Handleiding

Countdown Timer

Inhoudsopgave

Welkom

Wat leuk dat je een Awesome Crate gaat bouwen!

Deze crate bevat de benodigdheden en uitleg om een countdown timer te bouwen waarmee je bijvoorbeeld kunt aftellen tot het moment dat je weer eens moet opstaan van je werkplek. Of je kunt er een zelfgemaakte stopwatch mee bouwen!

Deze kit is gericht op volwassenen. Uiteraard kan de kit ook door kinderen in elkaar gezet worden. Het kind dient dan wel altijd begeleid te worden door een volwassene.

In jouw Awesome Crate zit een microcontroller, een kleine computer waar we later meer over gaan uitleggen.

ⓘ Als je al eerder een van onze Awesome Crates hebt gebouwd zul je sommige uitleg al eens gelezen hebben. Voel je vrij om deze over te slaan, of juist nog een keer te lezen als opfrisser.

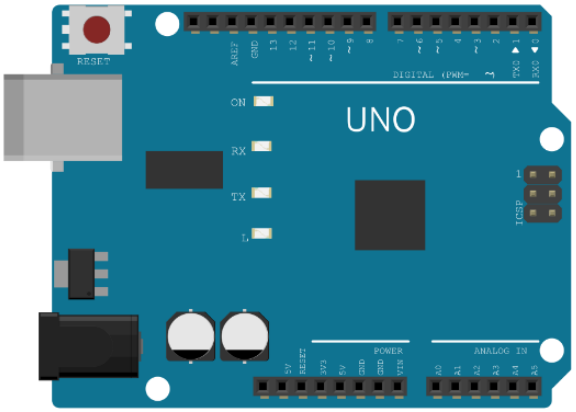
Veiligheidsregels

Om zo veel mogelijk plezier te beleven zijn hier een paar regels die ervoor zorgen dat de microcontroller niet beschadigt:

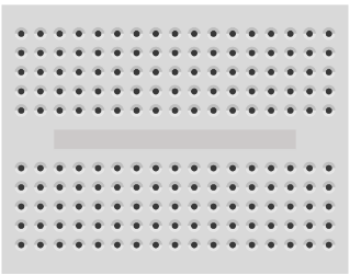
- Raak de microcontroller niet aan terwijl deze aan is.
- Zorg er altijd voor dat de microcontroller uit is (stroomkabeltje niet verbonden) voordat je onderdelen inplugt.
- Sluit de USB kabel eerst aan op de microcontroller en daarna pas op de stroombron (computer, USB stekker, of power bank). Daarmee voorkom je dat je de microcontroller aanraakt terwijl er stroom op staat.
- Plaats de microcontroller op een oppervlak dat geen stroom geleidt voordat je hem aan zet. Gebruik bijvoorbeeld een houten tafel, plastic mat of een tijdschrift.
- Schakel de microcontroller altijd uit nadat je er klaar mee bent.

ⓘ Dit is geen speelgoed. Bij het gebruik van dit artikel dienen kinderen onder de 14 jaar altijd begeleid te worden door een volwassene. Houd altijd toezicht op het product tijdens gebruik. Ontkoppel een eventuele stroombron na gebruik.

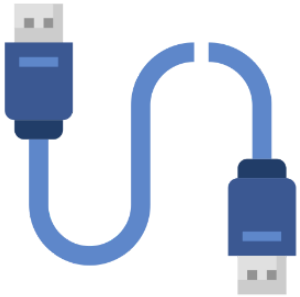
Microcontroller



Breadboard

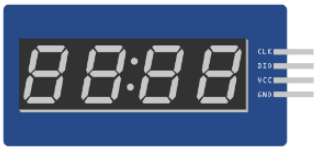


USB kabel



Weerstandjes
100 Ω

7 segment klok

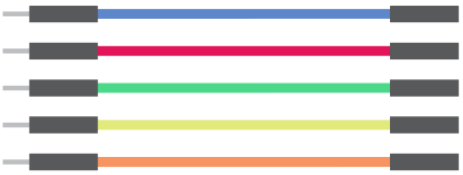


Knop

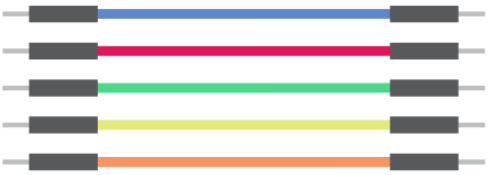
Kapje

Buzzer

Jumper kabels
m-v



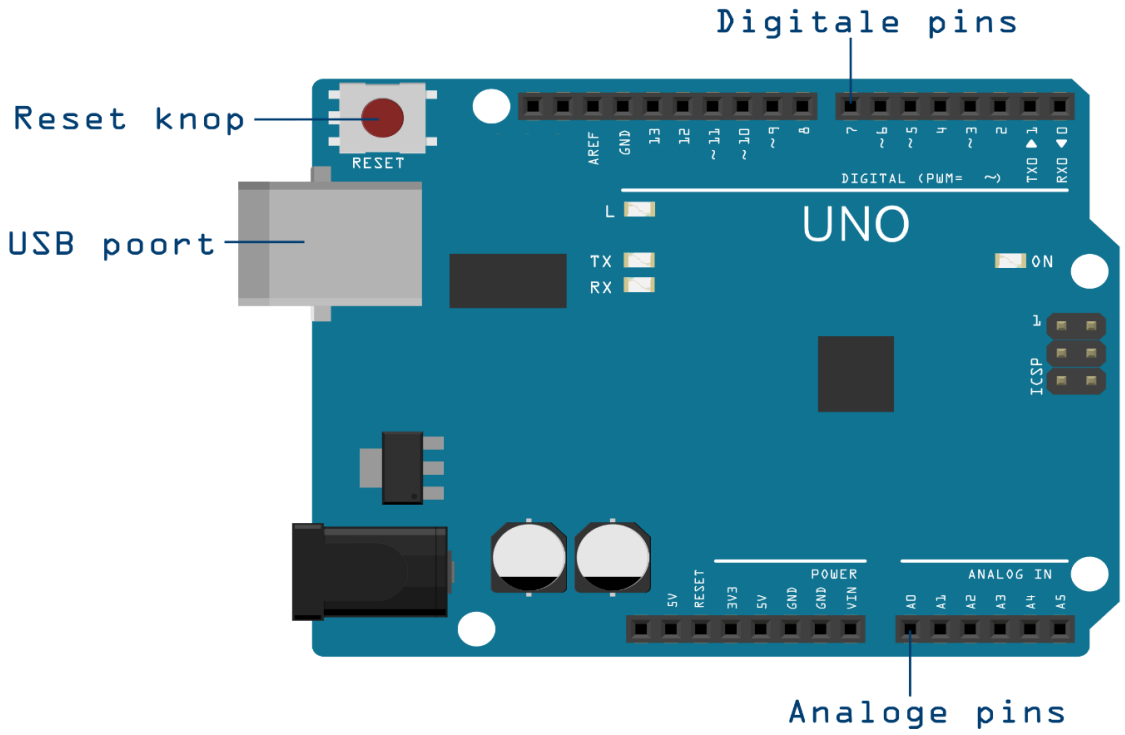
Jumper kabels
m-m



De microcontroller

Het belangrijkste onderdeel in jouw Awesome Crate is de microcontroller. Dit is een klein computertje waarmee je andere elektronica kunt besturen. De microcontroller kan bijvoorbeeld een lampje aan en uit doen, of een motortje rond laten draaien.

De andere elektronica, bijvoorbeeld een lampje, kun je aansluiten op de microcontroller door deze in een pin te prikken. De pins van de microcontroller zijn de gaatjes in de zwarte plastic opstaande randen van de microcontroller. (Zie Digitale pins en Analoge pins op het plaatje)



Afbeelding 0.1

Arduino IDE installeren

We gaan straks de microcontroller programmeren. Hiervoor gebruiken we de Arduino IDE (Integrated Development Environment). De Arduino IDE is een applicatie waarmee je code voor een microcontroller kunt schrijven en waarmee je de code op de microcontroller kunt zetten.

Gebruik je Mac?

Ga dan verder bij Arduino IDE installeren op Mac (blz. 5).

Gebruik je Windows?

Ga dan verder bij Arduino IDE installeren op Windows (blz. 7).

Arduino IDE installeren op Mac

Stappen

1. Ga in de browser naar https://www.arduino.cc/en/software.

2. Klik in het vak "DOWNLOAD OPTIONS" (rechts op de pagina) op Mac OS X.

3. Je kan er voor kiezen om een bijdrage te leveren aan de ontwikkeling van de Arduino IDE maar dit is niet verplicht. Wanneer je geen bijdrage wenst te leveren klik je op de knop "JUST DOWNLOAD".

4. De Arduino IDE wordt nu gedownload. Verplaats het bestand daarna naar de applicatie map.

5. Open het programma. Het kan zijn dat jouw besturingssysteem nu om toestemming vraagt om de applicatie te openen. Geef in dit geval toestemming hiervoor.

6. De Arduino IDE wordt nu geopend.

Arduino IDE installeren op Windows

Stappen

1. Ga in de browser naar https://www.arduino.cc/en/software.

2. Klik in het vak "DOWNLOAD OPTIONS" (rechts op de pagina) op Windows (Win 7 and newer).

3. Je kan er voor kiezen om een bijdrage te leveren aan de ontwikkeling van de Arduino IDE maar dit is niet verplicht. Wanneer je geen bijdrage wenst te leveren klik je op de knop "JUST DOWNLOAD".

4. De Arduino IDE wordt nu gedownload.

5. Open het bestand (deze staat in je Downloads map). Het kan zijn dat jouw besturingssysteem nu om toestemming vraagt om de applicatie te openen. Geef in dit geval toestemming hiervoor.

6. Volg de stappen in de installatiewizard, je kunt alle standaardwaarden gebruiken.

7. Je besturingssysteem vraagt mogelijk meerdere keren om verschillende onderdelen te installeren. Geef hiervoor toestemming.

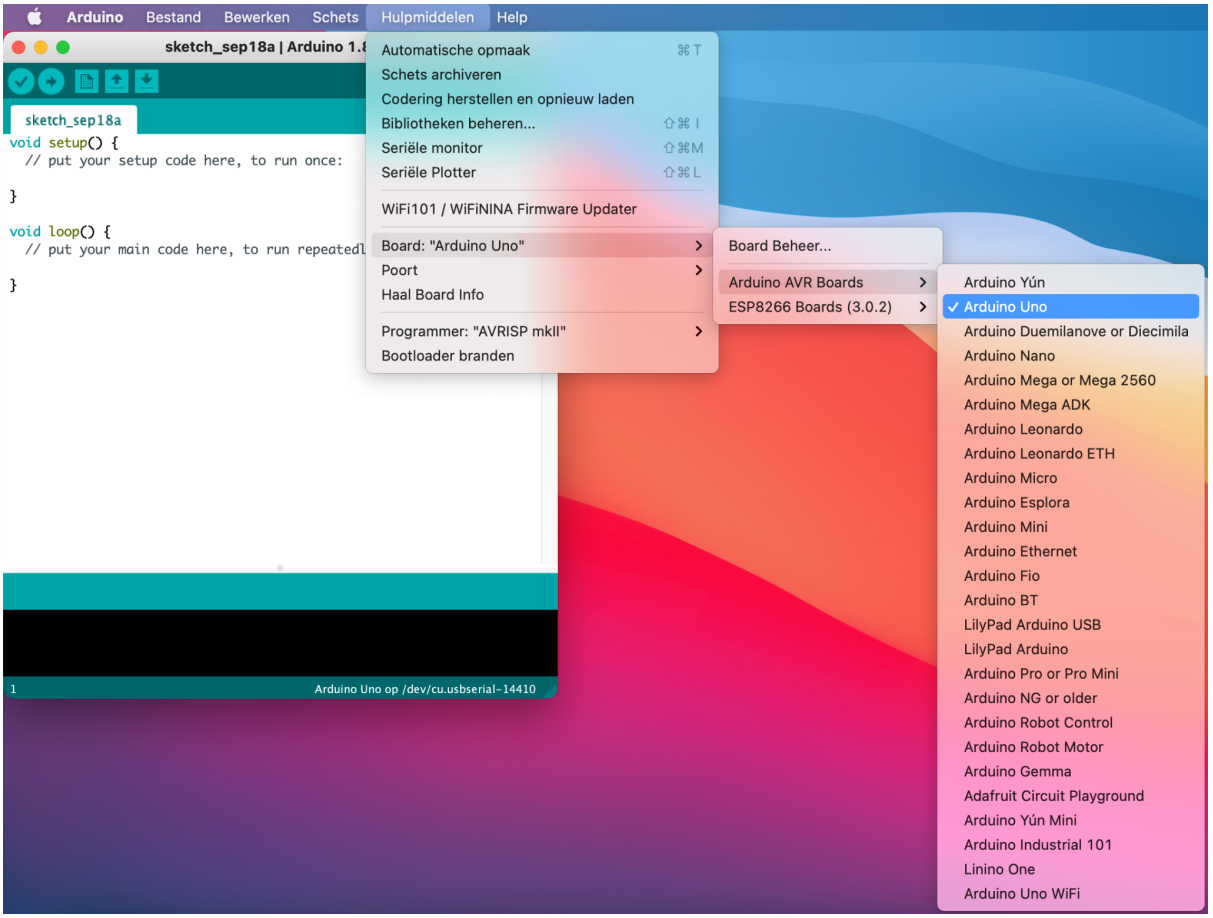
8. Open nu de Arduino IDE. Mogelijk vraagt je firewall om toestemming, geef ook hier toestemming.

Het juiste bord kiezen

Voor dit project gebruiken we als microcontroller een Arduino Uno. Er zijn een heleboel andere soorten microcontrollers die je ook kunt programmeren via de Arduino IDE. Daarom moeten we in de IDE het juiste bord selecteren.

Stappen

1. Klik in de menubalk op ***Hulpmiddelen*** en vervolgens op ***Board***, dan op ***Arduino AVR Boards*** en selecteer ***Arduino Uno***.



Mooi, de Arduino IDE is nu geconfigureerd. We kunnen de microcontroller nu op de computer gaan aansluiten.

De microcontroller op de computer aansluiten

Nu gaan we de microcontroller via een USB kabel verbinden met de computer. Dit is de laatste stap voordat we code kunnen uploaden naar de microcontroller.

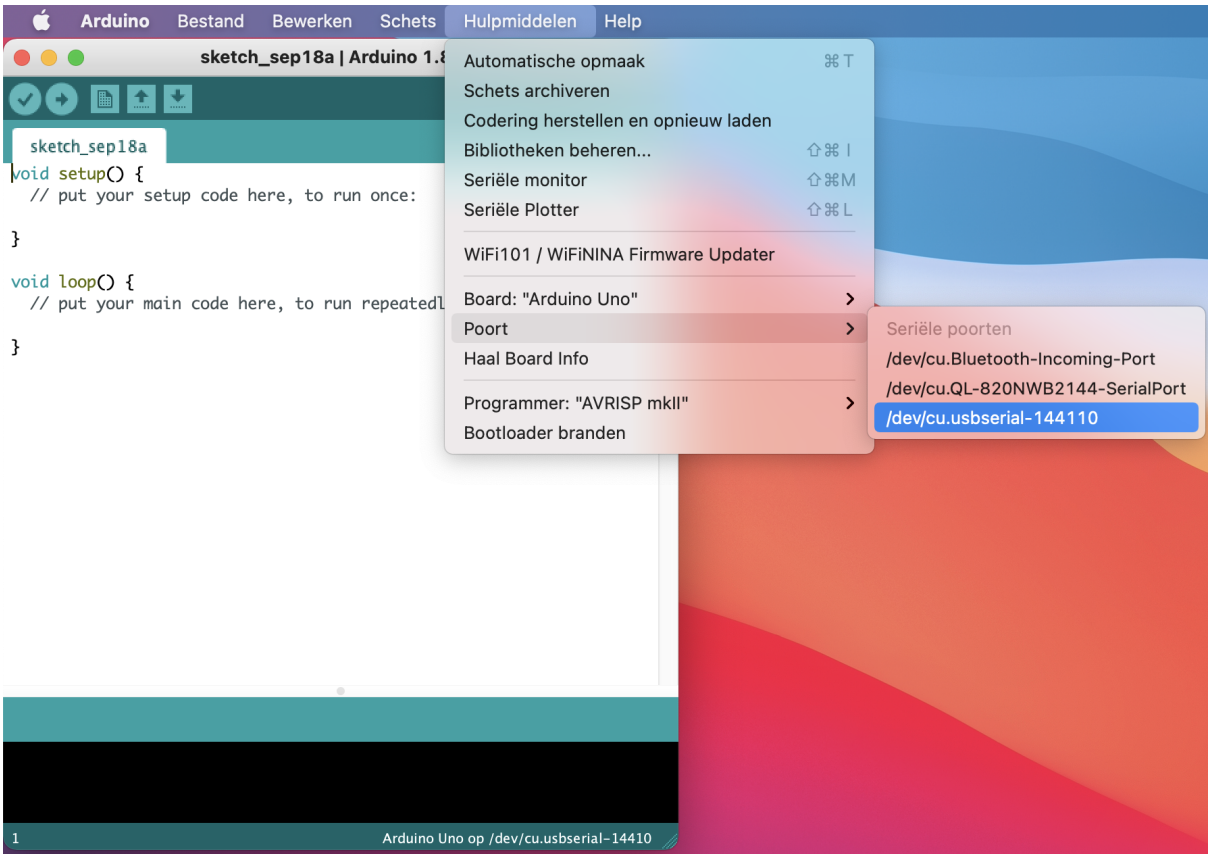
Stappen

Het is belangrijk om bij het aansluiten van de microcontroller de veiligheidsregels (blz. 1) na te leven.

1. Sluit de USB kabel eerst aan op de microcontroller en vervolgens op je computer.

2. Klik in de menubalk op ***Hulpmiddelen*** en vervolgens op ***Poort***. Selecteer nu de poort waarop de microcontroller is aangesloten. Deze poort zal een naam hebben zoals "*cu.usbserial-14410*"(op Mac) of *"COM1"* (op Windows).

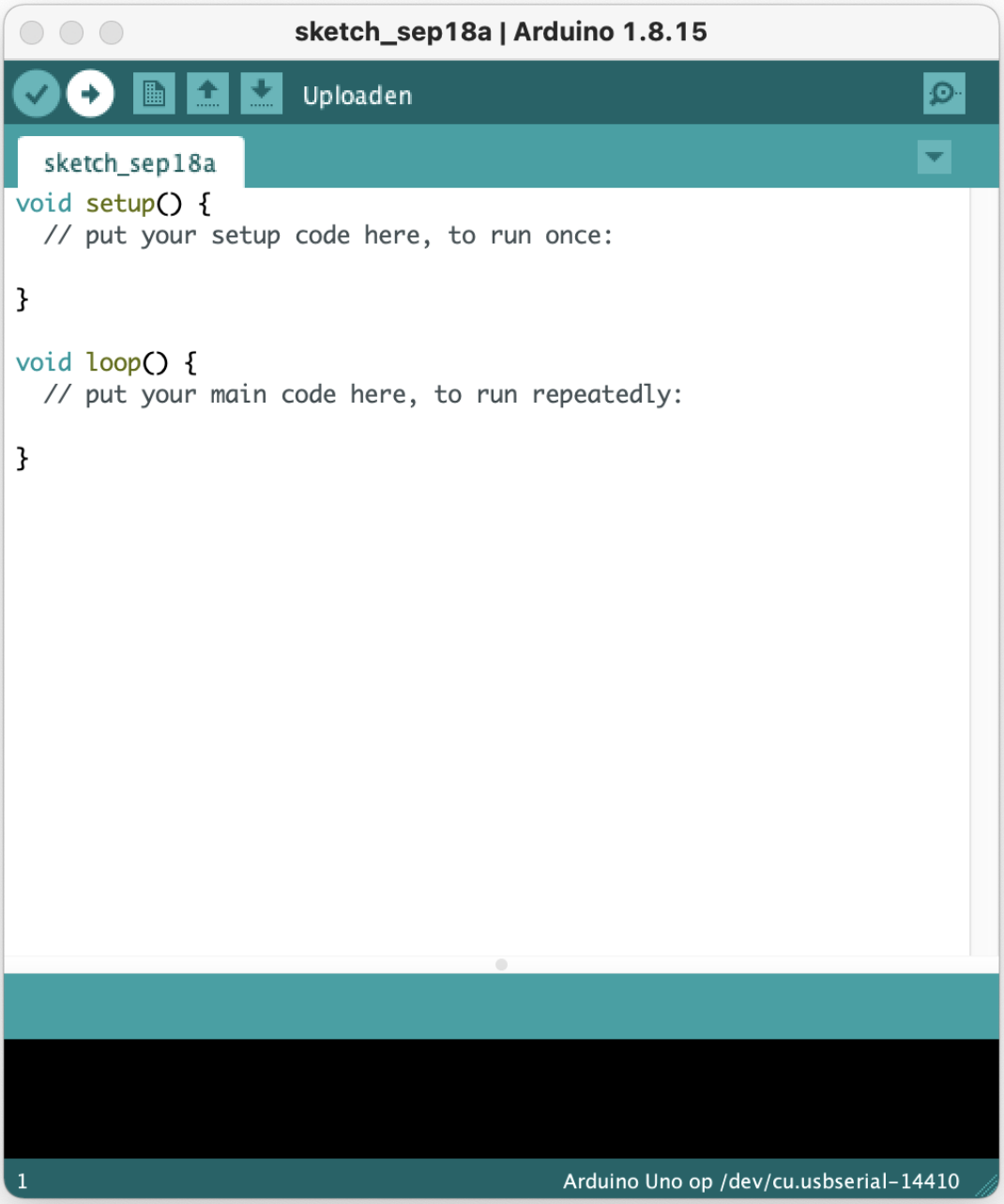
   Lukt het niet om te verbinden met de microcontroller? Dan kan het zijn dat je eerst nog de CH340 driver moet installeren. Ga hiervoor naar https://www.awesomemakes.com/drivers/ en download en installeer de driver voor jouw besturingssysteem.



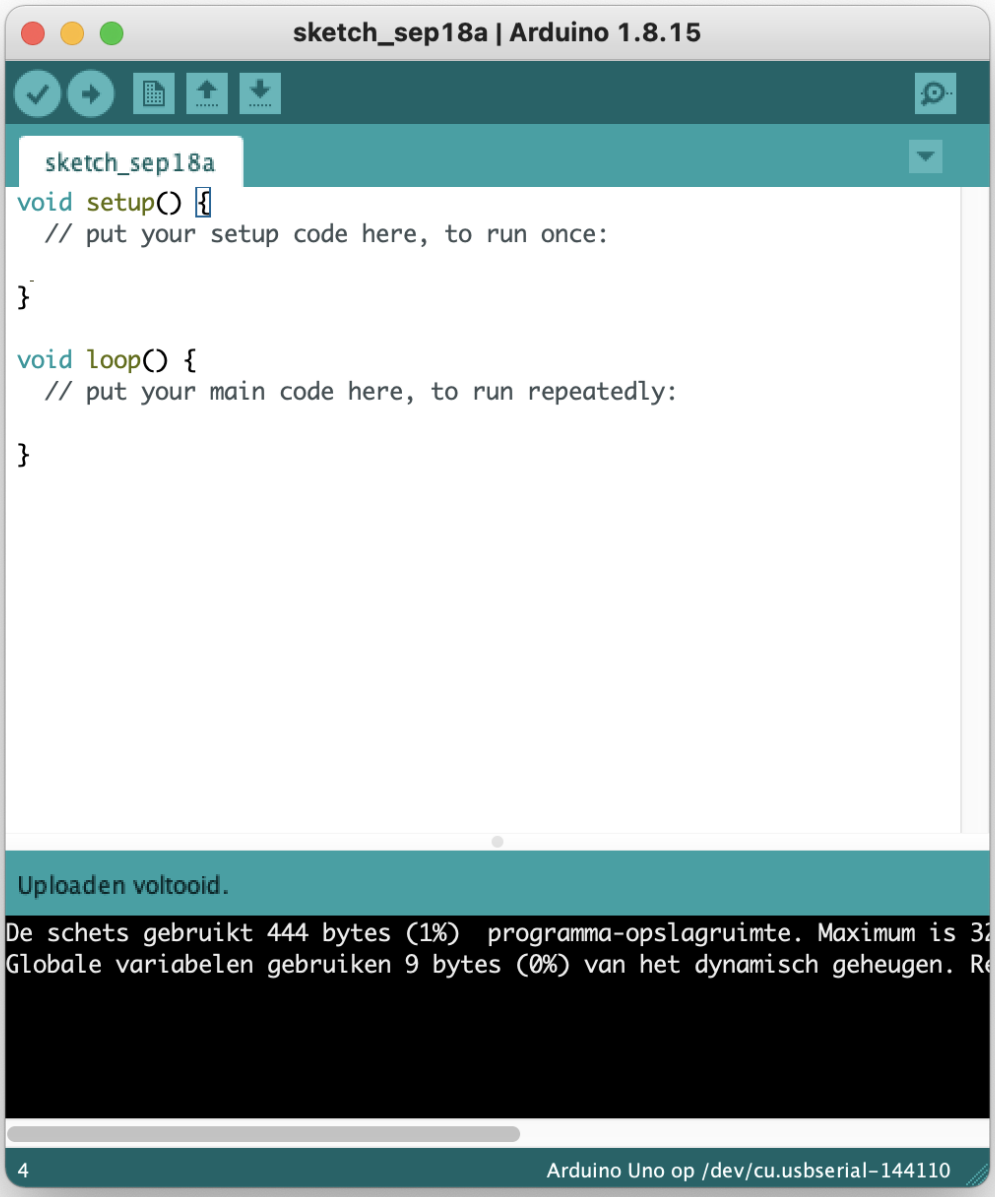
Weet je niet welke poort je moet gebruiken? Kijk dan welke poorten er nu in de lijst staan. Sluit het menu, ontkoppel de microcontroller en kijk opnieuw welke poorten er in de lijst staan. De poort die niet langer in de lijst staat is de poort van de microcontroller.

Nu de juiste poort geselecteerd is kunnen we de verbinding testen. Dit doen we door een vrijwel leeg programma naar de microcontroller te uploaden.

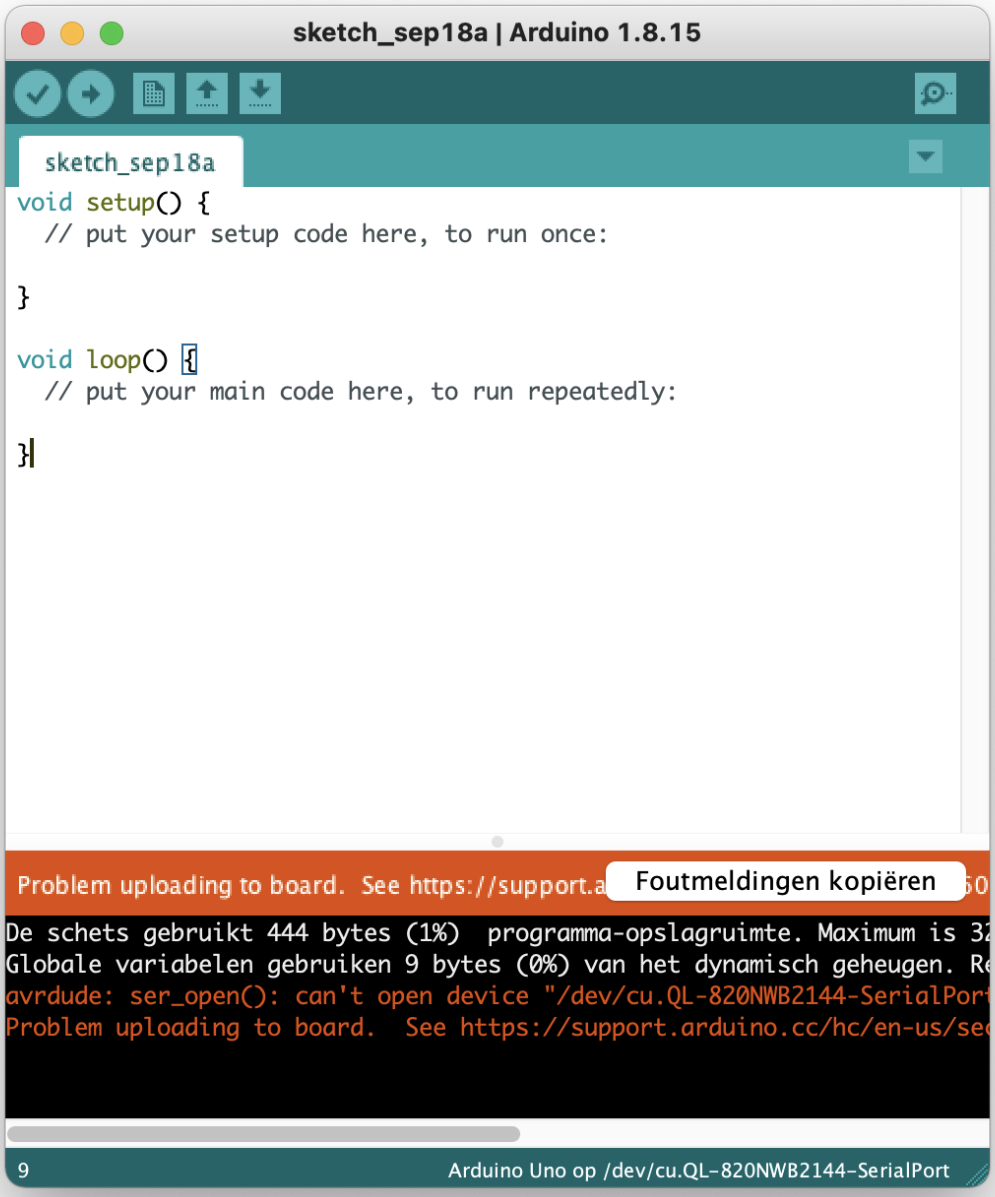
3. Klik op de ***Uploaden*** knop  .



4. In het zwarte vlak onder in het scherm zie je nu een aantal berichten verschijnen. In de groene balk daarboven (statusbalk) zie je de status van het uploaden. Staat er in de groene balk "*Uploaden voltooid*"? Dan is het uploaden geslaagd.



Wordt de statusbalk oranje? Dan is er een fout opgetreden tijdens het uploaden. Doorloop in dit geval de stappen voor het kiezen van het bord (blz. 9) en het aansluiten van de microcontroller (blz. 10) nogmaals om te zien of je daar iets vergeten bent. Upload het programma daarna opnieuw.



Oefening 1: Laat een LED knipperen

De microcontroller heeft een ingebouwd LED lampje. Om gewend te raken aan het programmeren van de microcontroller gaan we deze LED eerst eens 2 keer aan en uit zetten. De ingebouwde LED op de microcontroller ziet eruit als een klein wit vierkantje met een L ernaast. Je kunt hem nu alvast zoeken, maar zodra hij knippert zie je het ook gauw genoeg!

De ingebouwde LED is te bedienen door een hoog of laag signaal op pin 13 te zetten. De interne LED is namelijk direct verbonden aan pin 13.

Benodigdheden

- Arduino
- USB kabel
- Computer

Stappen

1. Neem de onderstaande code over in de Arduino IDE. (Vervang hierbij altijd alle bestaande code, tenzij anders vermeld.)

```
// Dit gedeelte wordt slechts 1 keer uitgevoerd wanneer de microcontroller wordt ingeschakeld
void setup() {
  pinMode(13, OUTPUT);       // We gaan pin 13 gebruiken als output

  digitalWrite(13, HIGH);    // Zet de LED aan door een hoog signaal op pin 13 zetten
  delay(1000);               // Wacht 1 seconde (1000 milliseconden)
  digitalWrite(13, LOW);     // Zet de LED uit door een laag signaal op pin 13 te zetten
  delay(1000);               // Wacht 1 seconde (1000 milliseconden)

  digitalWrite(13, HIGH);    // Zet de LED aan door een hoog signaal op pin 13 zetten
  delay(1000);               // Wacht 1 seconde (1000 milliseconden)
  digitalWrite(13, LOW);     // Zet de LED uit door een laag signaal op pin 13 te zetten
  delay(1000);               // Wacht 1 seconde (1000 milliseconden)
}

void loop() {
}
```

Arduino code oefening 1

ⓘ In de code staan regels achter twee schuine strepen ("//"), dit zijn aantekeningen. Deze regels zijn puur ter verduidelijking en hebben geen invloed op de uitvoer van het programma.

2. Zorg ervoor dat de microcontroller verbonden is (blz. 10) met de computer, dat je het juiste board (blz. 9) en de juiste poort (blz. 10) geselecteerd hebt.

3. Klik op de ***Verifiëren*** knop  . De Arduino IDE controleert nu of de code die je hebt ingevoerd juist is.

   Waarschijnlijk vraagt Arduino IDE je nu om een locatie en een naam te kiezen om de sketch onder op te slaan. Kies hier een naam en locatie waarmee je jouw code weer terug weet te vinden. Iedere keer dat je een ***Verifiëren*** uitvoert wordt de nieuwe versie van je code op deze plek opgeslagen.

   Wanneer alles goed gaat verschijnt de tekst "*Done compiling.*" in de statusbalk.

   Wordt de statusbalk oranje? Dan staat er een fout in de code of je hebt niet het juiste bord geselecteerd. Kijk of je de code juist gekopieerd hebt en controleer of de port en het bord dat je geselecteerd hebt juist zijn.

De code is nu gecontroleerd door de Arduino IDE. Deze stap wordt ook standaard uitgevoerd wanneer je op de ***Uploaden*** knop  klikt. Het is dus niet altijd nodig het programma te verifiëren voordat je het uploadt.

4. Klik op de ***Uploaden*** knop  om het programma te uploaden naar de microcontroller.

Wanneer alles goed is gegaan gaat de LED op de microcontroller nu twee keer aan en uit. Goed gedaan!

Druk op de reset knop (zie afbeelding 0.1 (blz. 3) van de microcontroller. Dit zorgt ervoor dat de microcontroller opnieuw opstart en dus de code nogmaals uitvoert.

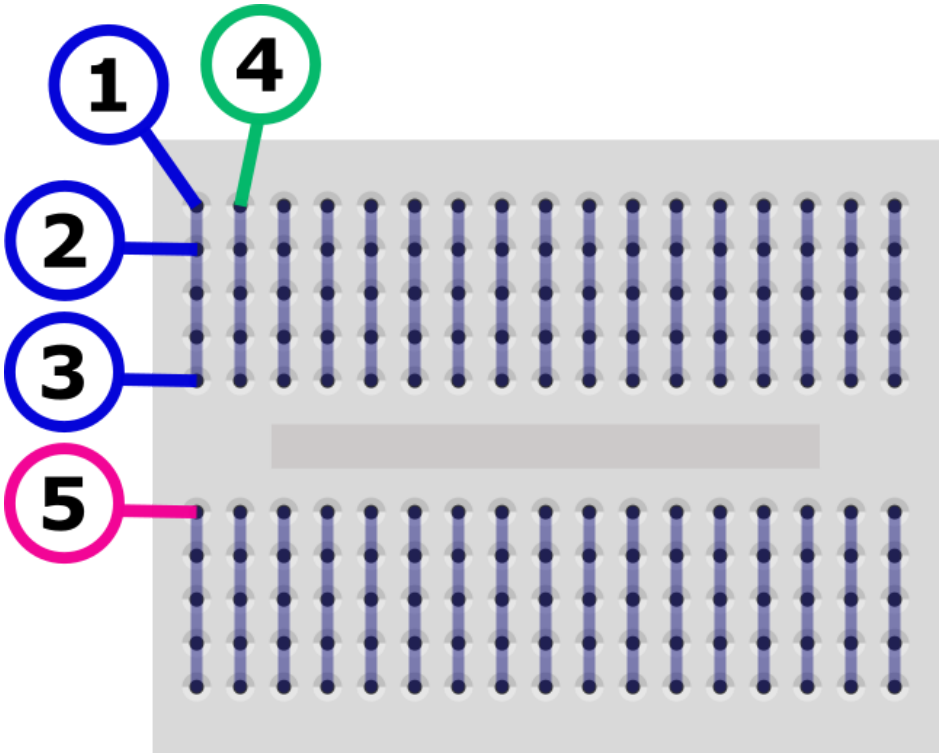
Knippert het lampje niet? Bekijk dan de tips onder Problemen oplossen (blz. 40).

Het breadboard

In oefening 1 (blz. 14) hebben we een ingebouwde led op de microcontroller laten knipperen. Het mooie aan microcontrollers is dat je ook externe onderdelen aan kunt sluiten. Om gemakkelijk meerdere onderdelen aan te sluiten op de microcontroller gebruiken we een breadboard.

Om de onderdelen met elkaar te verbinden kun je de draadjes en pootjes van de onderdelen in de gaatjes op het breadboard steken. Binnenin het breadboard zijn de gaatjes op een speciale manier met elkaar verbonden, dit is te zien in het plaatje.

Dit betekent dat gaatje 1 verbonden is met gaatje 2 en 3, maar niét met gaatje 4 en ook niet met gaatje 5.



Afbeelding 0.2

Oefening 2: Een buzzer aansluiten

In deze oefening gaan we het breadboard gebruiken om een buzzer aan te sluiten en te laten piepen.

Benodigdheden

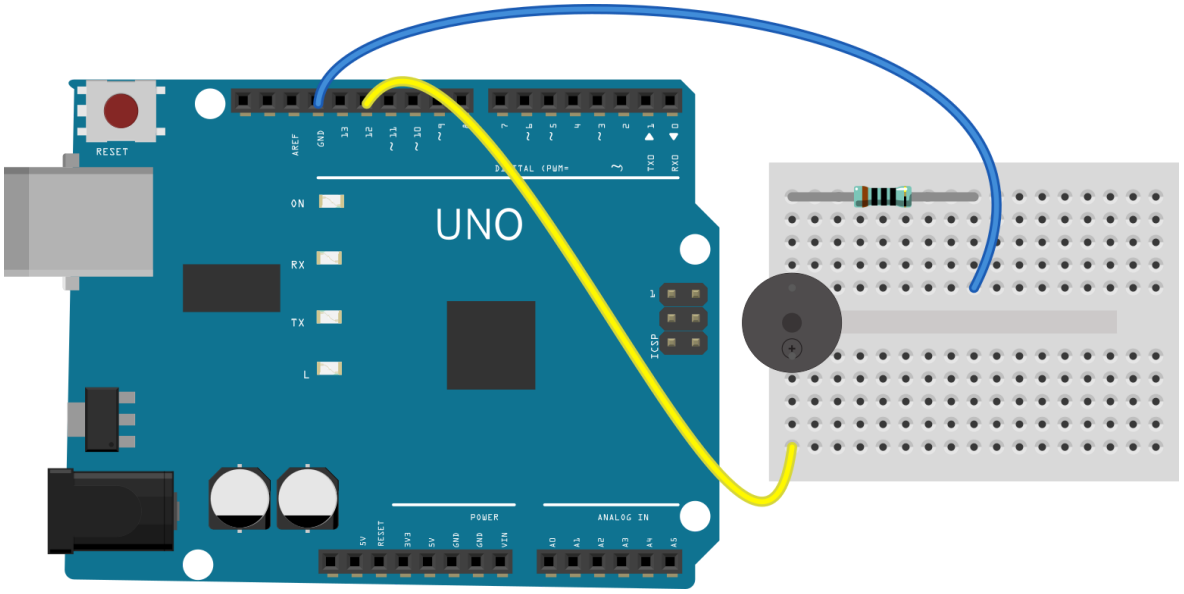
- Microcontroller
- USB kabel
- Computer
- Breadboard
- Jumper kabeltjes (m-m)
- Buzzer
- Weerstand

Verbindingen

Microcontroller	*Onderdeel*	*Via*
GND	Buzzer korte pootje	Weerstand en blauwe jumper kabel
12	Buzzer lange pootje	Gele jumper kabel

Stappen

5. Maak de schakeling uit afbeelding 2.1. Gebruik de jumper kabels om in de microcontroller en het breadboard te steken.



Afbeelding 2.1

6. Neem de onderstaande code over in de Arduino IDE.

```
// Dit gedeelte wordt slechts 1 keer uitgevoerd wanneer de microcontroller wordt ingeschakeld
void setup() {
  pinMode(12, OUTPUT);      // We gaan pin 12 gebruiken als output

  digitalWrite(12, HIGH);    // Zet de buzzer aan door een laag signaal op 12 zetten
  delay(1000);               // Wacht 1 seconde (1000 milliseconden)
  digitalWrite(12, LOW);    // Zet de buzzer uit door een hoog signaal op 12 te zetten
  delay(1000);               // Wacht 1 seconde (1000 milliseconden)

  digitalWrite(12, HIGH);    // Zet de buzzer aan door een laag signaal op 12 te zetten
  delay(1000);               // Wacht 1 seconde (1000 milliseconden)
  digitalWrite(12, LOW);    // Zet de buzzer uit door een hoog signaal op 12 te zetten
  delay(1000);               // Wacht 1 seconde (1000 milliseconden)
}

// Dit gedeelte wordt herhaald tot de microcontroller wordt uitgezet
void loop() {
}
```

Arduino code oefening 2

We gebruiken hier bijna dezelfde code als in oefening 1 (blz. 14). We hebben de pin die we willen aansturen veranderd van 13 naar 12. We willen namelijk niet meer de ingebouwde LED op pin 13 gebruiken maar de buzzer die wij hebben aangesloten op pin 12.

7. Zorg ervoor dat de microcontroller verbonden is (blz. 10) met de computer, dat je het juiste board (blz. 9) en de juiste poort (blz. 10) geselecteerd hebt.

8. Klik op de ***Uploaden*** knop  om het programma te uploaden naar de microcontroller.

Wanneer alles goed is gegaan piept de buzzer die je op het breadboard hebt aangesloten twee keer.

Druk op de reset knop (zie afbeelding 0.1 (blz. 3) van de microcontroller om de code nogmaals uit te voeren.

Piept de buzzer niet? Bekijk dan de tips onder Problemen oplossen (blz. 40).

ⓘ De kleur van de kabeltjes maakt niet uit voor de werking van de schakeling. Deze kleuren gebruiken we puur ter verduidelijking van de schakeling. Zo wordt er voor 5V of 3V3 vaak rood gebruikt, en voor de GND vaak zwart, bruin of blauw. De andere kleuren worden gebruikt voor de signaal pins.

Oefening 3: functies

In oefening 2 (blz. 17) hebben we de buzzer aan en weer uitgezet met het volgende stukje code:

```
  digitalWrite(12, HIGH);   // Zet de buzzer aan door een hoog signaal op 12 te zetten
  delay(1000);              // Wacht 1 seconde (1000 milliseconden)
  digitalWrite(12, LOW);    // Zet de buzzer uit door een laag signaal op 12 te zetten
  delay(1000);              // Wacht 1 seconde (1000 milliseconden)
```

Dit stuk code herhalen we twee keer. Dat kan eenvoudiger. Hiervoor gaan we een functie gebruiken. Een functie is een verzameling van instructies welke samen een taak uitvoeren.

Wanneer we de code om de buzzer aan en dan weer uit te zetten omzetten naar een functie dan ziet dat er als volgt uit:

```
void zetBuzzerAanEnUit() {
  digitalWrite(12, LOW);    // Zet de LED aan door een laag signaal op D4 zetten
  delay(1000);              // Wacht 1 seconde (1000 milliseconden)
  digitalWrite(12, HIGH);   // Zet de LED uit door een hoog signaal op D4 te zetten
  delay(1000);              // Wacht 1 seconde (1000 milliseconden)
}
```

In ons programma kunnen we nu de functie `zetBuzzerAanEnUit` aanroepen. De inhoud van de functie wordt dan op die plek uitgevoerd. Het programma ziet er dan als volgt uit:

```
// Deze functie zet de buzzer aan en na 1 seconde weer uit
void zetBuzzerAanEnUit() {
  digitalWrite(12, HIGH);   // Zet de buzzer aan door een hoog signaal op 12 te zetten
  delay(1000);              // Wacht 1 seconde (1000 milliseconden)
  digitalWrite(12, LOW);    // Zet de buzzer uit door een laag signaal op 12 te zetten
  delay(1000);              // Wacht 1 seconde (1000 milliseconden)
}

// Dit gedeelte wordt slechts 1 keer uitgevoerd wanneer de Arduino wordt ingeschakeld
void setup() {
  pinMode(12, OUTPUT);      // We gaan pin 12 gebruiken als output
  zetBuzzerAanEnUit();         // Voer de inhoud van de functie zetBuzzerAanEnUit uit
  zetBuzzerAanEnUit();         // Voer de inhoud van de functie zetBuzzerAanEnUit uit
}

// Dit gedeelte wordt herhaald tot de microcontroller wordt uitgezet
void loop() {
}
```

Arduino code oefening 3

Benodigdheden

- Microcontroller
- USB kabel
- Computer
- Breadboard
- Jumper kabeltjes (m-m)
- Buzzer
- Weerstand

Stappen

1. Maak de schakeling die we in oefening 2 (blz. 17) gebruikt hebben.

2. Neem de code uit het blok Arduino code oefening 3 (blz. 19) over in de Arduino IDE.

3. Zorg ervoor dat de microcontroller verbonden is (blz. 10) met de computer, dat je het juiste board (blz. 9) en de juiste poort (blz. 10) geselecteerd hebt.

4. Klik op de ***Uploaden*** knop  om het programma te uploaden naar de microcontroller.

Wanneer alles goed is gegaan piept de buzzer die je op het breadboard hebt aangesloten net als in oefening 2 (blz. 17).

Piept de buzzer niet? Bekijk dan de tips onder Problemen oplossen (blz. 40).

Oefening 4: de "loop" gebruiken

In oefening 3 (blz. 19) hebben we de functie **zetBuzzerAanEnUit** twee keer aangeroepen om de buzzer te laten piepen. In deze opdracht gaan we de buzzer laten piepen zolang de microcontroller aan staat. We doen dit door de loop (Engels voor "lus") functie te gebruiken.

In al de vorige voorbeelden zie je onderaan de volgende code staan:

```
void loop() {
}
```

Wanneer de microcontroller wordt aangezet voert hij eerst de code uit binnen de setup functie. Daarna wordt de code die binnen de loop functie staat herhaald zolang de microcontroller aan staat.

> ⓘ Misschien is het je opgevallen dat **setup()** en **loop()** er hetzelfde uitzien als de functie **zetLedAanEnUit()** die we net hebben gemaakt. Goed gezien, **setup()** en **loop()** zijn functies waarvan jij de definitie mag schrijven, maar waarbij de aanroep automatisch wordt geregeld door de microcontroller.

Door de aanroep van onze functie `zetBuzzerAanEnUit` in de loop te zetten wordt deze herhaald tot de microcontroller uit gaat.

Benodigdheden

- Microcontroller
- USB kabel
- Computer
- Breadboard
- Jumper kabeltjes (m-m)
- Buzzer
- Weerstand

Verbindingen

Microcontroller	*Onderdeel*	*Via*
GND	Buzzer korte pootje	Weerstand en zwarte jumper kabel
12	Buzzer lange pootje	Gele jumper kabel

Stappen

1. Maak de schakeling die we in oefening 2 (blz. 17) gebruikt hebben.

2. Neem de onderstaande code over in de Arduino IDE.

```
// Deze functie zet de LED aan en na 1 seconde weer uit
void zetBuzzerAanEnUit() {
  digitalWrite(12, HIGH);   // Zet de buzzer aan door een hoog signaal op 12 te zetten
  delay(1000);              // Wacht 1 seconde (1000 milliseconden)
  digitalWrite(12, LOW);    // Zet de buzzer uit door een laag signaal op 12 te zetten
  delay(1000);              // Wacht 1 seconde (1000 milliseconden)
}

// Dit gedeelte wordt slechts 1 keer uitgevoerd wanneer de Arduino wordt ingeschakeld
void setup() {
  pinMode(12, OUTPUT);      // We gaan pin 12 gebruiken als output
}

// Dit gedeelte wordt herhaald tot de microcontroller wordt uitgezet
void loop() {
  zetBuzzerAanEnUit();         // Voer de inhoud van de functie zetBuzzerAanEnUit uit
}
```

Arduino code oefening 4

De code waarmee we aangeven dat we pin 12 als output gaan gebruiken blijft in de setup functie staan. Dit hoeven we namelijk slechts één keer aan te geven.

3. Zorg ervoor dat de microcontroller verbonden is (blz. 10) met de computer, dat je het juiste board (blz. 9) en de juiste poort (blz. 10) geselecteerd hebt.

4. Klik op de ***Uploaden*** knop  om het programma te uploaden naar de microcontroller.

Wanneer alles goed is gegaan piept de buzzer die je op het breadboard hebt aangesloten. In tegenstelling tot de vorige oefeningen blijft de buzzer dit keer aan en uit gaan tot je de stroom van de microcontroller afhaalt.

Piept de buzzer niet? Bekijk dan de tips onder Problemen oplossen (blz. 40).

Oefening 5: variabelen

In deze oefening gaan we onze buzzer sneller laten piepen. Dit doen we door de instructie die de microcontroller laat wachten aan te passen. Deze code ziet er op dit moment als volgt uit:

```
delay(1000);              // Wacht 1 seconde (1000 milliseconden)
```

De door de functie delay aan te roepen met de waarde 1000 laten we de microcontroller 1000 milliseconden wachten. Door de waarde die we aan de functie delay meegeven aan te passen kunnen we de microcontroller korter of juist langer laten wachten.

ⓘ Net zoals de functie zetBuzzerAanEnUit is delay ook een functie. De inhoud (definitie) van de delay functie hoeven we alleen niet zelf te schrijven maar is ingebouwd in de Arduino programmeertaal. Dit zorgt ervoor dat we makkelijk en snel code kunnen schrijven zonder dat we veel gebruikte functionaliteit zelf opnieuw hoeven te schrijven.

We zijn inmiddels al 4 functies tegengekomen die op verschillende manieren worden gedefinieerd en aangeroepen. Mocht je even niet meer precies weten hoe het zat dan geeft onderstaande tabel misschien wat duidelijkheid.

functie	definitie	aanroep
setup en loop	Geschreven door jou	Automatisch aangeroepen door microcontroller
delay	Geschreven door Arduino ontwikkelaars	Aangeroepen door jou
zetBuzzerAanEnUit	Geschreven door jou	Aangeroepen door jou

Op de plekken in onze functie zetBuzzerAanEnUit waar we delay aanroepen kunnen we de waarde 1000 vervangen door een lager getal om de buzzer sneller te laten piepen.

We gaan hier echter geen harde waardes invullen maar gebruik maken van een variabele. Een variabele is een plek waar je informatie kunt opslaan en die je later weer kunt uitlezen.

Je maakt een variabele op de volgende manier:

```
int aantalMillisecondenWachten = 1000;
```

Net als bij een functie, noemen we het maken van een variabele *declareren*. De declaratie van een variabele kent drie onderdelen:

Het datatype van de variabele
In dit voorbeeld is het datatype `int` wat staat voor integer oftewel "geheel getal". Aan de hand van het datatype bepaal je wat voor gegevens er in de variabele opgeslagen kunnen worden. In dit geval kunnen we hele getallen in onze variabele stoppen. Veel gebruikte datatypes zijn:

Datatype	*Voor het opslaan van*	*Voorbeeldwaarde*
`int`	Gehele getallen	12345
`boolean`	Iets dat slechts twee waardes kan hebben	`true` of `false`
`char`	Een teken (character)	'A' of '!'
`float`	Getallen met een komma *(Deze programmeertaal gebruikt de Engelse notatie voor getallen. We gebruiken daarom een punt in plaats van een komma)*	123.45

Naam van de variabele
In dit voorbeeld is de naam van de variabele `aantalMillisecondenWachten`. Je mag deze naam zelf verzinnen. Het is verstandig om een naam te kiezen die duidelijk omschrijft welke waarde er in de variabele zit.

De waarde van de variabele
In dit voorbeeld is de waarde van de variabele 1000. Het type waarde dat je kunt toewijzen is afhankelijk van het datatype van de variabele. In ons voorbeeld gebruiken we een `int` waardoor we alleen hele getallen kunnen toewijzen (zoals 1000). Hadden we bijvoorbeeld het datatype `char` gekozen dan hadden we een teken toe kunnen wijzen. De code had er dan zo uit kunnen zien:

```
char eenVoorbeeldTeken = 'c';
```

Benodigdheden

- Microcontroller
- USB kabel
- Computer
- Breadboard
- Jumper kabeltjes (m-m)
- Buzzer
- Weerstand

Verbindingen

Microcontroller	*Onderdeel*	*Via*
GND	Buzzer korte pootje	Weerstand en zwarte jumper kabel
12	Buzzer lange pootje	Gele jumper kabel

Stappen

1. Maak de schakeling die we in oefening 2 (blz. 17) gebruikt hebben.

2. Neem de onderstaande code over in de Arduino IDE.

```
// Deze functie zet de buzzer aan en na 0,5 seconde weer uit
void zetBuzzerAanEnUit() {
  int aantalMillisecondenWachten = 500;  // Variabele die bepaalt hoe lang we wachten
  digitalWrite(12, HIGH);                // Zet de buzzer aan met een hoog signaal op 12
  delay(aantalMillisecondenWachten);     // Wacht een halve seconde (500 milliseconden)
  digitalWrite(12, LOW);                 // Zet de LED uit met een laag signaal op 12
  delay(aantalMillisecondenWachten);     // Wacht een halve seconde (500 milliseconden)
}

// Dit gedeelte wordt slechts 1 keer uitgevoerd wanneer de Arduino wordt ingeschakeld
void setup() {
  pinMode(12, OUTPUT);      // We gaan pin 12 gebruiken als output
}

// Dit gedeelte wordt herhaald tot de microcontroller wordt uitgezet
void loop() {
  zetBuzzerAanEnUit();         // Roep de functie zetLedAanEnUit aan
}
```

Arduino code oefening 5

3. Zorg ervoor dat de microcontroller verbonden is (blz. 10) met de computer, dat je het juiste board (blz. 9) en de juiste poort (blz. 10) geselecteerd hebt.

4. Klik op de ***Uploaden*** knop  om het programma te uploaden naar de microcontroller.

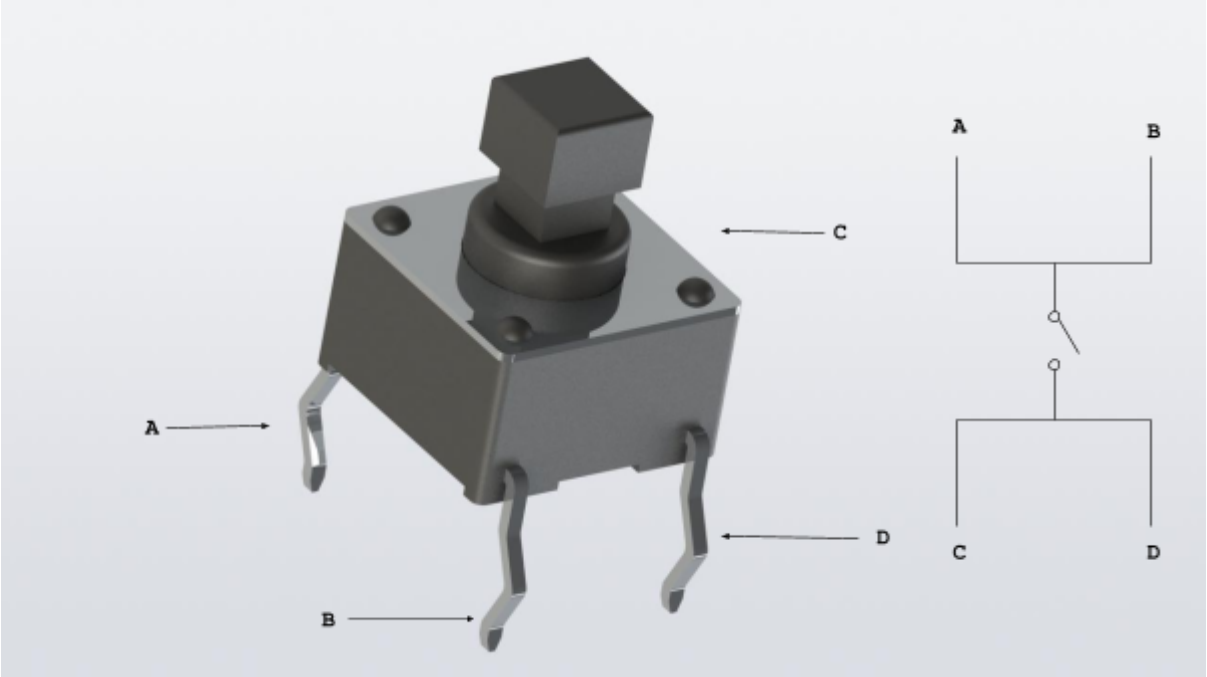
Wanneer alles goed is gegaan piept de buzzer nu twee keer zo snel aangezien we de tijd die de microcontroller wacht gehalveerd hebben.

Je kunt nu de snelheid waarmee de buzzer piept makkelijk aanpassen door de variabele **aantalMillisecondenWachten** aan te passen en de code opnieuw te uploaden.

Piept de buzzer niet? Bekijk dan de tips onder Problemen oplossen (blz. 40).

Oefening 6: Een knop toevoegen

We hebben nu geleerd hoe we de buzzer kunnen laten piepen op een manier die we van tevoren vastleggen in de code. Met een microcontroller kun je naast output genereren (zoals de buzzer laten piepen) ook input verwerken. In dit geval gaan we een knop gebruiken om de buzzer mee aan te sturen.



Afbeelding 6.1

Zoals te zien is in afbeelding 6.1, zijn de pootjes A en B intern met elkaar verbonden. Hetzelfde geldt voor de pootjes C en D. Tussen deze twee kanten van het knopje in, zit een schakelaar. Deze kun je beschouwen als een brug. Wanneer de schakelaar dicht is, kan de stroom er doorheen lopen. Wanneer de schakelaar open is, niet.

Waar we voor het genereren van output de `digitalWrite` functie gebruikten, gebruiken we voor het uitlezen van input de `digitalRead` functie.

```
digitalRead(4);
```

Omdat we de buzzer aan willen zetten wanneer de knop ingedrukt is, maar uit willen zetten wanneer de knop niet ingedrukt is, moeten we een manier hebben om hier in de code onderscheid tussen te maken. Dit gaan we doen met een `if...else statement`.

```
if(conditie){
  // Deze code wordt uitgevoerd wanneer wordt voldaan aan de conditie
} else {
  // Deze code wordt uitgevoerd wanneer niet wordt voldaan aan de conditie
```

```
}
```

Een **if...else statement** is een manier om de microcontroller een stuk code uit te laten voeren wanneer voldaan wordt aan een conditie, en een ander stuk wanneer hier niet aan voldaan wordt.
De **conditie** moet een uitdrukking zijn waarvan de uitkomst van het datatype **boolean** is. (Kijk nog eens naar Het datatype van een variabele voor de uitleg over datatypes.)

In ons geval willen we de status van de knop in de conditie van het if-statement zetten. Als de knop ingedrukt is, is de waarde van de pin **HIGH**. Dit is dus wat we willen controleren.
In de Arduino programmeertaal kunnen we hiervoor de **==** operator gebruiken. Deze operator vergelijkt de waardes van de uitdrukkingen aan beide kanten. Als deze hetzelfde zijn is de uitkomst van de operatie **true**, als ze niet gelijk zijn is de uitkomst van de operatie **false**. In de Arduino programmeertaal zijn een heleboel operatoren bekend, in de onderstaande tabel staan een aantal voorbeelden.

Operator	*Voorbeeld uitdrukking*	*Voorbeeld uitkomst*
+	1 + 1	2
-	2 - 1	1
/	10 / 2	5
% (de rest-operator)	10 % 2 11 % 2 13 % 2	0 1 3
==	1 == 2 2 == 2 false == false	false true true
>	1 > 2 2 > 1 1 > 1	false true false

Nu we begrijpen hoe de == operator werkt, kunnen we deze gaan toepassen. We gaan straks de knop aansluiten op pin 4, de conditie die we tussen de haakjes van het if-statement gaan zetten ziet er dan zo uit:

```
digitalRead(4) == HIGH
```

Benodigdheden

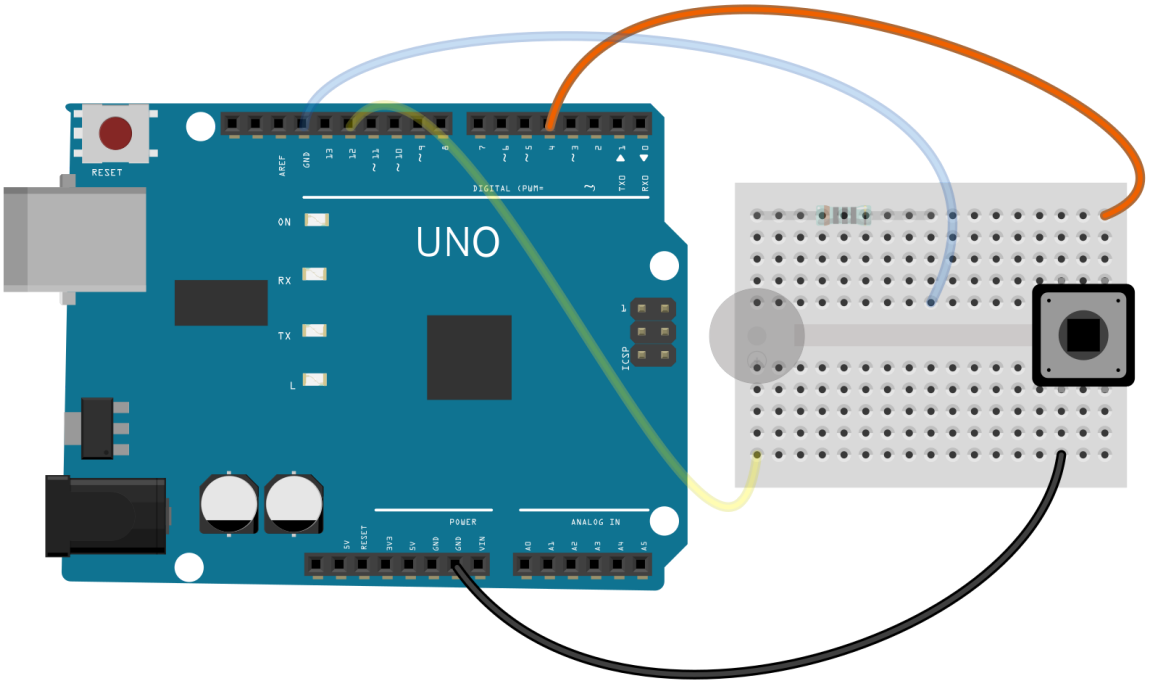
- Schakeling uit oefening 2
- USB kabel
- Computer
- Jumper kabeltjes (m-m)
- Weerstand
- Drukknop met kapje

Verbindingen

Microcontroller	*Onderdeel*	*Via*
GND	Button pin A of B	Zwarte jumper kabel
4	Button pin C of D	Oranje jumper kabel

Stappen

1. Maak de schakeling uit afbeelding 6.2. Gebruik de jumper kabels om in de microcontroller en het breadboard te steken.



Afbeelding 6.2

2. Klik het kapje op het knopje, zo kun je hem makkelijker indrukken.
3. Neem de onderstaande code over in de Arduino IDE.

```
// Deze functie zet de buzzer aan
void zetBuzzerAan() {
  digitalWrite(12, HIGH);       // Zet de buzzer aan met een hoog signaal op 12
}
```

```
// Deze functie zet de buzzer uit
void zetBuzzerUit() {
  digitalWrite(12, LOW);        // Zet de buzzer aan met een laag signaal op 12
}

// Dit gedeelte wordt slechts 1 keer uitgevoerd wanneer de Arduino wordt ingeschakeld
void setup() {
  pinMode(12, OUTPUT);          // We gaan pin 12 gebruiken als output
  pinMode(4, INPUT_PULLUP);     // We gaan pin 4 gebruiken als input
}


// Dit gedeelte wordt herhaald tot de microcontroller wordt uitgezet
void loop() {
  if(digitalRead(4) == LOW) {   // Als het signaal op pin 4 LOW is:
    zetBuzzerAan();             // Roep de functie zetBuzzerAan() aan
  } else {                      // Als het signaal op pin 4 niet LOW is:
    zetBuzzerUit();             // Roep de functie zetBuzzerUit() aan
  }
}
```

Arduino code oefening 6

4. Zorg ervoor dat de microcontroller verbonden is (blz. 10) met de computer, dat je het juiste board (blz. 9) en de juiste poort (blz. 10) geselecteerd hebt.
5. Klik op de ***Uploaden*** knop  om het programma te uploaden naar de microcontroller.

Wanneer alles goed is gegaan piept de buzzer nu alleen wanneer je op de knop drukt.

Gaat er nog iets mis? Bekijk dan de tips onder Problemen oplossen (blz. 40).

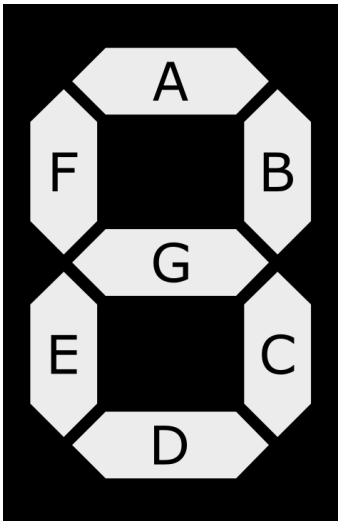
ⓘ De pinMode van pin 4 zetten we op `INPUT_PULLUP`. Dit betekent dat we pin 4 gebruiken als input en dat deze met behulp van een grote weerstand die ingebouwd is in de microcontroller “omhoog getrokken” wordt. Dit zorgt ervoor dat wanneer de knop niet ingedrukt is, er op pin 4 een HIGH signaal staat. Wanneer de knop is ingedrukt, wordt deze naar de GND getrokken en staat er dus een LOW signaal op.

Oefening 7: Zeven segment klok

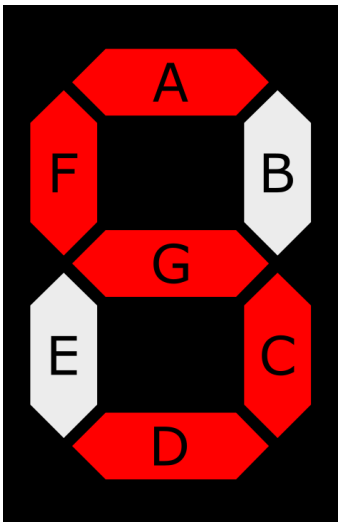
In deze oefening gaan we de display aansturen waar de resterende tijd op af te lezen is. We gebruiken hiervoor een 7 segment klok module (beter bekend onder de Engelse term 7 segment clock).

Op de module die wij hier gebruiken kunnen vier cijfers worden weergegeven. Ieder cijfer bestaat uit zeven streepjes waar een LED lampje in zit. Door de LED's van bepaalde streepjes aan te zetten en die van andere streepjes uit te zetten kunnen we een cijfer weergeven.

In afbeelding 8.1 zie je een zeven segment module met één cijfer. Door de LED's A,F, G, C en D aan te zetten kunnen we bijvoorbeeld het cijfer 5 vormen.



Afbeelding 7.1



Afbeelding 7.2

Aangezien ieder streepje op de zeven segment display een LED is, zouden we iedere LED individueel met de microcontroller kunnen aansturen. We hebben dan alleen wel heel veel pins nodig. In ons geval achtentwintig (7 per cijfer keer 4 cijfers).

De microcontroller die wij gebruiken heeft maar 20 pins die we kunnen aansturen, we komen er dus acht tekort. Gelukkig is hier een oplossing voor.

Wanneer je naar de achterkant van de zeven segment klok kijkt zie je een zwarte chip zitten. Deze chip is verbonden met al de LED's en zorgt ervoor dat we deze kunnen aansturen met slechts 2 pins (en een 5V en GND pin).

Om de chip van de zeven segment klok aan te sturen gebruiken we in de Arduino IDE een library (bibliotheek). Een library is een verzameling van code die al voor ons geschreven is. Dit maakt het een stuk makkelijker om verschillende componenten aan te sturen.

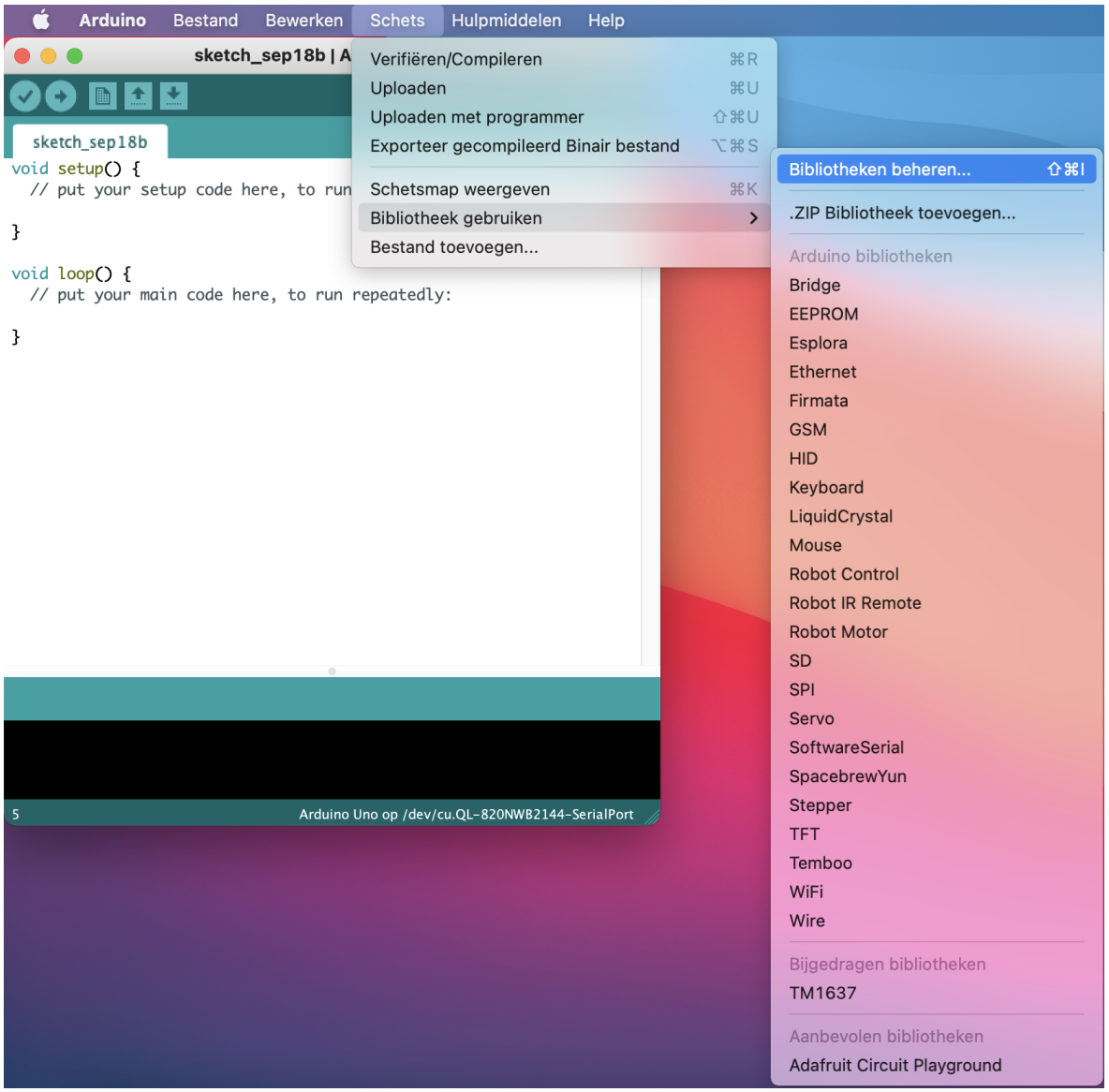
Voor onze zeven segment klok gebruiken we de Grove 4-Digit Display library.

Library installeren

Om de Grove 4-Digit Display library te gebruiken moeten we deze eerst installeren in de Arduino IDE.

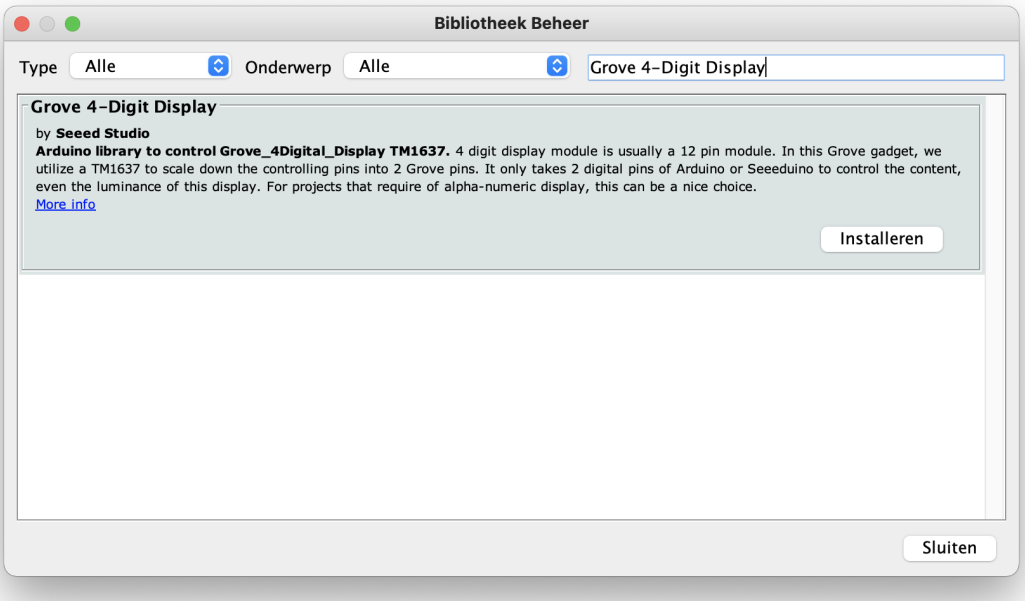
Stappen

1. Klik in de menubalk op ***Schets*** en vervolgens op ***Bibliotheek gebruiken*** en selecteer ***Bibliotheken beheren***.



Afbeelding 7.3

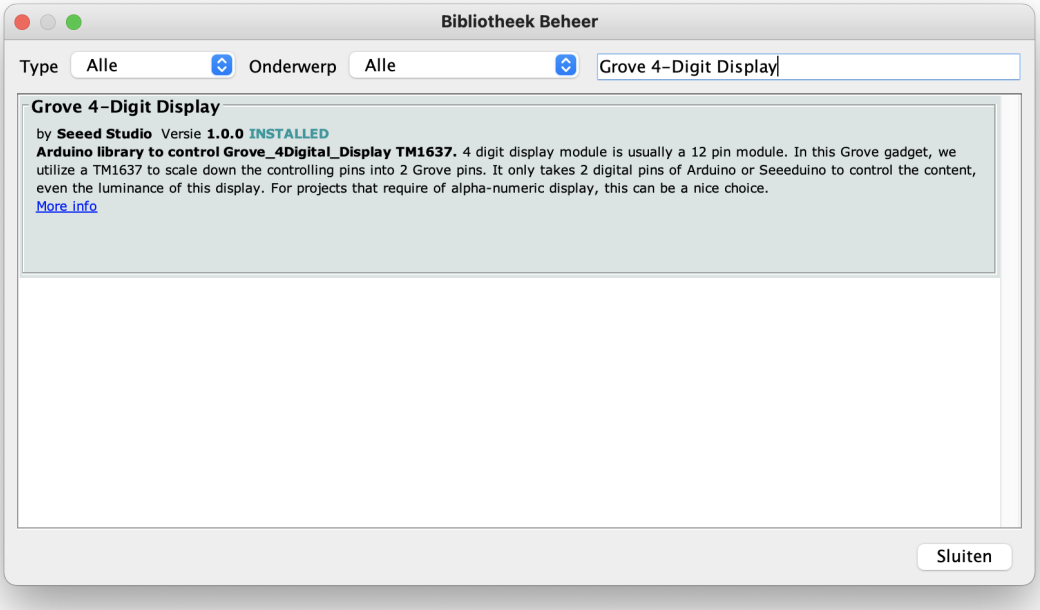
2. Er opent nu een nieuw scherm genaamd Bibliotheek Beheer. In het veld rechts boven in het scherm, waar nu "*Filter je zoekresultaten...*" staat. kan je een zoekterm invullen. Vul hier de naam van de library in die wij willen gebruiken (Grove 4-Digit Display).



Afbeelding 7.4

3. Klik nu op de knop **"Installeren"** om de library te installeren.

4. Na een aantal seconden is de library geïnstalleerd. Een geïnstalleerde library herken je aan de groene tekst "INSTALLED" naast het versienummer van de library. Je kunt de Library Manager nu sluiten door op de knop **"Sluiten"** te klikken.



Afbeelding 7.5

Library gebruiken

Nu de Grove 4-Digit Display library geïnstalleerd is kunnen we deze gebruiken in onze code. We beginnen met eenvoudige code welke vier cijfers op het display weergeeft.

Benodigdheden

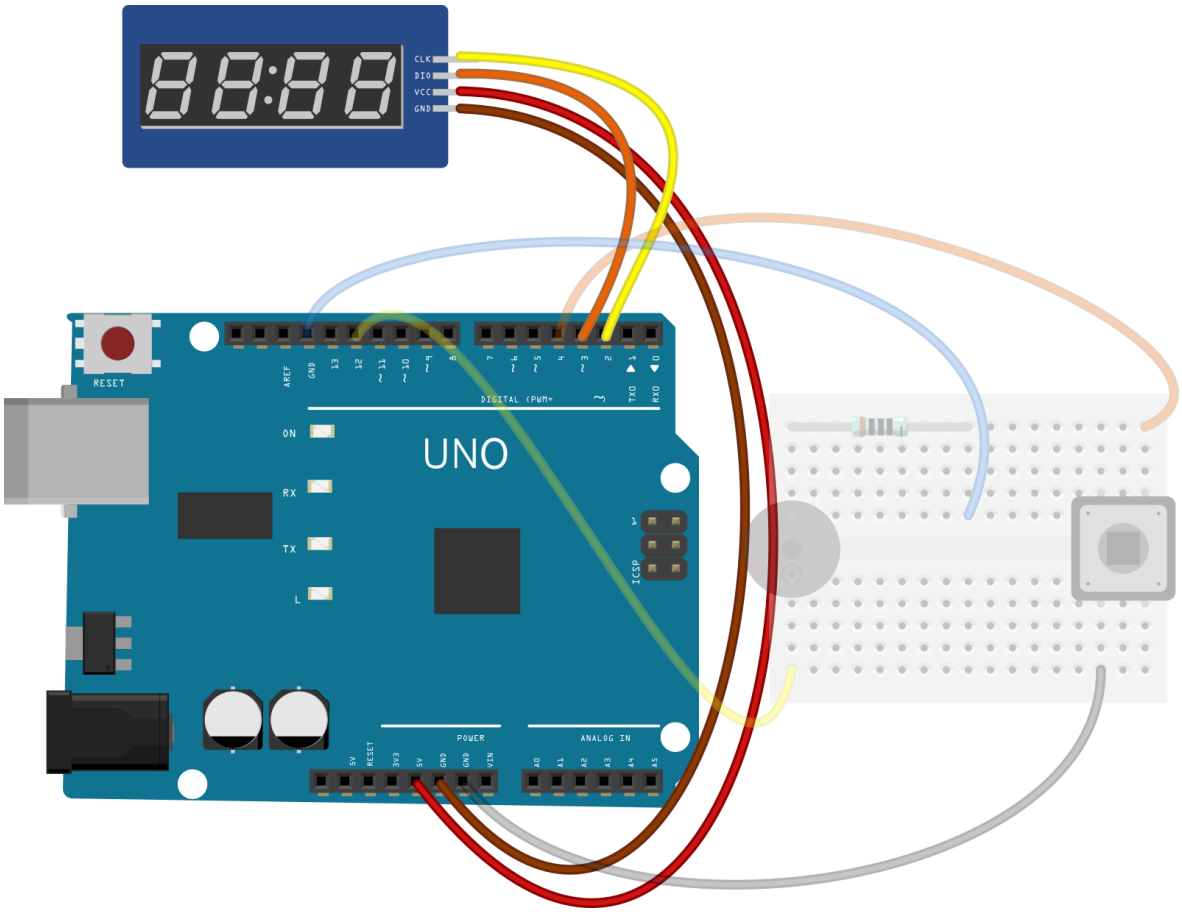
- Schakeling uit oefening 6
- Jumper kabeltjes (m-v)
- 7 segment klok
- USB kabel
- Computer

Verbindingen

Microcontroller	*Onderdeel*	*Via*
GND	7 Segment Klok GND	Bruine jumper kabel
5V	7 Segment Klok VCC	Rode jumper kabel
2	7 Segment Klok CLK	Gele jumper kabel
3	7 Segment Klok DIO	Oranje jumper kabel

Stappen

1. Maak de schakeling uit afbeelding 7.6. We bouwen hierbij voort op de schakeling uit oefening 6 (blz. 26).



Afbeelding 7.6

2. Neem de onderstaande code over in de Arduino IDE.

```
#include <TM1637.h>

TM1637 klokDisplay(2, 3);    // Maak een variabele klokDispay waarmee we de display bedienen

// Dit gedeelte wordt slechts 1 keer uitgevoerd wanneer de Arduino wordt ingeschakeld
void setup() {

  klokDisplay.init();               // Start de display
  klokDisplay.set(3);               // Stel de helderheid in op 3 (0-7)

  klokDisplay.point(1);             // Zet de dubbele punt aan
  klokDisplay.display(0, 1);        // Toon het cijfer 1 op op positie 0
  klokDisplay.display(1, 2);        // Toon het cijfer 2 op op positie 1
  klokDisplay.display(2, 3);        // Toon het cijfer 3 op op positie 2
  klokDisplay.display(3, 4);        // Toon het cijfer 4 op op positie 3
}

// Dit gedeelte wordt herhaald tot de microcontroller wordt uitgezet
void loop() {
}
```

Arduino code oefening 7

De code die de Grove 4-Digit Display library gebruikt staat in het bestand TM1637.h. Om de Grove 4-Digit Display library in onze code te

gebruiken moeten we dit bestand importeren. We doen dit op regel 1 van onze code op de volgende manier:

```
#include <TM1637.h>
```

Nu de library geïmporteerd is kunnen we deze gaan gebruiken. Op regel 3 maken we een nieuwe variabele met de naam klokDisplay welke we kunnen gebruiken om de display te bedienen.

```
TM1637 klokDisplay(2, 3);
```

Bij het maken van de variabele klokDisplay geven we twee argumenten mee, een 2 en een 3. Dit zijn de digitale pins van de microcontroller waarmee de display verbonden is.

ⓘ Bij deze oefening kan je de code om de library te importeren gewoon kopiëren en plakken.

Een andere manier om een library in je code te gebruiken is door in de menubalk te klikken op ***Schets*** en vervolgens op ***Bibliotheek gebruiken*** selecteer daarna de library die je in de code wilt gebruiken.

De Arduino IDE zal dan zelf de nodige code toevoegen.

3. Zorg ervoor dat de microcontroller verbonden is (blz. 10) met de computer, dat je het juiste board (blz. 9) en de juiste poort (blz. 10) geselecteerd hebt.
4. Klik op de ***Uploaden*** knop  om het programma te uploaden naar de microcontroller.

Wanneer alles goed is gegaan zie je nu de getallen 1 2 3 en 4 op de display verschijnen. Ook moeten de twee punten in het midden van display oplichten.

Gaat er nog iets mis? Bekijk dan de tips onder Problemen oplossen (blz. 40).

Oefening 8: Countdown Timer Bouwen

Het is nu tijd om alles dat we in de voorgaande oefeningen geleerd hebben samen te voegen en de Countdown Timer te bouwen.

De bedoeling is dat we een timer bouwen die begint met aftellen wanneer we op de knop drukken. Wanneer de timer op 0 staat klinkt er een geluid zodat je weet dat de timer voltooid is.

We gebruiken hier de schakeling uit oefening 7. Die bevat al de onderdelen die nodig zijn voor de Countdown Timer.

Benodigdheden:

- Schakeling uit oefening 7
- USB kabel
- Computer

Stappen

1. Neem de onderstaande code over in de Arduino IDE.

```
#include <TM1637.h>          // Importeer de Grove 4-Digit Display library

TM1637 klokDisplay(2, 3);    // Maak een variabele klokDispay waarmee we de display bedienen

int tijdTotBuzzer = 20; // De timer loopt af na 20 seconden

// Dit gedeelte wordt slechts 1 keer uitgevoerd wanneer de Arduino wordt ingeschakeld
void setup() {
  Serial.begin(9600);
  pinMode(4, INPUT_PULLUP);          // We gaan pin 4 gebruiken als input
  pinMode(12, OUTPUT);               // We gaan pin 12 gebruiken als output

  klokDisplay.init();                // Start de display
  klokDisplay.set(3);                // Stel de helderheid in op 3 (0-7)
}

// Dit gedeelte wordt herhaald tot de microcontroller wordt uitgezet
void loop() {
  int knopwaarde = digitalRead(4);   // Lees of er een hoog of laag signaal op pin 4 staat

  if(knopwaarde == LOW) {           // Als het signaal op pin 4 LOW is:
    aftellen(tijdTotBuzzer);         // Roep de functie aftellen() aan
  }

  // Wacht 20 milliseconden zodat de microcontroller het niet te druk krijgt
  delay(20);
}

void aftellen(long aantalSecondenAftellen){
  // Bereken het aantal milliseconden dat we gaan aftellen
  long aantalMillisecondenAftellen = aantalSecondenAftellen * 1000;
  // De functie millis geeft het aantal milliseconden dat de microcontroller op dit moment aan staat
  long begintijd = millis();
  // Bereken op welk moment de timer moet afgaan
  long eindtijd = begintijd + aantalMillisecondenAftellen;
```

```
  // Voer onderstaande uit zolang de eindtijd niet bereikt is
  while(millis() < eindtijd) {
    // Bereken het aantal verstreken seconden
    long verstrekenSeconden = (millis() - begintijd) / 1000;
    // Bereken het aantal seconden totdat de timer afgaat
    long aantalSecondenTotEinde = aantalSecondenAftellen - verstrekenSeconden;
    // Roep tijdTotBuzzerWeergeven aan met het aantal seconden totdat de timer afgaat
    tijdTotBuzzerWeergeven(aantalSecondenTotEinde);
    // Wacht 20 milliseconden zodat de microcontroller het niet te druk krijgt
    delay(20);
  }

  //Roep nog een keer tijdTotBuzzerWeergeven() aan met 0 als argument zodat ook 00:00 nog wordt
weergegeven
  tijdTotBuzzerWeergeven(0);

  // Wanneer de eindtijd bereikt is komen we hier en zetten we de buzzer aan
  zetBuzzerAan();
  // Wacht 2 seconden
  delay(2000);
  // Zet de buzzer uit
  zetBuzzerUit();
}

// Deze functie zet de buzzer aan
void zetBuzzerAan() {
  digitalWrite(12, HIGH);       // Zet de buzzer aan met een hoog signaal op 12
}

// Deze functie zet de buzzer uit
void zetBuzzerUit() {
  digitalWrite(12, LOW);        // Zet de buzzer aan met een laag signaal op 12
}

// Deze functie toont de resterende tijd totdat de buzzer afgaat
void tijdTotBuzzerWeergeven(int aantalSecondenTotEinde){
    // Bereken het aantal hele minuten welke weergegeven moeten worden
    int minuten = aantalSecondenTotEinde / 60;
    // Bereken het aantal seconden welke weergegeven moeten worden
    int seconden = aantalSecondenTotEinde % 60;
    // Bereken de hele seconden welke we gaan weergeven op positie 3 van de display
    int heleSeconden = seconden % 10;
    // Bereken de tientallen seconden welke we gaan weergeven op positie 2 van de display
    int tientallenSeconden = seconden / 10 % 10;
    // Bereken de hele minuten welke we gaan weergeven op positie 1 van de display
    int heleMinuten = minuten % 10;
    // Bereken de tientallen minuten welke we gaan weergeven op positie 0 van de display
    int tientallenMinuten = minuten / 10 % 10;

    // Toon de juiste getallen op de display en zet de dubbele punt aan
    klokDisplay.display(0, tientallenMinuten);
    klokDisplay.display(1, heleMinuten);
    klokDisplay.point(1);
    klokDisplay.display(2, tientallenSeconden);
    klokDisplay.display(3, heleSeconden);
}
```

Arduino code oefening 8

2. Zorg ervoor dat de microcontroller verbonden is (blz. 10) met de computer, dat je het juiste board (blz. 9) en de juiste poort (blz. 10) geselecteerd hebt.

3. Klik op de ***Uploaden*** knop  om het programma te uploaden naar de microcontroller.

De klok zou nu gedurende het opgegeven aantal seconden moeten aftellen, en aan het einde hiervan zou de buzzer moeten piepen.

Is dit niet het geval? Bekijk dan de tips onder Problemen oplossen (blz. 40).

Problemen oplossen

Het verifiëren van mijn code is mislukt

- Doorloop in dit geval de stappen voor het kiezen van het bord (blz. 9) en het aansluiten van de microcontroller (blz. 10) nogmaals om te zien of je daar iets vergeten bent. Upload het programma daarna opnieuw.

- Heb je de code goed gekopieerd? Als je wijzigingen hebt gedaan kan het zijn dat je bijvoorbeeld een ";" bent vergeten. Begin eerst met het exact kopiëren van onze code. Als je daarna aanpassingen wilt doen, doe dit dan één voor één en controleer na iedere aanpassing of de code nog werkt.

- Gaat het uploaden van de code naar de microcontroller niet goed? Probeer eens een andere poort te selecteren, nog een keer te uploaden, weer terug te switchen naar de juiste poort en nog een keer te uploaden.

- Gaat het uploaden van de code fout en zie je in het rood de melding: “programmer is not responding”. Druk dan op de reset knop op de microcontroller vlak voordat je op upload klikt. Helpt dit nog niet? Haal de microcontroller dan van de stroom, sluit hem weer aan en probeer opnieuw te uploaden.

Mijn code is geupload maar de microcontroller doet niet wat ik verwacht.

- Het kan helpen om de microcontroller te resetten. Dit doen je door op de reset knop te drukken (zie afbeelding 0.1 (blz. 3).

De buzzer piept niet

- Let goed op of je de buzzer wel in de juiste oriëntatie in het breadboard hebt geprikt. Het korte pootje is de minpool en moet verbonden zijn met de GND, het lange pootje is de pluspool en moet verbonden zijn met een digitale pin op de microcontroller.

Gefeliciteerd!

Nu heb je een zelfgemaakte Countdown Timer. Met de onderdelen uit deze kit valt nog veel meer te bouwen. Probeer bijvoorbeeld de bovenstaande code eens om te bouwen totdat je een stopwatch hebt. Veel plezier!

Inspiratie

Voor nog meer inspiratie en projectideeën kun je ons volgen:

instagram.com/awesome.makes

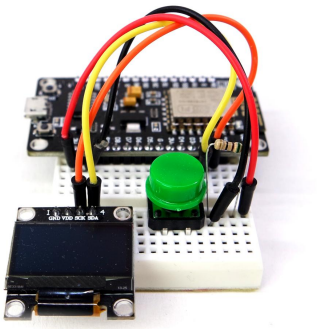
twitter.com/awesome_makes

facebook.com/AwesomeMakes

linkedin.com/company/awesome-makes

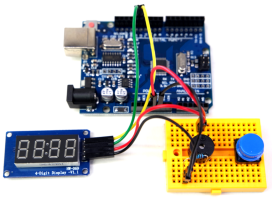
Meer bouwen?

Heb je de smaak te pakken en wil je meer bouwen? Bekijk dan eens de andere Awesome Crates op www.awesomemakes.com

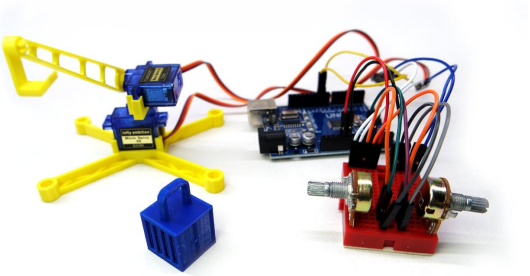


IOT Smart Display

IOT Mood Light



Countdown Timer



Bestuurbare Hijskraan