Arduino als dimmer

Arduino-experiment	130125-EN	*Trefwoorden*	analogWrite(), PWM, compound assignment		
Versie	2018-07-04 / HS	*Niveau*	Basiscursus, module 4	*p.*	1/3

Wat je leert:

- Gebruik een analoge Arduino output voor een dimmer
- Update een variabele met +=

- Ontdek dat "analoge" outputs niet simpelweg een DC-spanning leveren
- Begrijp *Pulse Width Modulation*

1 – Sluit een LED aan op een analoge output

"Analoge" output betekent hier een pin die het mogelijk maakt de spanning min of meer soepel te laten variëren, in tegenstelling tot "digitale" outputs die alleen hoog of laag kunnen zijn (5 V of 0 V).

Alle genummerde pinnen op de Arduino (inclusief de A0- tot A5-pinnen) kunnen worden gebruikt als **digitale** in- en outputs. Aan de andere kant kan slechts een beperkt aantal pinnen worden gebruikt als **analoge** in- en outputs. De analoge *inputs* (A0 tot A5) worden in een latere module behandeld. De beschikbare pinnen voor analoge *outputs* zijn de pinnen 3, 5, 6, 9, 10 en 11.

Om technische redenen (die we hier buiten beschouwing laten) zijn pinnen 5 en 6 niet zo flexibel als de rest – maar deze zijn gemakkelijk te vermijden als je Arduino pinnen in je project hebt staan.

Er is een soortgelijk probleem met pinnen 3 en 11. Ze kunnen niet gebruikt worden als je de `tone()` functie wilt gebruiken (zoals we gaan doen in experiment 130135 Tone output).

Om een lang verhaal kort maken: Gebruik pin 9 of 10 als analoge uitgang – tenzij er een reden is om het anders te doen.

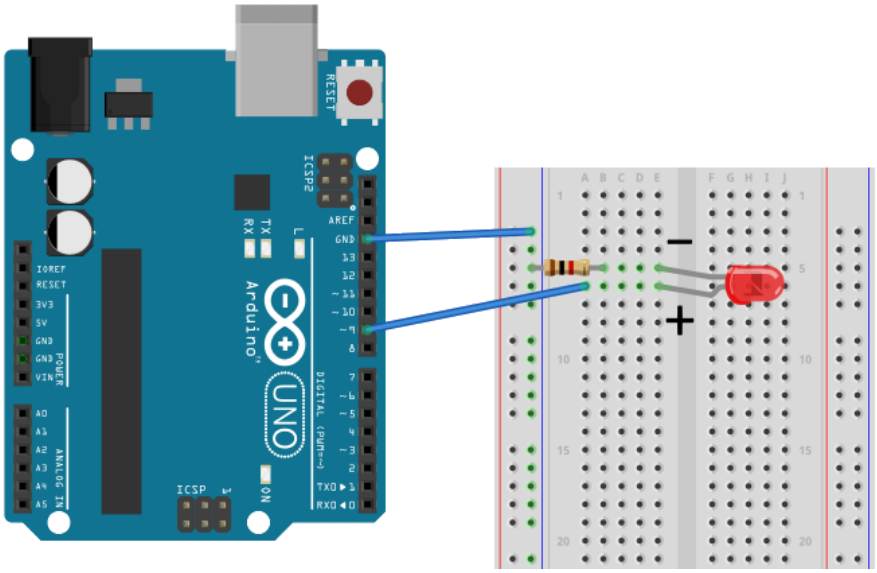
Het programma volgt hieronder. De opstelling is te zien op de tekening.

```
const byte pinOut = 9;

void setup() {
  // Niets in dit geval
}

int x=64;              // toegestane waarden: 0 .. 255

void loop() {
  analogWrite(pinOut,x);
}
```



Het programma op p. 1 zou de LED merkbaar minder helder moeten laten schijnen dan wat je in eerdere experimenten hebt gezien.

Probeer de beginwaarde van `x` te variëren en kijk wat er gebeurt.

Je eerste uitdaging hieronder is om de helderheidsverandering soepel op en neer te laten verlopen. Eerst een beetje hulp:

Maak de integer-variabele `dir` aan die de richting naar boven of beneden aangeeft. Geef het een beginwaarde 1. De waarde +1 betekent “omhoog”, de waarde -1 betekent “omlaag”.

We kunnen nu `x` laten stijgen of dalen met 1 voor elke uitgevoerde `loop()` , met behulp van deze instructie:

```
x += dir;
```

Toolbox: Compound operators

Voorbeeld:

```
x += dir;
```

Tel de waarde van `dir` op bij `x` .

Uitleg:

```
a += b;
```

is precies hetzelfde als

```
a = a + b;
```

Evenzo,

```
a -= b;
```

is precies hetzelfde als

```
a = a - b;
```

(Er bestaan meer compound assignment operators, maar deze zijn veruit de meest voorkomende)

De vreemde combinatie van + en = wordt een compound operator genoemd en wordt uitgelegd in de toolbox hiernaast.

Plaats een vertraging van 5 - 10 milliseconden voor elke uitgevoerde loop() , zodat alles iets minder snel gaat.

Tenzij je er iets aan doet, zal `x` snel over de grens van 255 schieten. Je zult dus een voorwaarde moeten inbrengen die `x` in de gaten houdt en het teken van `dir` verandert om `x` weer te laten afnemen.

Evenzo: wanneer `x` afneemt, zal deze snel onder nul duiken – je moet dit controleren met een andere voorwaarde en `dir` weer veranderen naar +1, net voordat dit gebeurt.

2 – Bouw een dimmer

Uitdaging 1

Bouw het programma op basis van de hierboven gepresenteerde ideeën.

De LED moet volledig donker beginnen. De helderheid moet vloeiend toenemen tot het maximum en daarna weer vloeiend afnemen tot nul.

(Dit moet zich oneindig blijven herhalen.)

Er zijn niet veel dimmers gemaakt om de intensiteit autonoom op en neer te laten gaan. Maar er zijn verschillende modellen die met een drukknop worden bediend.

Uitdaging 2

Voeg een schakelaar toe aan het project.

Zolang de knop wordt ingedrukt, moet de intensiteit variëren zoals in uitdaging 1 – maar de intensiteit moet op het gegeven niveau blijven staan wanneer de knop wordt losgelaten. Als de knop vervolgens weer wordt ingedrukt moet de helderheid niet vanaf nul beginnen, maar vanaf het huidige niveau.

Uitdaging 3

Verander het gedrag van het programma om de intensiteit te laten toenemen zolang de knop wordt ingedrukt – mogelijk tot maximaal. De volgende keer dat de knop wordt ingedrukt, moet de intensiteit afnemen terwijl hij wordt ingedrukt – mogelijk tot nul.

Met andere woorden: De richting verandert bij elke bediening van de knop – en niet automatisch naar de uitersten.

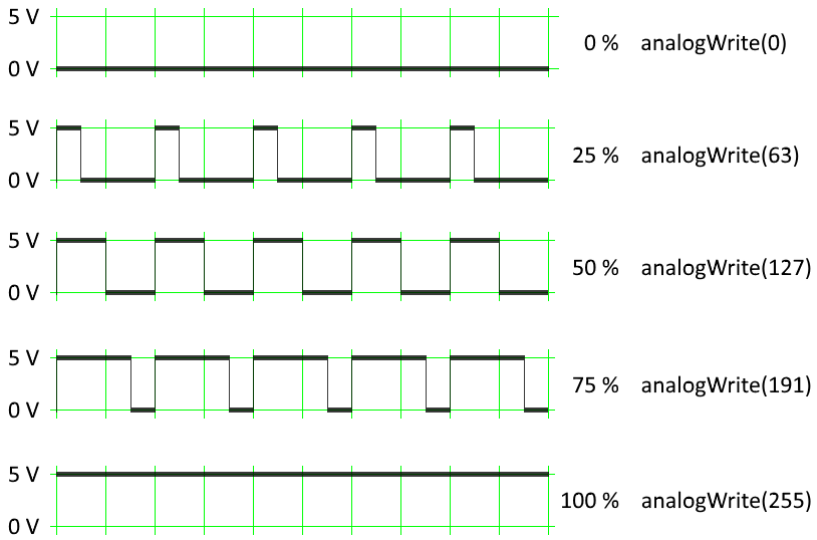
3 – PWM: Pulse Width Modulation

Wanneer je de LED vloeiend in intensiteit ziet variëren, denk je wellicht dat de spanning op de uitgangspin een gelijkspanning is die net zo soepel op en neer varieert. Niets is minder waar!

Als je de kans krijgt om een oscilloscoop aan te sluiten op de “analoge” output, zul je zien dat het een digitaal signaal betreft dat bijna 500 keer per seconde tussen 0 en 5 V slingert.

Wanneer je `analogWrite()` met verschillende parameters oproept, verander je in feite alleen maar hoe lang de output in elke periode hoog is. Met andere woorden, het is de *gemiddelde* waarde van het signaal die varieert wanneer `analogWrite()` wordt opgeroepen met verschillende waarden tussen 0 en 255. Dit soort signaal noemt men *pulse width modulated*, afgekort *PWM*.

Het percentage van de tijd dat het signaal hoog is, wordt de *duty cycle* van het signaal genoemd. In het onderstaande diagram vind je enkele voorbeelden van PWM-signalen met *duty cycles* tussen 0 en 100 %. Het is een goed idee om je te richten op de *duty cycle* als een percentage in plaats van de parameterwaarden van 0 tot 255 – deze waarden zijn geldig voor een Arduino, maar zullen niet noodzakelijkerwijs nuttig zijn met andere microcontrollers.



De LED knippert in feite als je een signaal geeft met een andere duty cycle dan 0 of 100 %. Maar het gebeurt zo snel dat het oog de variaties niet kan waarnemen.

Andere systemen reageren net zo traag als het menselijk oog. Zo kan een elektromotor zonder problemen worden aangestuurd met een dergelijk PWM-signaal.

4 – Bijkomend probleem

Dit gedeelte kan worden overgeslagen als je geen toegang hebt tot een oscilloscoop (of niet weet hoe je deze gebruikt).

Uitdaging 4 (vereist een oscilloscoop)

Sluit een paar korte, enkeldraadse draadstukken aan op het breadboard om een oscilloscoop-sonde te kunnen vastklemmen. De sondetip gaat naar pin 9, de aardeklem gaat naar GND op de Arduino.

Controleer of de golfvorm overeen komt met de beschrijving hierboven. Meet de frequentie van het PWM-signaal.