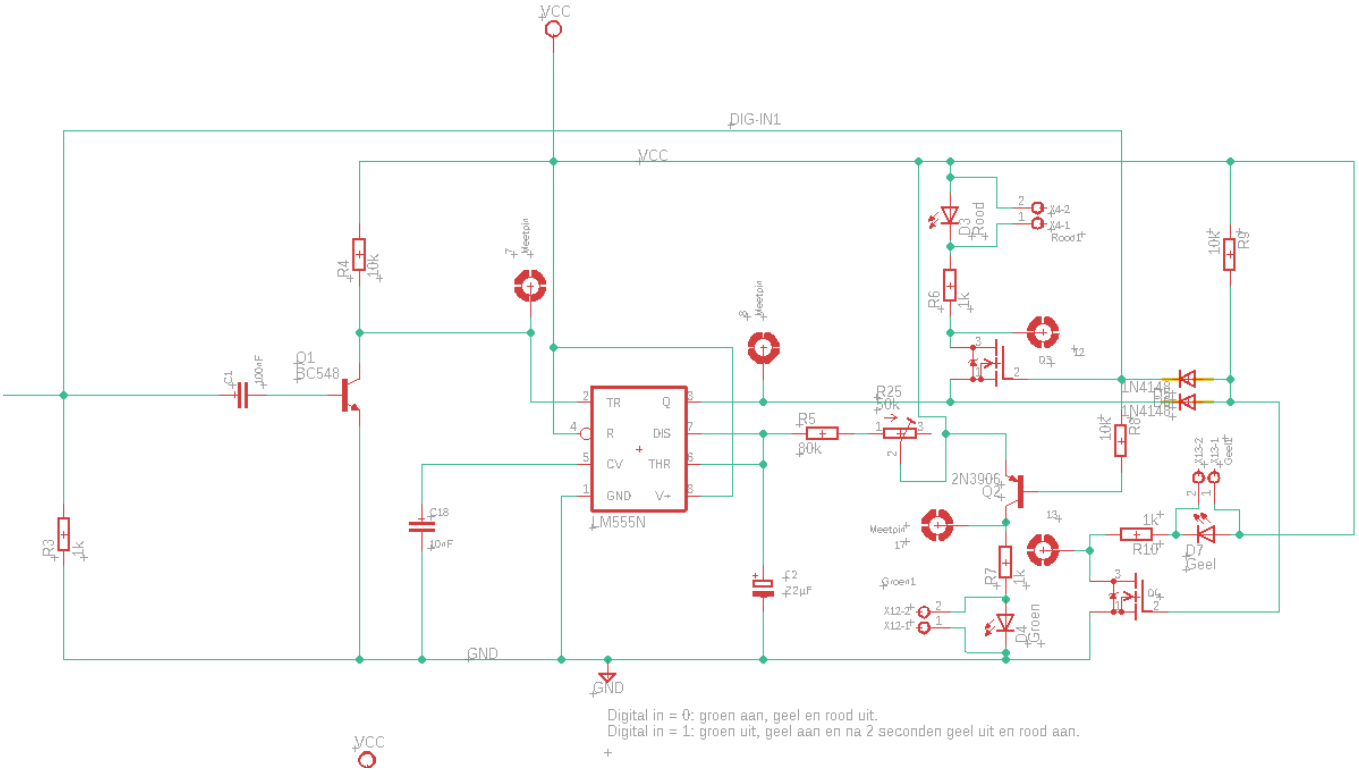
Schema:



8 Lichtsturing

Om met één signaal groen, geel en rood te besturen maak ik gebruik van een LM555.

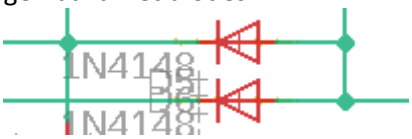
Schema:



Wanneer er aan het ingangssignaal een logische 0 komt gaat het groene licht branden en wanneer er een logische 1 komt gaat geel enkele seconden aan (tijd RC-Constante) en daarna het rode licht.

Groen:
Wanneer er een 0 aan de ingang staat wordt de transistor Q2 niet meer aangestuurd waardoor deze gaat geleiden en de groene led gaat branden.

Geel:
Wanneer er een puls wordt gegeven op poort 2 van de IC gaat er een aantal seconde een signaal aan bij poort 3 (tijd RC-Constante). Dit signaal gaat samen met het ingangssignaal door een EN-poort gemaakt met diodes.



Hierdoor wordt Mosfet Q4 aangestuurd en gaat de gele led branden.

Rood:
Wanneer het ingangssignaal 1 is wordt Mosfet Q3 aangestuurd en gaat deze geleiden waardoor de Rode led gaat branden maar als poort 3 van de IC 1 is dan is er geen spanningsverschil tussen anode en kathode waardoor geel en rood dus nooit tegelijk gaan branden.

9 Gebruikte microcontrollers

Om de sturing te regelen maak ik gebruik van twee microcontrollers, de ATTiny85 en Arduino Uno. Het originele doel was om enkel de ATTiny85 te gebruiken maar omdat deze niet genoeg timers heeft om de frequentie van de lusdetectie te meten gebruik ik ook een Arduino Uno die dit dan doorgeeft aan de ATTiny85.

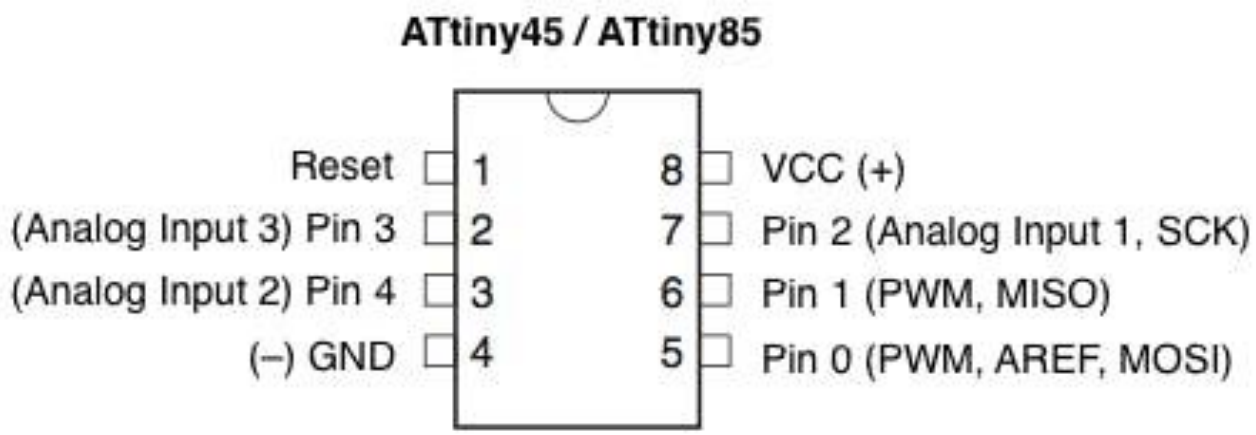
9.1 Digispark ATTiny85

De Digispark ATTiny85 is een klein microcontroller ontwikkelingsbord dat gebaseerd is op de ATmel ATTiny85. Het is te vergelijkbaar met een Arduino. Echter veel kleiner en een stuk goedkoper maar is minder krachtig. Deze programmeer ik met de Arduino IDE. De ATTiny85 heeft vijf pinnen waaronder 3 PWM pinnen. Hij kan geprogrammeerd worden met SPI, I2C en USB. Omdat de computer op school niet via USB wouw programmeren heb ik gekozen om het met SPI te programmeren waardoor pin 5 niet kan gebruikt worden als output maar als reset pin wordt gebruikt.

9.1.1 ATTiny85 programmeren met ISP

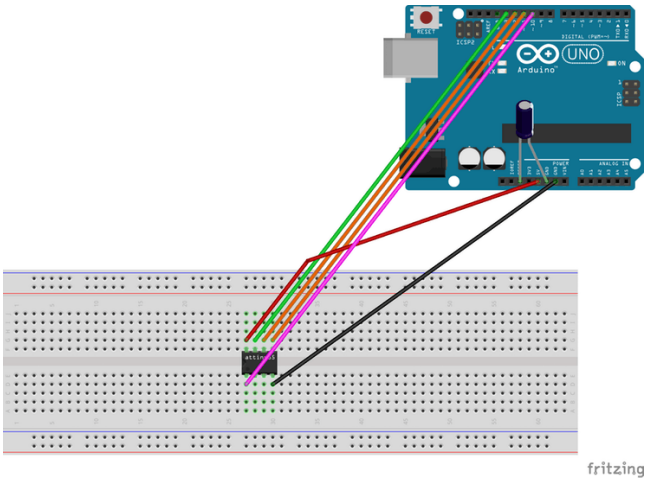
Om de ATTiny85 te programmeren zonder USB gebruik ik een Arduino met ISP verbinding.
ISP of In System Programming wil zeggen dat een microcontroller zoals de ATTiny85 geprogrammeerd kan worden zonder dat hij van de printplaat gehaald moet worden. De controller hoeft dus niet geprogrammeerd te worden voordat deze in een systeem wordt geïnstalleerd.

ATTiny85 pinconfiguratie:



Stap 1: ArduinoISP uploaden op de Arduino Uno

Stap 2: ATTiny85 verbinden met de Arduino Uno
Arduino Uno -> ATTiny85
5V -> Vcc
Gnd -> Gnd
Pin 13 -> Pin 2
Pin 12 -> Pin 1
Pin 11 -> Pin 0
Pin 10 -> Reset



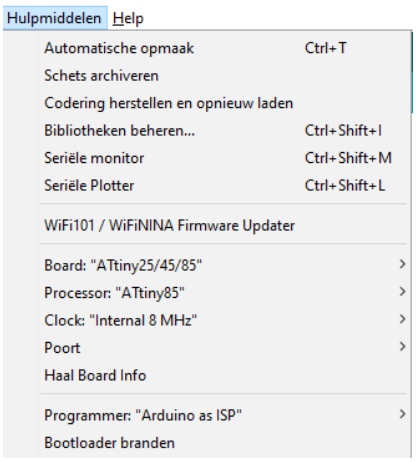
Arduino Uno
Een 10µF capaciteit tussen RESET en GND
(Om de Arduino niet te laten resetten tijdens het uploaden van het programma).

Optioneel:

Pin 7 -> Groene LED	Programmeer status
Pin 8 -> Rode LED	Error
Pin 9 -> Gele LED	Werking status

Stap 3: ATTiny85 bootloader branden
(Enkel bij het gebruiken van een nieuwe ATTiny85)

- Board: ATTiny25/45/85
- Processor: ATTiny85
- Clock: Internal 8MHz



Stap 4: Uploaden

9.2 Arduino Uno

De Arduino Uno is een microcontroller gebaseerd op de ATmega328P. Het heeft 14 Digitale Input/output pinnen waarvan 6 PWM pinnen en 6 analoge input pinnen. Verder heeft het board een 16 MHz kristal oscillator, een USB-aansluiting, een voedingsaansluiting, een ICSP header-connector en een reset knopje. Deze wordt ook geprogrammeerd met de Arduino IDE.

10 Software

Om onze microcontrollers te besturen maken programmeer ik de code met de Arduino IDE. Dit is een computerplatform bedoeld om microcontrollers programmeren eenvoudig te maken.

10.1 Frequentieteller

Het programma: “Frequentieteller” die geprogrammeerd staat op de Arduino Uno meet de frequentie die uit de lusdetectie komt doormiddel van de timers in de processor.

Programma:

```
#include <avr/interrupt.h>
#define LED 2


unsigned long previousMillis;

volatile unsigned long frequency = 0;
volatile boolean       measurement_ready;
volatile unsigned char overflow_counter;     // number of overflows within gate_time
volatile unsigned int  time_so_far;              // number of ISR calls
volatile unsigned int  gate_time;

long setupFrq;
bool firstCheck;

void measurement(int ms) {

  bitClear(TIMSK0, TOIE0);     // disable counter0 in order to disable millis() and delay()
  // this will prevent extra interrupts that disturb the measurement
  delayMicroseconds(66);       // wait for other interrupts to finish
  gate_time = ms;                 // usually 1000 (ms)

  // setup of counter 1 which will be used for counting the signal impulses

  TCCR1A = 0;                // reset timer/counter1 control register A
  TCCR1B = 0;                // reset timer/counter1 control register B
  TCCR2A = 0;                // reset timer/counter1 control register A
  TCCR2B = 0;                // reset timer/counter2 control register A

  // setup of counter2 which will be used to create an interrupt every millisecond (used for gate time)

  TCCR2B |= B00000101;         // set prescale factor of counter2 to 128 (16MHz/128 = 125000Hz)
  // by setting CS22=1, CS21=0, CS20=1

  bitSet(TCCR2A, WGM21) ;      // set counter2 to CTC mode
  // WGM22=0, WGM21=1, WGM20=0
  OCR2A = 124;                        // CTC divider will divide 125Kz by 125

  measurement_ready = 0;       // reset
  time_so_far = 0;                    // reset
  bitSet(GTCCR, PSRASY);        // reset the prescaler
  TCNT1 = 0;                           // set frequency counter1 to 0
  TCNT2 = 0;                           // set gate time counter2 to 0

  bitSet(TIMSK2, OCIE2A);        // enable counter2 interrupts
  TCCR1B |= B00000111;         // set CS12, CS11 and CS10 to "1" which starts counting
  // on T1 pin (Arduino pin D5)
}
```

```
ISR(TIMER2_COMPA_vect) {

  if (time_so_far >= gate_time) {          // end of gate time, measurement is ready
    TCCR1B &= B11111000;                  // stop counter1 by setting CS12, CS11 and CS10 to "0"
    bitClear(TIMSK2, OCIE2A);              // disable counter2 interrupts
    bitSet(TIMSK0, TOIE0);       // enable Timer0 again // millis and delay
    measurement_ready = true;              // set global flag for end count period
    // calculate now frequeny value
    frequency = 0x10000 * overflow_counter; // mult #overflows by 65636 (0x10000)
    frequency += TCNT1;                   // add counter1 contents for final value
    overflow_counter = 0;                         // reset overflow counter
  }
  else {
    time_so_far++;                             // count number of interrupt events
    if bitRead(TIFR1, TOV1)  {              // if Timer/Counter 1 overflow flag = "1" then ...
      overflow_counter++;                  // increase number of counter1 overflows
      bitSet(TIFR1, TOV1);               // reset counter1 overflow flag
    }
  };
}


void setup() {
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(5, INPUT);
  Serial.begin(9600);

}


void loop() {
  float period;
  float floatfrq;
  int range;
  long frq;


  measurement(1000);       // 1000ms standard gate time
```

```
  while (measurement_ready == false);
  frq = frequency;

  floatfrq = frq;                    // type conversion (required!!)
  period = (1 / floatfrq);              // period = 1/Frequens -

  if ((frq >= 0) && (frq < 10))            {
    range = 0;
  };                                                  // Hertz
  if ((frq >= 10) && (frq < 100))          {
    range = 1;
  };
  if ((frq >= 100) && (frq < 1000))        {
    range = 2;
  };
  if ((frq >= 1000) && (frq < 10000))      {
    range = 3;
    floatfrq = floatfrq / 1000;
  };       // KHz
  if ((frq >= 10000) && (frq < 100000))    {
    range = 4;
    floatfrq = floatfrq / 1000;
  };
  if ((frq >= 100000) && (frq < 1000000)) {
    range = 5;
    floatfrq = floatfrq / 1000;
  };
  if (frq >= 1000000)                     {
    range = 6;
    floatfrq = floatfrq / 1000000;
  };           // MHz

  if (firstCheck == 0) {
    firstCheck = 1;
    setupFrq = frq;
    Serial.println("SET!");
  }

  Serial.print("Frequency (Hz): ");
  Serial.print(frq);
  Serial.print("     Period (sec): ");
  Serial.println(period, 7);



  if (frq > (setupFrq + 2500)) {
    digitalWrite(2, HIGH);
    digitalWrite(3, HIGH);
  }
  else {
    digitalWrite(2, LOW);
    digitalWrite(3, LOW);
  }

}
```

10.2 Kruispunt

Het programma “Kruispunt” is het programma dat geprogrammeerd is op de ATTiny85. Dit programma geeft de timing aan om van groen naar rood te springen. Deze timing is afhankelijk van de input van de lusdetectie en voetgangersknoppen.

Programma:

```
/*
 * Het programma om de lichtsturing aan te sturen met een ATTiny85.
 *
 * P0 -> Lusdetectie 1
 * P1 -> Lichtsturing
 * P2 -> Lusdetectie 2
 * P3 -> Voetgangersknop 1
 * P4 -> Voetgangersknop 2
 *
 */



#define LIGHTS 1
#define BUTTON1 3
#define BUTTON2 4
#define LOOP1 0
#define LOOP2 2

unsigned long previousMillis = 0; // Variabele om de vorige tijd bij te houden
const int interval1 = 30000;      // Variabele die het interval tussen de lichtwisseling weergeeft
const int interval2 = 5000;       // Variabele die het interval tussen de lichtwisseling weergeeft wanneer er een knop is ingedrukt / Lusdetectie
bool onOff = true;                // Variabele die weergeeft of het licht aan/uit is
bool buttonPress = false;         // Variabele die weergeeft of de knop is ingedrukt



void setup()
{
  pinMode(LIGHTS, OUTPUT);
  pinMode(LOOP1, INPUT);
  pinMode(LOOP2, INPUT);
  pinMode(BUTTON1, INPUT);
  pinMode(BUTTON2, INPUT);
}

void loop()
{
  if ((millis() - previousMillis >= interval1) || ((buttonPress && (millis() - previousMillis >= interval2))))
  {
    previousMillis = millis();
    buttonPress = false;
    digitalWrite(LIGHTS, onOff);
    onOff = !onOff;
  }
  if (((digitalRead(BUTTON1) && onOff) || (digitalRead(BUTTON2) && !onOff) || (digitalRead(LOOP1) && onOff) || (digitalRead(LOOP2) && !onOff)) && (!buttonPress  && ((millis() - previousMillis) <= (interval1 - interval2))))
  {
    buttonPress = true;
    previousMillis = millis();
  }
}
```

11 Besluit en zelfreflectie

12 Logboek

Schoolweek : 45

Datum	Vak	Omschrijving van de studie, taak of voorbereiding
6/11	EE	Opnieuw testen van de lusdetectie met dummiespoel, Spoel winden, testen met de echte spoel zonder succes
8/11	EE	Testen van de oscilator met de spoel en het uitvoeren van de lusdetectie met arduino nano dmv een led aan te sturen. Schakeling kapot.
9/11	EE	Maken van het oschillatorschema met ipv een shmitt trigger een Emittervolger.

Schoolweek : 46

Datum	Vak	Omschrijving van de studie, taak of voorbereiding
12/11	EE	Testen van de transistorschakelingen
13/11	EE	Verder testen van de transistorschakeling + maken van het update verslag van de oscillator.
15/11	EE	Verder testen van de zelfgewonden spoel: Spanning verhogen, parallel schakelen, serie schakelen. Conclusie: nieuwe spoel winden.
16/11	EE	Maken van het schema: Hardware lichtsturing. Denken om een oplossing om zo weinig mogelijk I/O's te gebruiken.

Schoolweek : 47

Datum	Vak	Omschrijving van de studie, taak of voorbereiding
20/11	EE	Maken en testen van het vertragingsschema + maken van het schema in Eagle. Aantal gewerkte uren: 2,5
22/11	EE	Maken + testen van het schema voor de lichtsturing, conclusie: Pin 3 van de 555 timer doet niks. Aantal gewerkte uren: 3
23/11	EE	Testen van het aangepaste schema voor de lichtsturing, conclusie: Het ingangssignaal ging niet helemaal naar 0V waardoor pin3 niks deed. De gele led doet nog niet wat hij moet doen. Aantal gewerkte uren: 3

Schoolweek : 48

Datum	Vak	Omschrijving van de studie, taak of voorbereiding
26/11	EE	Verbeteren van het schema van de lichtsturing + De library samenstellen voor het eindbord. Aantal gewerkte uren: 2
27/11	EE	Testen van het schema van de lichtsturing en het maken van het programma ervoor (Drivers installeren voor de ATTiny85) + verder werken aan het eindschema. Aantal gewerkte uren: 3.
29/11	EE	Maken van het eindschema (afgewerkt). Aantal gewerkte uren: 3.
30/11	EE	Maken van het bord van het eindschema. Aantal gewerkte uren: 3.

Schoolweek : 49

Datum	Vak	Omschrijving van de studie, taak of voorbereiding
3/12	EE	Maken van het eindbord met toevoeging van een 24V -> 5V schakeling. Aantal gewerkte uren: 2
4/12	EE	Afwerken van het eindbord. Aantal gewerkte uren: 3

Schoolweek : 1 & 2

Datum	Vak	Omschrijving van de studie, taak of voorbereiding
5/01	EE	Aanpassen van het schema en bord. Aantal gewerkte uren: 2
11/01	EE	Verder aanpassen van het schema en bord + exporteren naar een gerber file. Aantal gewerkte uren: 3

Schoolweek : 3

Datum	Vak	Omschrijving van de studie, taak of voorbereiding
14/01	EE	Opzoekwerk ATTiny85: Interrupt, Timers, Frequentieteller aantal gewerkte uren: 2
15/01	EE	Verder zoeken naar een oplossing voor de frequentieteller + programmeren van het programma: Kruispunt. Aantal gewerkte uren: 3
17/01	EE	Uitzoeken van de inputs van de ATTiny85 + programmeren van het programma: Kruispunt. Aantal gewerkte uren: 2.
18/01	EE	Verder uitzoeken van de inputs van de ATTiny85 + programmeren van het programma: Kruispunt. Aantal gewerkte uren: 3.

Schoolweek : 4

Datum	Vak	Omschrijving van de studie, taak of voorbereiding
21/01	EE	Programmeren van de ATTiny85. Aantal gewerkte ren: 2
22/01	EE	Drivers proberen installeren van de ATTiny85 + Een OR-Gate maken van diodes + opzoeken: programmeren van de ATTiny85 met ISP . Aantal gewerkte uren: 3
24/01	EE	Programmeren van de ATTiny85 (en de ATTiny85 Chip) met ISP. Aantal gewerkte uren: 2
25/01	EE	Programmeren van de ATTiny85 (en de ATTiny85 Chip) + resetten van de fuses Aantal gewerkte uren: 4

Schoolweek : 5

Datum	Vak	Omschrijving van de studie, taak of voorbereiding
28/01	EE	Aanpassing van het programma: Kruispunt + programmeren van de ATtiny85 met AVRDude. Aantal gewerkte: 2
29/01	EE	Programmeren van de ATTiny85 met AVRDude + Opzoeken van frequentietelling met de ATTiny85. Aantal gewerkte uren: 3
31/01	EE	Opzoeken van frequentietelling met de ATTiny85. Aantal gewerkte uren: 2

Schoolweek : 6

Datum	Vak	Omschrijving van de studie, taak of voorbereiding
05/02	EE	Solderen van de printplaat. Aantal gewerkte uren: 2
07/02	EE	Solderen van de printplaat. Aantal gewerkte uren: 3

Schoolweek : 7

Datum	Vak	Omschrijving van de studie, taak of voorbereiding
11/02	EE	Solderen van de printplaat. Aantal gewerkte uren: 2
12/02	EE	Testen en debuggen van de printplaat, wikkelen van de nieuwe spoelkernen en maken van de colpittsoscillator op een breadbord om deze te testen. Aantal gewerkte uren: 4
14/02		Testen van de colpittsoscillator met de 2 verschillende spoelen + schmitt triggers probleem proberen oplossen + aanpassingen schema. Aantal gewerkte uren: 3

Schoolweek : 8

Datum	Vak	Omschrijving van de studie, taak of voorbereiding
19/02	EE	Verder testen van de colpittsoscillator + testen van het programma: Lusdetectie + versie 2 van de printplaat aanpassen. Aantal gewerkte uren: 3
21/02	EE	Opstellen van de Demo van de lusdetectie voor de jury + verder aanpassen van Versie 2 van de printplaat. Aantal gewerkte uren: 3
22/02		Afwerken van versie 2 van de printplaat en omzetten in gerber files + opzoeken om de frequentie te meten met een ESP8266. Aantal gewerkte uren: 4

Schoolweek : 9

Datum	Vak	Omschrijving van de studie, taak of voorbereiding
25/02	EE	Aanpassen van het programma Lusdetectie om een setup frequentie toe te voegen + samenwerking met het programma kruispunt. Aantal gewerkte uren: 2
26/02	EE	Programmeren van de ATTiny85 met ArduinoISP + Samenwerking van de arduino en ATTiny85 maken. Aantal gewerkte uren: 2
28/02	EE	Verder testen van de samenwerking van lusdetectie en de lichtsturing. Aantal gewerkte uren: 2

Schoolweek : 11

Datum	Vak	Omschrijving van de studie, taak of voorbereiding
11/03	EE	Componenten van de oude printplaat halen. Aantal gewerkte uren: 2
12/03	EE	Componenten van de oude printplaat halen. Aantal gewerkte uren: 2

Schoolweek : 12

Datum	Vak	Omschrijving van de studie, taak of voorbereiding
21/03	EE	Solderen van de printplaat. Aantal gewerkte uren: 1

Schoolweek : 13

Datum	Vak	Omschrijving van de studie, taak of voorbereiding
21/03	EE	Solderen van de printplaat. Aantal gewerkte uren: 1
22/03	EE	Solderen van de printrplaat. Aantal gewerkte uren: 2
28/03	EE	Solderen van de printrplaat. Aantal gewerkte uren: 2
29/03	EE	Solderen van de printrplaat (afgewerkt). Aantal gewerkte uren: 3

Schoolweek : 14

Datum	Vak	Omschrijving van de studie, taak of voorbereiding
01/04	EE	Debuggen van de printplaat. Aantal gewerkte uren: 2
02/04	EE	Debuggen van de printplaat (lichtsturing: LM555) . Aantal gewerkte uren: 1

Schoolweek : 17

Datum	Vak	Omschrijving van de studie, taak of voorbereiding
23/04	EE	Maken van versie 3 van de printplaat. Aantal gewerkte uren: 2

13 Bijlagen

13.1 GIP-opdracht godsdienst

Normen en waarden op het werkterrein

Beroepswereld:

Ik zou graag als softwareontwikkelaar werken. Hiervoor ga ik Elektronica-ICT gaan studeren met als hoofdvak ICT. Bij dit beroep moet je de software ontwikkelen, testen en onderhouden voor het bedrijf waar je voor werkt of voor verschillende opdrachtgevers. Hierbij moet je luisteren naar de wensen van de klanten en dit dan omzetten in toepassingen op maat. Dit zijn professionele computerprogramma's, databases, webpagina's, ….

Beroepsethiek:

Software speelt meer en meer een rol in het dagelijkse leven waardoor ethische uitdagingen worden groter en groter. Er zit software in thermostaten, koelkasten, auto's, … en wanneer deze niet goed beveiligd zijn kan dit grote problemen veroorzaken. Ontwikkelaars komen dagelijks in contact met ethiek ook al zijn ze er niet altijd van bewust. Daarom moet je als softwareontwikkelaar er goed over na denken.

Logbestanden:
Elke actie die een programma uitvoert wordt gedocumenteerd om het systeem te kunnen debuggen. Deze logbestanden houden vaak ook bij wat gebruikers allemaal doen en bevatten daarom regelmatig informatie die gebruikers liever voor zichzelf houden. Veel bedrijven zijn hier afhankelijk aan. Het bestaan van deze logbestanden geeft ons ook ethische vragen zoals wie hier allemaal toegang tot heeft. Veel informatie wordt verkocht aan andere bedrijven die daarvan goed gebruik kunnen maken.

Beveiliging:
Wanneer je een goede beveiliging wil hebben wordt het programmeren zelf veel lastiger. Als er ook maar een bit veranderd of een deel van het algoritme een fout bevat raak je al je gegevens kwijt doordat ze niet meet te ontsleutelen zijn. Daardoor kiezen sommigen ervoor om gewoon helemaal niet te beveiligen. Degene die zich niet druk maken over de privacybescherming maken ook programma's die veel mensen gebruiken omdat het ook verkoopt. Meer beveiliging is beter tot wanneer die ervoor zorgt dat je data kwijtgeraakt.

Misbruik:
Hackers kunnen jouw producten altijd misbruiken. Bijvoorbeeld webcams in laptops kunnen worden gebruikt om iemand af te luisteren. Hiervoor hebben meeste laptops een lampje die aanstaat wanneer deze wordt gebruikt maar er is altijd wel een manier om deze ook uit te schakelen. De uitdaging die ontwikkelaars dus hebben is om deze misbruiken te voorkomen.

Arbeidsattitudes:

Softwareontwikkelaars werken ook samen in een team waarbij iedereen zijn taken heeft. Deze moeten goed verdeeld zijn. Je moet natuurlijk ook proberen goed overeen te komen met je collega's. Deze collega's hebben ook ideeën waarvoor je je moet kunnen laten openstaan. Je opdrachtgevers die al hun ideeën geven moet je ook respecteren en vriendelijk tegen zijn.

Arbeidscompetenties:

Als softwareontwikkelaar moet je klantgericht zijn, je moet naar al de wensen van je klanten luisteren en deze toepassen. Je moet nauwkeurig zijn, je moet je code goed verzorgen en alles documenteren. Creativiteit is ook een must aangezien je altijd een oplossing moet bedenken voor een mogelijk probleem. Omdat in de IT-wereld constant alles veranderd moet je natuurlijk ook leergierig zijn om alle nieuwe technologieën te kunnen toepassen. Je moet zelfstandig kunnen werken en zelf je problemen opzoeken. Aangezien je in teams werkt is kunnen samenwerken nodig. Door de vele bugs die zouden kunnen opduiken moet je kunnen problemen onderzoeken en deze zo efficiënt mogelijk kunnen oplossen. De klanten geven vaak ook een deadline die je moet halen, daarom moet je ook alles goed inplannen en moet je deze ook volgen om in geen tijdsnood te zitten