DE CONRAD ARDUINO STARTERKIT

```
void setup ()   {
   // maak kennis met Arduino
}
void loop ()   {
   // bouw zelf de mooiste projecten
}

/* de perfecte start voor beginners
   ideaal voor het onderwijs */
```

www.conrad.nl

“Ar-doe-wie-no”

Voorwoord

Leuk dat je meer wilt weten over Arduino. Wij helpen je graag een eindje op weg. De Conrad Arduino Starterkit dient om je te inspireren en biedt je een introductie tot de wereld van Arduino. Het is geen compleet naslagwerk, maar het geeft je wel de benodigde basis om zelf verder op ontdekkingsreis te gaan. Perfect voor iedere beginner, zeker als je (nog) geen ervaring hebt met elektronica of programmeren!

Veel plezier met je Arduino!

Conrad Electronic Benelux B.V.

Inhoudsopgave

Wat is Arduino

Arduino is de merknaam van een populaire serie microcontroller-boards. Deze boards bestaan (meestal) uit een Atmel ATmega microcontroller, wat ondersteunende componenten en een aantal aansluitingen. Samen vormen ze een soort kleine ‘computer’ waarmee je verschillende projecten zelf kunt aansturen.

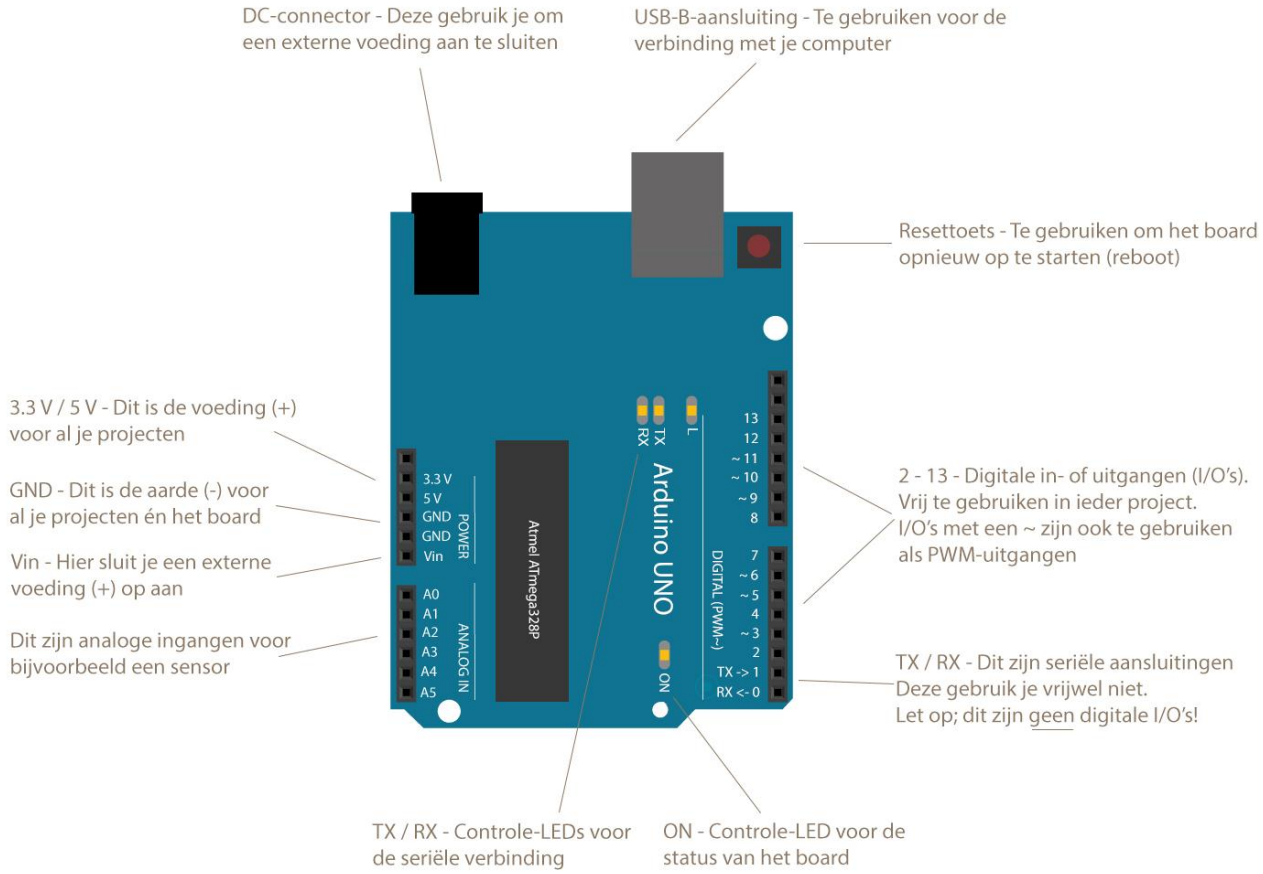
Een Arduino-board is gemaakt om op een voordelige manier je elektronicaprojecten aan te sturen. De microcontroller op het board is hier dan ook op geselecteerd. Een volwaardige computer met veel rekenkracht zou namelijk overbodig zijn voor dit doel en alleen tot hoge kosten leiden. De Arduino Uno heeft bijvoorbeeld een geheugen van slechts 32kB. Veel minder dan bijvoorbeeld een laptop, maar ruim voldoende om je elektronicaprojecten aan te sturen.

Arduino is een open-source-systeem. Dit betekent dat alle ontwerpen van de diverse boards door iedereen te bekijken zijn. Wanneer je zelf je eigen Arduino-board wilt maken, dan mag dat ook! De makers van Arduino hebben daarbij wel een voorwaarde gesteld; je mag het board geen Arduino noemen. Het grote voordeel van open-source-initiatieven als deze is dat veel gebruikers hun kennis en creativiteit in kunnen brengen. Er ontstaat zo al snel een grote groep mensen (community) die samen het originele idee kunnen verbeteren.

De Arduino hardware

Arduino-boards zijn er in diverse uitvoeringen. De verschillende boards hebben elk hun eigen voordelen, maar kennen ook veel overeenkomsten. Elke Arduino bestaat namelijk uit een microcontroller met daaromheen een aantal ingangen en uitgangen, ook wel I/O's genoemd (I/O staat voor Input/Output). Op de input sluit je een sensor aan, op de output een actor (met een maximaal verbruik van 40 mA per aansluiting). De sensor geeft de Arduino een reden om iets te doen. De actor voert vervolgens de daadwerkelijke actie uit. De software bepaalt tussen deze beide stappen in wat er moet gebeuren.

De meest eenvoudige (en tegelijk ook de populairste) Arduino is de Arduino Uno. De meest relevante onderdelen van dit board vind je hieronder. Voor de projecten in deze Conrad Arduino Starterkit ga je alleen een aantal van deze aansluitingen gebruiken.



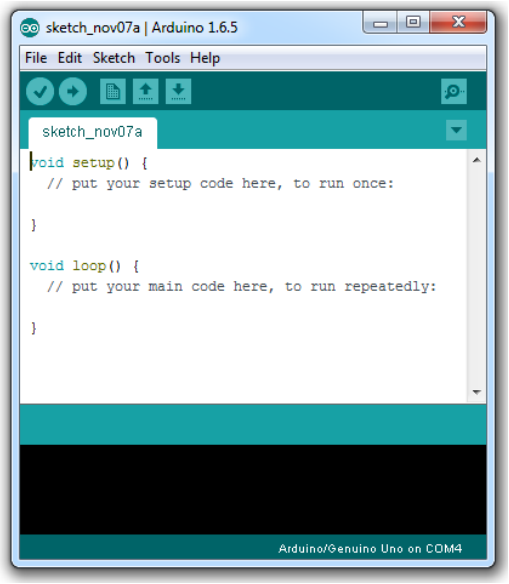
De Arduino software

Je weet nu wat Arduino is en hoe de hardware van een board is opgebouwd. Deze hardware kan echter geen taken uitvoeren zonder dat iemand vertelt wat er moet gebeuren. Hier komt de Arduino software om de hoek kijken.

Het programmeren van een Arduino-board doe je via de Arduino IDE (Integrated Development Environment). Deze IDE is gratis te downloaden van de officiële Arduino-website (www.arduino.cc) en biedt je een volledige programmeeromgeving met alle noodzakelijke elementen. De Arduino programmeertaal is gebaseerd op C/C++. De Arduino IDE is ook weer open-source. Dit betekent dat ook de Arduino IDE door iedereen vrij te gebruiken en bewerken is.

Arduino sketch

Een programma waarmee je het Arduino-board vertelt wat deze moet doen heet een ‘sketch’. Een sketch bevat alle noodzakelijke elementen om jouw project goed te laten functioneren. Deze elementen worden via de IDE omgezet in concrete taken voor de hardware. Je kunt een sketch zelf schrijven, maar via de grote Arduino-community (playground.arduino.cc) zijn ook al heel veel kant-en-klare sketches te vinden voor tal van inspirerende projecten. Je kunt deze sketches compleet overnemen of er juist delen uitpakken die voor jouw project interessant zijn.



Setup() en loop()

Iedere Arduino-sketch bestaat uit minimaal twee elementen; de setup() en de loop(). In de setup() benoem je alles wat van belang is voordat de Arduino daadwerkelijk een taak gaat uitvoeren. Aangeven dat een aansluiting functioneert als output vindt bijvoorbeeld plaats in de setup(). In de loop() vertel je welke stappen de Arduino daadwerkelijk moet doorlopen. Wil je bijvoorbeeld een sensor uitmeten, dan geef je dit aan in de loop(). Een loop herhaalt zichzelf keer op keer, de setup vindt eenmalig plaats bij het starten van de sketch.

```
// Hier geven we eenmalig aan wat de instellingen voor deze sketch moeten zijn
void setup() {
  pinMode(led, OUTPUT);
  pinMode(toets, INPUT);
}

// Hier geven we aan welke stappen de Arduino moet afwerken in de sketch
void loop() {
  int toetsWaarde = digitalRead(toets);
```

Functies

In een Arduino sketch ga je een aantal functies gebruiken. Een functie is een vooraf gedefinieerde code die je gebruikt om een instructie te geven aan het board. Met de *pinMode* functie geef je bijvoorbeeld aan of een I/O zich moet gedragen als input of output. De functie *analogRead* geeft aan dat er een analoge waarde uitgelezen moet worden. Met *digitalWrite* geef je het board de instructie om een output aan (HIGH) of uit (LOW) te zetten.

```
digitalWrite(led, HIGH);
```

Variabelen

Een variabele bestaat uit een naam, een waarde en een type. Met een variabele sla je als het ware een waarde op zodat je deze later in de sketch weer kunt gebruiken. Er zijn verschillende variabelen, sommige maak je zelf, andere zijn al vooraf gedefinieerd.

Met de variabele ‘int ledPin = 13; geef je bijvoorbeeld aan dat je de aansluiting met waarde ‘13’ in de sketch gaat aanspreken met de naam ‘ledPin’. Met het type ‘int’ geef je aan dat er een geheel getal wordt opgeslagen, een integer.

```
int ledPin = 13;
```

Speciale karakters

Je komt verschillende leestekens tegen in een Arduino-sketch. De belangrijkste zijn:
; (semicolon) –gebruikt om vrijwel alle opdrachten af te sluiten
{ } (braces) – gebruikt om een commando (statement) te starten of te eindigen (vaak vergeten)
= (assignment operator) – slaat de waarde rechts van de = op in de variabele links er van
== (equal to) – deze gebruik je als ‘gelijk aan’ voorwaarde

Comments

De meeste Arduino-sketches bevatten commentaar (comments). Via deze comments kan de schrijver van de sketch je een toelichting geven op wat er gebeurt. Door zelf comments toe te voegen in je sketch weet je later ook nog waarom je bepaalde functies en variabelen hebt gebruikt.
Alles tussen /* en */ wordt door de Arduino IDE gezien als comment en wordt niet uitgevoerd.
Hetzelfde geldt voor alles na // op dezelfde regel.

```
/*
 deze comment kun je over
 meerdere regels plaatsen
*/

// Deze comment plaats je achter een deel van je sketch
```

Tot slot

Je hebt nu kennis gemaakt met de meest gebruikte onderdelen uit een Arduino-sketch. De officiële Arduino-website (www.arduino.cc) biedt je altijd een compleet naslagwerk met informatie over variabelen, functies en andere elementen rondom het programmeren van Arduino.

In de volgende hoofdstukken ga je de opgedane kennis toepassen in de praktijk. Tijd om je handen uit de mouwen te steken!

Inhoud van de starterkit

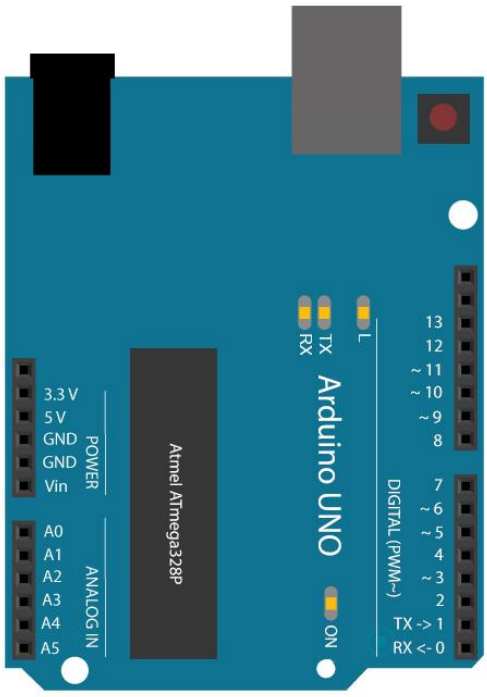
Deze starterkit bevat – naast de Arduino UNO – een aantal onderdelen om zelf verschillende leuke projecten te bouwen. We zetten de onderdelen hier even voor je op een rijtje.

Tip: met een 3D-printer maak je zelf een mooie houder voor de Arduino en het breadboard (design: conr.nl/arduino3D).

Aantal	Onderdeel	Bestnr.
1x	Arduino Uno	1275279
1x	USB-kabel	973569
1x	Breadboard	526819
1x	Draadbruggen	524530
5x	LED rood 5mm	184543
5x	LED groen 5 mm	184705
5x	LED blauw 5mm	1050457
5x	LED wit 5mm	185890
2x	LED RGB 5mm	1050465
10x	Weerstand 150 Ohm 5% 0,25W	1089137
10x	Weerstand 100 Ohm 5% 0,25W	1089135
5x	Weerstand 68 Ohm 5% 0,25W	1089133
5x	Weerstand 10kOhm 5% 0,25W	1089159

Aantal	Onderdeel	Bestnr.
2x	Diode (1N4001)	162213
2x	Transistor (PN2222)	563907
1x	Piëzo zoemer	717609
2x	Optocoupler (4N35)	140257
2x	Lichtsensor (VT93N2)	140375
1x	Temperatuursensor (LM35)	156600
1x	Reedcontact	1233073
1x	Magneet	505892
2x	Druktoets	700324
2x	Potmeter	432024
2x	Potmeter as	425508
1x	USB-netvoeding	518397

Arduino UNO
Het Arduino-board waar deze starterkit om draait. Dit board is de basis voor alle volgende projecten.

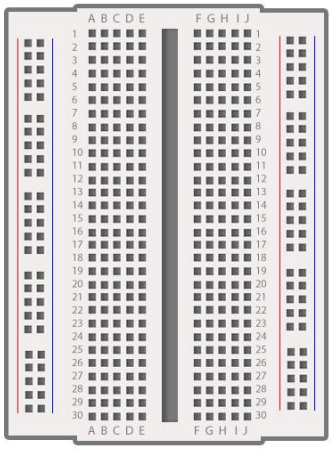


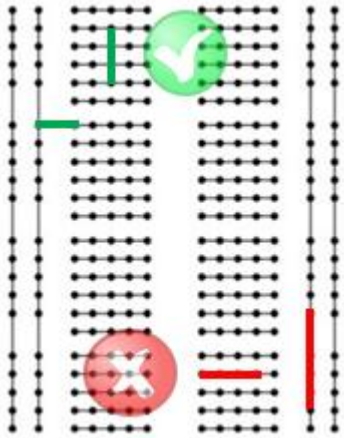
USB-B-kabel
Je sluit het Arduino-board via deze kabel aan op de USB-poort van je computer.

Wil je het Arduino-board aansluiten zonder tussenkomst van een computer. Dan kun je hier de (meegeleverde) stekkernetvoeding op aansluiten.

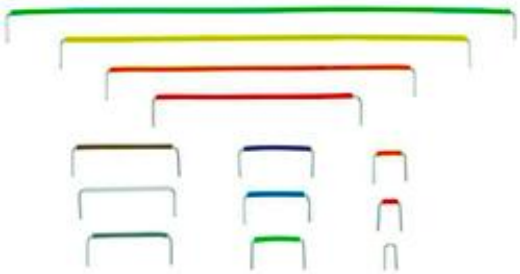
Breadboard
Een breadboard is perfect voor het maken van testopstellingen. Het is in feite een printplaat waarbij je niet hoeft te solderen. Je steekt de verschillende onderdelen eenvoudig in de gaten. Deze gaatjes zijn onderling met elkaar verbonden. Het meegeleverde breadboard heeft 400 gaatjes en 64 stroken.

Let op; maak altijd verbindingen tussen de stroken op het breadboard. Een verbinding binnen een strook geeft kortsluiting. Zie ook de schematische weergave van het breadboard hieronder.



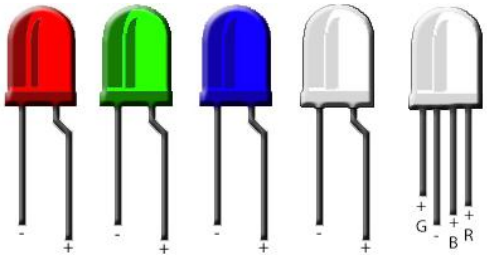


Draadbrug
Het maken van een verbinding tussen de Arduino en het breadboard of tussen de componenten op het breadboard maak je met deze draadbruggen. Ze zijn er in verschillende afmetingen en kleuren. De kleur zorgt voor overzichtelijkheid, maar heeft verder geen functie



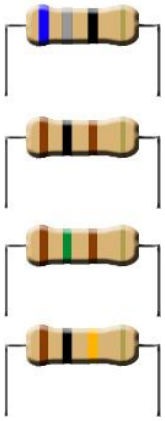
LED
De starterkit wordt geleverd met LED's in verschillende kleuren. Deze kleur is niet het enige verschil; de rode en groene LED's werken namelijk op een andere spanning dan de blauwe LED, ook de witte (transparante) LED en de RGB-LED (deze heeft 4-pootjes) hebben een andere voedingsspanning nodig. Je zult hier rekening mee moeten houden bij het bepalen van de voorschakelweerstand.

Wat wel gelijk is bij iedere LED is de plus- en min. Bij een component noem je dit de anode (+) en kathode (-). Je herkent de verschillende polen aan de lengte van het pootje. Het langere pootje is bij LED's altijd de anode (+). Een RGB-LED heeft één kathode (-) en voor iedere kleur een anode (+).



Weerstand
Een weerstand is een onderdeel met de elektrische eigenschap ‘weerstand’. Spanning, stroom en weerstand staan tot elkaar in verhouding (de Wet van Ohm). Verhoog je de weerstand (in Ohm), dan heeft dit effect op de spanning en stroom. In deze projecten ga je de weerstanden gebruiken om spanning te verlagen of zelfs compleet tegen te houden.

De weerstanden hebben verschillende waarden. Je herkent deze waarden via de gekleurde ringen op de weerstand (bruin/groen/bruin/goud is een 150 Ohm weerstand met 5% tolerantie). Alle weerstanden in deze set hebben een tolerantie van 5% (we noemen de gouden ring dan ook niet bij het benoemen van de weerstand in een project). Een weerstand heeft geen polariteit (plus- of minpool), het maakt dus niet uit of je ze links- of rechtsom gebruikt.



Kleur	Ring 1 1e cijfer	Ring 2 2e cijfer	Ring 3 multiplicator	Ring 4 tolerantie
zwart		0	1	
bruin	1	1	10	1 %
rood	2	2	100	2 %
oranje	3	3	1.000	
geel	4	4	10.000	
groen	5	5	100.000	0,5 %
blauw	6	6	1.000.000	
violet	7	7	10.000.000	
grijs	8	8		
wit	9	9		
goud			0,1	5 %
zilver			0,01	10 %

Potmeter
Een potmeter gedraagt zich als een soort variabele weerstand. De potmeter in deze set heeft een instelbare weerstandswaarde tussen 0 Ohm en 10kOhm (oftewel 10.000 Ohm). Door te draaien aan de as stel je de potmeter in en kun je onderdelen van je schakeling aansturen. Een potmeter heeft een plus-, min- en een V^{out}. Het laatste staat voor de uitgangsspanning, deze aansluiting is het pootje aan de overzijde van de andere twee en gebruik je als variabele uitgang.

Druktoets
De druktoetsen in deze set zijn van het type uit/(aan). Dit houdt in dat het contact in de rustpositie open staat, de toets maakt in dat geval geen contact. Wanneer je de toets indrukt, dan maakt deze contact zolang de toets wordt ingehouden. Dit contact wordt gevormd tussen links en rechts (zie de afbeelding). De tegenover elkaar liggende aansluitingen maken standaard al contact.

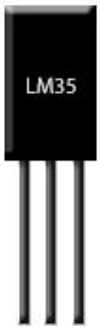
De contacten in de toets

Reedcontact
Een reedcontact is in feite een magnetische toets. Wanneer je er een magneet naast houdt, dan sluit het contact zich. Haal je de magneet weg, dan wordt het contact onderbroken. Je kunt dit type ‘toets’ daardoor perfect gebruiken in bijvoorbeeld alarmsystemen of zelfs toerentellers.

Piëzo-module (zoemer)
Deze module maakt gebruik van het piëzo-effect; er ontstaat elektrische spanning bij een drukverschil en andersom (er ontstaat een drukverschil en dus geluid bij een elektrische spanning). In deze starterkit gebruiken we de module alleen om geluid te produceren.

Temperatuursensor
Via deze sensor kan de Arduino uitlezen wat de actuele temperatuur is. Je kunt hierdoor een interactieve schakeling bouwen die zelfstandig kan reageren op de omgeving!

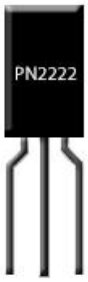
De sensor heeft drie aansluitingen. Vanaf de platte kant gezien zijn dit van links naar rechts de plus-, V^{out}en de min.



Lichtsensor
We noemen dit een lichtsensor omdat het onderdeel reageert op de hoeveelheid licht. Het is eigenlijk een lichtgevoelige weerstand. De geboden weerstand verandert namelijk zodra de hoeveelheid licht verandert. Ook deze sensor maakt het mogelijk om een interactieve schakeling te bouwen.

Diode
Een diode laat elektriciteit in één richting door. Alle stroom in de tegenovergestelde richting (sperrichting) wordt tegen gehouden. Dit maakt een diode handig als beveiliging van ongewilde stromen binnen je schakeling (denk aan een kleine DC-motor waarmee je onbedoeld stroom genereert door er aan te draaien). De grijze ring geeft de kathode (-) van de diode aan.

Transistor
Een transistor is in feit een soort schakelaar. Door een kleine spanning op de transistor zelf te zetten kan deze op zijn beurt een grotere spanning doorlaten. Ideaal als je met de Arduino een zwaardere verbruiker (met een externe voeding) wil inschakelen. Het direct aansluiten op je Arduino-board zou tot beschadigingen kunnen leiden.



Optocoupler
Een optocoupler is een soort schakelaar met binnenin een LED én een transistor. Wanneer de LED gaat branden, dan zal de (lichtgevoelige) transistor gaan werken. De optocoupler is daardoor ‘galvanisch gescheiden’. De afzonderlijke delen maken namelijk geen fysiek contact. Je kunt een optocoupler gebruiken om eenvoudig bestaande apparatuur te schakelen met je Arduino.

USB-netvoeding
Als je een mooi project hebt gebouwd, dan wil je deze natuurlijk ook gebruiken. Via deze USB-netvoeding kun je het project overal gebruiken. Als er maar een stopcontact in de buurt is. Je kunt deze USB-netvoeding combineren met dezelfde USB-B-kabel die je ook gebruikt bij het verbinden met de computer.

Arduino installeren

Het programmeren van een Arduino board doe je via een computer. Hiervoor moet wel eerst de benodigde software worden geïnstalleerd. We zullen je hier stap-voor-stap door het installatieproces op een Windows besturingssysteem loodsen.

Heb je Mac OS of Linux? Kijk dan even op www.arduino.cc voor een handleiding over deze besturingssystemen.

Arduino installeren op een Windows-computer
1. Download de laatste versie van de Arduino IDE op conr.nl/arduinoIDE (deze software wordt regelmatig geüpdatet).

2. Sluit het Arduino-board via een USB-kabel aan op de USB-poort van je computer.

3. Installeer de drivers
- Windows gaat eerst zelf proberen om de drivers te vinden. Dit zal mislukken, dit ligt niet aan jou of je computer.

- Open het configuratiescherm in Windows (Control Panel), vervolgens 'Systeem en Beveiliging', 'Systeem' en als laatste 'Apparaatbeheer'.

- Kijk onder poorten (COM & LPT). Hier vind je de Arduino. Is dit niet het geval, kijk dan onder de overige apparaten voor onbekende apparaten.

- Klik met je rechtermuisknop op de gevonden Arduino en kies 'Update Driver Software'.

- Kies vervolgens voor de optie ‘Zoek op mijn computer’ om de driver software te vinden.

- Navigeer naar het bestand 'arduino.inf' in de bij stap 1 gedownloade map. Het bestand staat in de submap 'Drivers'.

- Windows gaat nu de benodigde drivers voor het Arduino-board installeren.

4. Open de zojuist geïnstalleerde Arduino software.

5. Open de demo-sketch genaamd 'LED Blink'. ‘File ->examples -> 01.Basics -> Blink’

6. Kies je Arduino-board bij ‘Tools -> board’

7. Selecteer de seriële poort waar je Arduino op is aangesloten bij Tool -> Serial Port menu

8. Klik op de 'Upload' knop in de software (het pijltje naar rechts) en wacht een paar seconden, de sketch wordt geüpload naar je Arduino-board.

9. Tijdens het uploaden zullen de RX en de TX LED’s op het board beginnen te knipperen, wanneer alles gelukt is, dan begint de LED bij pin 13 te knipperen.

10. Gefeliciteerd! De Arduino-software is geïnstalleerd en je hebt zojuist je eerste sketch op het Arduino-board gezet.

Je kunt nu beginnen met het schrijven van je eigen sketches in de Arduino-software. Wanneer je tevreden bent met het resultaat dan kun je deze uploaden naar het Arduin-board.

Wil je weer compleet opnieuw beginnen? Dan kun je bij 'File -->Examples' de sketch 'BareMinimum' vinden. Als je deze upload naar je board, dan is alles weer teruggezet naar de fabrieksinstelling.

Arduino projecten

Je weet nu genoeg van Arduino en de onderdelen uit deze starterkit om zelf aan de slag te gaan. We hebben een aantal leuke projecten voor je bedacht. Je zult merken dat je later bij het maken van eigen projecten veel elementen uit deze projecten weer zult gebruiken.

Deze projecten zijn uitermate geschikt voor beginners en laten je stap-voor-stap kennis maken met het programmeren van de Arduino en het opbouwen van elektrische schakelingen. Ieder project bestaat uit een deel waarbij de schakeling wordt opgebouwd en een deel waarbij je de Arduino programmeert.

Je kunt er zelf voor kiezen om beide onderdelen door te werken of één van de onderdelen simpelweg over te nemen van het voorbeeld wanneer je hier geen tijd of zin in hebt.

Je kunt alle Arduino-sketches ook kant & klaar downloaden via conr.nl/arduinostarterkit.

Project: Arduino schakelaar

In dit project ga je een eenvoudige schakelaar bouwen. Wanneer je op de toets drukt dan gaat de LED branden. Dit kan in feite ook zonder Arduino door de LED en de toets in serie te schakelen, maar dit voorbeeld geeft erg goed de werking van de I/O's op je Arduino-board aan.

Benodigde onderdelen:
- 1 Rode LED
- 1 Weerstand 150 Ohm (bruin/groen/bruin)
- 1 Druktoets
- 1 Weerstand 10 kOhm (bruin/zwart/oranje)
- 5 Draadbruggen
- 1 Breadboard

Arduino schakelaar | de schakeling

Op de volgende pagina vind je een duidelijk schema van de schakeling.

1. Plaats een draadbrug tussen de 5V-aansluiting van je Arduino en de rood gemarkeerde strook op het breadboard. Het maakt niet uit welke draadbrug je hier voor kiest.
2. Plaats een draadbrug tussen de GND-aansluiting van je Arduino en de blauw gemarkeerde strook op het breadboard.

   // De eerste stap bij het bouwen van vrijwel iedere schakeling is het aansluiten van de voeding op het breadboard. De 5V- en GND-aansluiting van je Arduino dienen in dit project als voeding, deze worden doorverbonden naar het breadboard waardoor ook hier spanning op staat.

3. Plaats de rode LED op het breadboard, let daarbij goed op de polariteit (+/-) van de LED en verbind de pootjes tussen twee stroken op het breadboard.
4. Plaats een draadbrug tussen de anode (+) van de LED en digitale I/O 3 op de Arduino.

   // Je sluit de anode van de LED via een draadbrug aan op de Arduino en niet op 5V omdat je de LED wilt aansturen met de Arduino op basis van input (de druktoets).

5. Plaats een weerstand van 150 Ohm (bruin/groen/bruin) tussen de kathode (-) van de LED en de blauwe GND-strook van het breadboard.

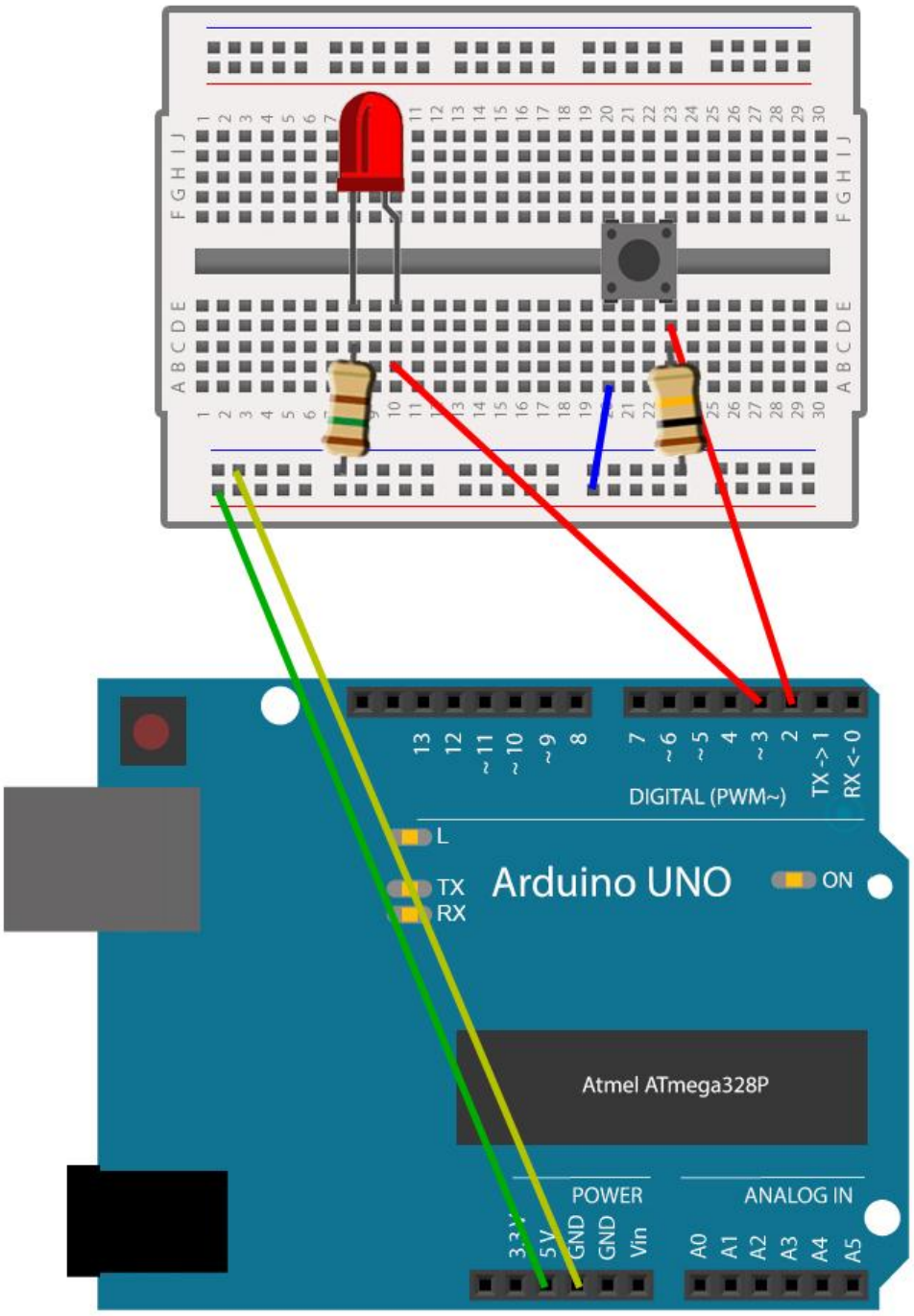
   // Deze LED heeft een voorschakelweerstand van 150 Ohm nodig om niet teveel spanning over de LED te laten lopen (deze is geschikt voor maximaal 2,25V).

6. Plaats de druktoets tussen de twee zijden van je breadboard.

   // De druktoets maakt pas contact wanneer je deze indrukt. Dit contact wordt gevormd tussen de twee aansluitingen aan dezelfde zijde. De overliggende aansluitingen maken standaard al contact met elkaar.

7. Sluit het linker pootje van de toets (zie schema) aan op de op 5V strook van het breadboard.
8. Plaats een weerstand van 10kOhm (bruin/zwart/oranje) tussen het pootje aan de rechterzijde van de toets en de op GND aangesloten strook van het breadboard.
9. Verbind het rechter pootje van de toets via een draadbrug met digitale aansluiting 2 van je Arduino. Plaats deze draadbrug tussen de 10 kOhm weerstand uit stap 8 en de toets.

   // Wanneer je een digitale I/O als ingang gebruikt, dan kan er storing ontstaan wanneer er weinig tot geen spanning op deze ingang staat. Voor momenten dat de toets niet wordt ingedrukt wil je een veiligheid inbouwen die alle spanning tegenhoudt (de spanning is daardoor altijd 0). Deze veiligheid noem je een 'pull-down-weerstand' en plaats je in de vorm van een 10 kOhm weerstand tussen GND en digitale aansluiting 2.


DIGITAL (PWM~)
TX -> 1
RX <- 0
L
TX
RX
Arduino UNO
ON
Atmel ATmega328P
POWER
ANALOG IN
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5

Arduino schakelaar | de sketch

Je gaat nu de sketch schrijven om alle verschillende onderdelen van de hardware aan te sturen. Open hiervoor de Arduino software op je computer. Er opent vanzelf een lege sketch.

We gaan nu stap-voor-stap de verschillende onderdelen van de sketch bespreken.

Je hebt twee aansluitingen gebruikt in je opstelling. Om de sketch begrijpelijk te houden is het makkelijk om deze aansluitingen aan het begin van de Sketch vast een naam te geven. Deze aansluiting verandert niet (het is een constante) en heeft een gehele numerieke waarde (integer). De LED is bijvoorbeeld aangesloten op aansluiting 3.

Let op: alles in de Arduino-sketch is hoofdlettergevoelig! Het is daarbij gebruikelijk om eigen waarden te beginnen met een kleine letter om vervolgens over te gaan naar een hoofdletter voor elk nieuw woord (kleinGroot). Dit hoeft niet, maar je zult zien dat veel mensen het wel zo toepassen.

```
// Benoem de noodzakelijke elementen van deze sketch
const int led = 3;
const int toets = 2;
```

Wat nu volgt is de setup(), hier geef je eenmalig aan wat de juiste instellingen voor deze sketch zijn.

Geef als eerste aan dat de aansluiting die je in de vorige stap hebt aangeduid als LED dient als uitgang (output) in deze sketch. De toets wordt gebruikt als ingang (input).

```
// Hier geven we eenmalig aan wat de instellingen voor deze sketch moeten zijn
void setup() {
  pinMode(led, OUTPUT);
  pinMode(toets, INPUT);
}
```

De sketch komt nu aan bij het loop() gedeelte waar daadwerkelijke acties zich keer op keer herhalen.

Als eerste maak je hier een waarde (int) aan genaamd ‘toetsWaarde’. Dit is de digitale waarde (aan of uit) die wordt uitgelezen bij de druktoets. Hiermee bepaal je dadelijk of de LED aan of uit moet.

```
// Hier geven we aan welke stappen de Arduino moet afwerken in de sketch
void loop() {
  int toetsWaarde = digitalRead(toets);
```

Je begint met het creëren van een voorwaarde. Namelijk; als (if) de toetsWaarde aan is ‘HIGH’. Als aan deze voorwaarde wordt voldaan (== betekent gelijk aan), dan wordt er spanning gezet op de LED-aansluiting. Dit doe je door deze digitale uitgang op ‘HIGH’ te zetten.
Als de toetswaarde niet ‘HIGH’ is (de waarde is dan dus anders, oftewel ‘else’), dan moet de LED uit blijven. Dit doe je door de LED-uitgang op ‘LOW’ te zetten.

Het haakje } op het einde dient als afsluiting van de loop(). Je hebt deze namelijk ook geopend met een haakje {.

```
if(toetsWaarde == HIGH){
  digitalWrite(led, HIGH);
} else {
  digitalWrite(led, LOW);
 }
}
```

Arduino schakelaar | uitvoeren van de sketch

Sluit nu de Arduino via de USB-kabel aan op de computer. Je zult zien dat het ON-lampje gaat branden. Nu kun je de sketch uploaden naar de Arduino via de upload-knop in de software. De RX/TX-lampjes gaan even knipperen, vervolgens zie je in de software staan dat de upload is geslaagd. De schakelaar is klaar voor gebruik!

Druk nu eens op de toets. Je zult zien dat de LED aan gaat. Zodra je de toets los laat schakelt de LED ook weer uit.

Je hebt nu zelf een eenvoudige schakelaar gebouwd. Je kunt hetzelfde principe natuurlijk ook gebruiken als schakelaar in verschillende andere projecten!

Project: Arduino Knight Rider

In dit project ga je het bekende lichteffect van de Knight Rider bouwen (voor de liefhebber; dit effect heet officieel een Larson Scanner). De LED's knipperen in dit project snel na elkaar. Door het toevoegen van een potmeter kun je zelf de snelheid regelen.

Benodigde onderdelen:
- 5 Rode LED's
- 5 Weerstanden 150 Ohm (bruin/groen/bruin)
- 1 Potmeter
- 10 Draadbruggen
- 1 Breadboard

Arduino Knight Rider | de schakeling

Op de volgende pagina vind je een duidelijk schema van de schakeling.

1. Plaats een draadbrug tussen de 5V-aansluiting van je Arduino en de rood gemarkeerde strook op het breadboard. Het maakt niet uit welke draadbrug je hier voor kiest.
2. Plaats een draadbrug tussen de GND-aansluiting van je Arduino en de blauw gemarkeerde strook op het breadboard.

   // De eerste stap bij het bouwen van vrijwel iedere schakeling is het aansluiten van de voeding op het breadboard. De 5V- en GND-aansluiting van je Arduino dienen in dit project als voeding, deze worden doorverbonden naar het breadboard waardoor ook hier spanning op staat.

3. Plaats de rode LED's naast elkaar op het breadboard, let daarbij goed op de polariteit (+/-) van de LED's en verbind de pootjes tussen twee stroken op het breadboard.
4. Plaats draadbruggen tussen de anode (+) van de LED's en digitale I/O 2 tot en met 6 op de Arduino (doe dit op volgorde, anders knipperen de LED's straks niet zoals gewenst).

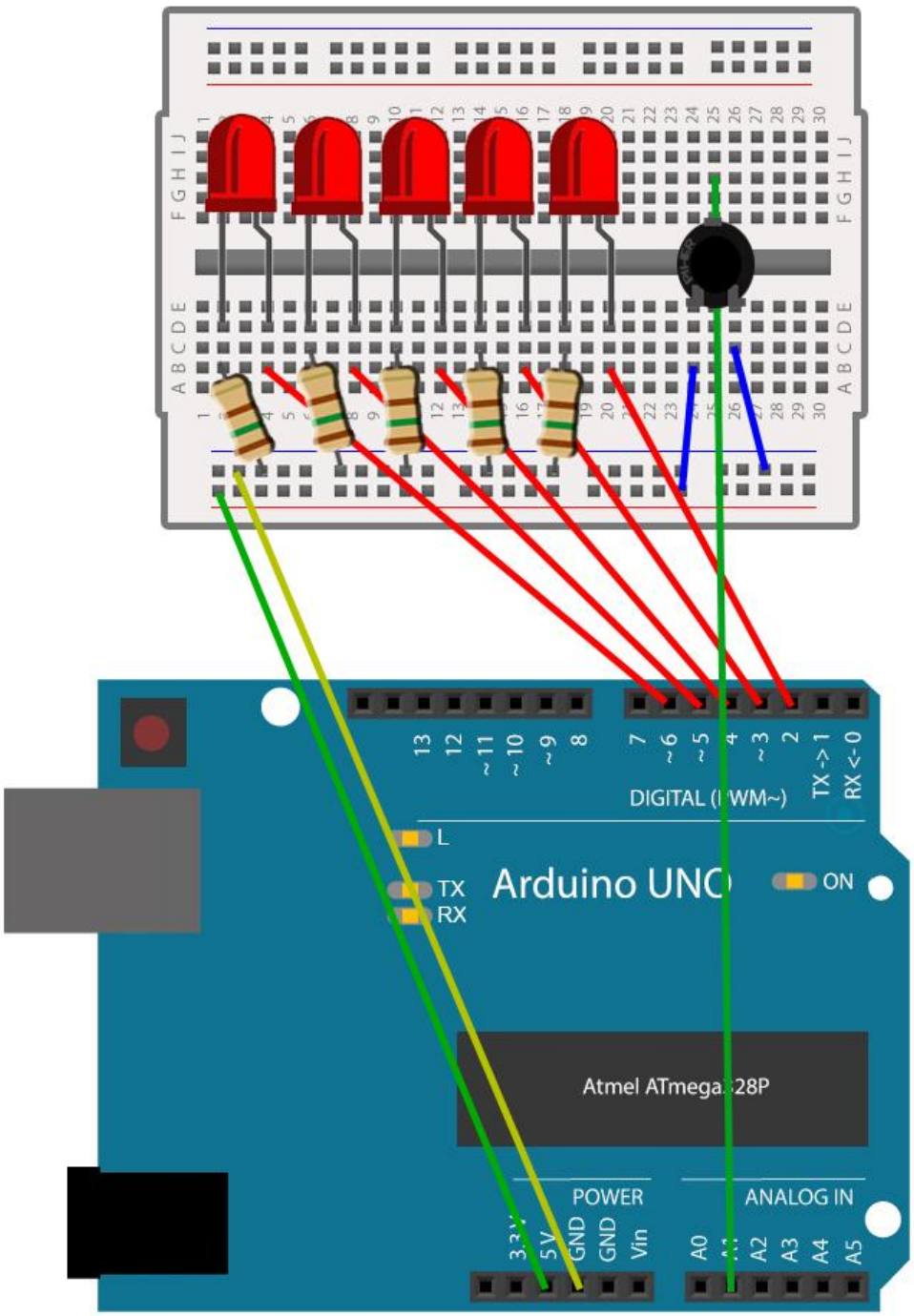
   // Je sluit de anode van de LED's via een draadbrug aan op de Arduino en niet op 5V omdat je de LED's wilt aansturen met de Arduino op basis van tijd.

5. Plaats bij iedere rode LED een weerstand van 150 Ohm (bruin/groen/bruin) tussen de kathode (-) van de LED en de blauwe GND-strook van het breadboard.

   // De rode LED heeft een voorschakelweerstand van 150 Ohm nodig om niet teveel spanning over de LED te laten lopen (deze is geschikt voor maximaal 2,25V).

6. Plaats de potmeter over het midden van je breadboard. Zorg ervoor dat de zijde met twee pinnen aan de kant van de LED's zit. De enkele pin wordt aan de overzijde in het breadboard geplaatst.
7. Plaats een draadbrug tussen de linkerpin van de potmeter en de 5V van het breadboard.
8. Plaats een draadbrug tussen de rechterpin van de potmeter en de GND van het breadboard.
9. Plaats een draadbrug tussen de achterste pin van de potmeter en de A1 van de Arduino.

   // Een potmeter is een spanningsdeler, maar dan in één behuizing. Door te draaien aan de as wijzig je de uitgangsspannning. De uitgangsspanning loopt in dit geval richting de analoge A1-aansluiting van de Arduino.


DIGITAL (PWM~)
TX -> 1
RX <- 0
L
TX
RX
ON
POWER
ANALOG IN
GND
GND
Vin
A0
A2
A3
A4
A5

Arduino Knight Rider | de sketch

Je gaat nu de sketch schrijven om alle verschillende onderdelen van de hardware aan te sturen. Open hiervoor de Arduino software op je computer. Er opent vanzelf een lege sketch.

We gaan nu stap-voor-stap de verschillende onderdelen van de sketch bespreken.

Je hebt veel aansluitingen gebruikt in je opstelling. Om de sketch begrijpelijk te houden is het makkelijk om deze aansluitingen een naam te geven. Je hoeft dit niet per se te doen (je kunt verderop in de sketch ook verwijzen naar het nummer van de I/O). Het benoemen van de I/O's houdt je sketch echter wel een stuk overzichtelijker.

```
// Benoem de elementen die je in deze sketch wilt gebruiken
const int led1 = 2;
const int led2 = 3;
const int led3 = 4;
const int led4 = 5;
const int led5 = 6;
const int potmeter = A1;
```

Wat nu volgt is de setup(), hier geef je eenmalig aan wat de juiste instellingen voor deze sketch zijn.

Geef aan dat de aansluitingen die je in de vorige stap hebt aangeduid als led1 tot en met led6 dienen als uitgang (output) in deze sketch. We benoemen ze hier allemaal afzonderlijk om het begrijpelijk te houden. Je kunt ze echter ook allemaal tegelijkertijd benoemen via een zogenaamde 'for loop'. Je bespaart daarmee wat code én dus geheugen *(zie het Arduino thermometer project voor een dergelijke 'for loop')*.

```
// Hier geven we eenmalig aan wat de instellingen voor deze sketch moeten zijn
void setup(){
  pinMode(led1, OUTPUT);
  pinMode(led2, OUTPUT);
  pinMode(led3, OUTPUT);
  pinMode(led4, OUTPUT);
  pinMode(led5, OUTPUT);
}
```

De sketch komt nu aan bij het loop() gedeelte waar daadwerkelijke acties zich keer op keer herhalen.

Als eerste maak je hier een waarde aan genaamd 'snelheidsRegelaar'. Dit is de analoge waarde (een cijfer tussen 0 en 1024) die wordt uitgelezen bij de potmeter. Deze waarde ga je vervolgens gebruiken om de snelheid van het knipperen te bepalen.

```
// Hier geven we aan welke stappen de Arduino moet afwerken in de sketch
void loop() {
  int snelheidsRegelaar = analogRead(potmeter);
```

Nu gaan we de waarde uit de vorige stap omrekenen. Als je de snelheid 0 zou gebruiken, dan knipperen de LED's namelijk niet. De snelheid 1024 gaat zo snel dat het bijna niet te zien is. Via de 'map' functie zet je de waarde tussen 0-1024 om naar een nieuwe waarde tussen de 50-500. Draai je de potmeter helemaal terug, dan wordt de waarde 50. Draai je de potmeter volledig open, dan wordt de waarde 500.

De waarde van 50 komt tijdens de sketch overeen met 50 milliseconden. De waarde 500 komt overeen met 500 milliseconden (oftewel ½ seconde). Door de waarde op deze plek te veranderen zal de potmeter tijdens het uitvoeren van de sketch anders reageren. Test dit na de tijd zelf maar eens.

```
int snelheid = map(snelheidsRegelaar, 0, 1024, 50, 500);
```

Het is nu tijd om de LED's te laten knipperen. Het knipperende effect krijg je door de LED aan te zetten (de digitale poort krijgt de waarde HIGH). Vervolgens pauzeer je even (delay). De LED blijft nu branden. De volgende stap is het uitzetten van de LED (de digitale poort krijgt de waarde LOW).

```
 digitalWrite(led1, HIGH);
   delay(snelheid);
 digitalWrite(led1, LOW);
```

Dit doe je voor alle LED's afzonderlijk. De volgorde in de sketch is ook de volgorde van knipperen in het uiteindelijke project. Je gaat daarom eerst led1 tot en met led5 laten knipperen, om vervolgens weer terug te gaan door led4 tot en met led1 te laten knipperen (je gaat dus heen- en weer).

Ook hier zou je een 'for loop' kunnen gebruiken. Om duidelijk te zien wat er gebeurt tijdens het uitvoeren van de sketch kies je daar nu niet voor.

```
  digitalWrite(led2, HIGH);
   delay(snelheid);
  digitalWrite(led2, LOW);

  digitalWrite(led3, HIGH);
   delay(snelheid);
  digitalWrite(led3, LOW);

  digitalWrite(led4, HIGH);
   delay(snelheid);
  digitalWrite(led4, LOW);

  digitalWrite(led5, HIGH);
   delay(snelheid);
  digitalWrite(led5, LOW);

  digitalWrite(led4, HIGH);
   delay(snelheid);
  digitalWrite(led4, LOW);

  digitalWrite(led3, HIGH);
   delay(snelheid);
  digitalWrite(led3, LOW);

  digitalWrite(led2, HIGH);
   delay(snelheid);
  digitalWrite(led2, LOW);
}
```

Het haakje } dient als afsluiting van de loop(). Je hebt deze namelijk ook geopend met een haakje {.

Arduino Knight Rider | uitvoeren van de sketch

Sluit nu de Arduino via de USB-kabel aan op de computer. Je zult zien dat het ON-lampje gaat branden. Nu kun je de sketch uploaden naar de Arduino via de upload-knop in de software. De RX/TX-lampjes gaan even knipperen, vervolgens zie je in de software staan dat de upload is geslaagd. De LED's zullen nu gaan knipperen!

Draai nu eens aan de potmeter. Je zult zien dat het effect vertraagt of versnelt. Dit komt doordat je de snelheid zelf bepaalt via de geprogrammeerde 'delay' in je sketch.

```
//      Er kleeft een nadeel aan de 'delay' functie. De Arduino doet op die momenten ook echt niets.
        Wanneer je aan de potmeter draait zal de Arduino eerst de hele loop uitvoeren (inclusief alle
        tijdrovende delays) om vervolgens pas de snelheid te wijzigen aan het begin van de loop. Voor
        dit project maakt het gelukkig weinig uit. Wil je dat de Arduino altijd paraat staat, dan zul je op
        een andere (lastiger) manier met tijd moeten werken. De millis () functie uit het volgende project
        (Arduino alarmsysteem) is daar een mooi voorbeeld van.
```

Project: Arduino alarmsysteem

In dit project ga je een eenvoudig alarmsysteem bouwen. Wanneer het contact wordt verbroken schakel je een alarm in. Je gaat in deze sketch met tijd werken door toepassing van de millis-functie.

Benodigde onderdelen:
- 1 Reedcontact
- 1 Magneet
- 1 Weerstand 10 kOhm (bruin/zwart/oranje)
- 1 Piëzo-zoemer
- 6 Draadbruggen
- 1 Breadboard

Arduino alarmsysteem | de schakeling

Op de volgende pagina vind je een duidelijk schema van de schakeling.

1. Plaats een draadbrug tussen de 5V-aansluiting van je Arduino en de rood gemarkeerde strook op het breadboard.
2. Plaats een draadbrug tussen de GND-aansluiting van je Arduino en de blauw gemarkeerde strook op het breadboard.

   // De eerste stap bij het bouwen van vrijwel iedere schakeling is het aansluiten van de voeding op het breadboard. De 5V- en GND-aansluiting van je Arduino dienen in dit project als voeding, deze worden doorverbonden naar het breadboard waardoor ook hier spanning op staat.

3. Plaats het reedcontact op het breadboard en plaats de magneet er vast tegenaan.
4. Plaats een draadbrug tussen de linkerkant van het reedcontact en de rood gemarkeerde strook op het breadboard.
5. Plaats een draadbrug tussen het pootje aan de rechterkant van het reedcontact en digitale I/O 2 op de Arduino

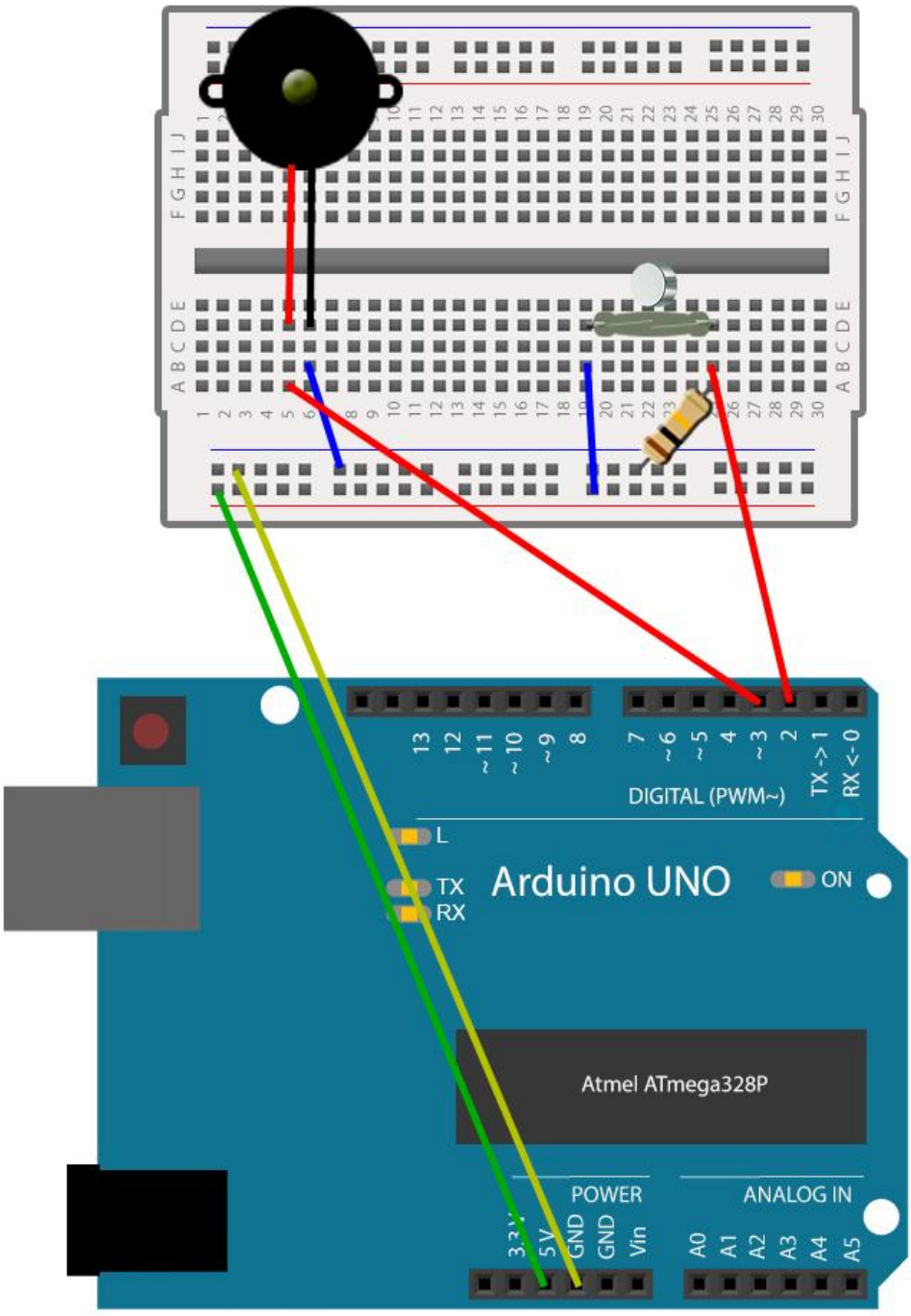
   // Je sluit het reedcontact via een draadbrug aan op de Arduino en niet op 5V omdat je de stand van het reedcontact wilt uitlezen via het board

6. Plaats een weerstand van 10 kOhm (bruin/zwart/oranje) tussen de rechterkant van het reedcontact en de blauwe GND-strook van het breadboard.

   // Wanneer je een digitale I/O als ingang gebruikt, dan kan er storing ontstaan wanneer er weinig tot geen spanning op deze ingang staat. Hierdoor kan je schakeling zich vreemd gaan gedragen. Als het contact is ingeschakeld loopt er spanning van 5V door het contact richting ingang 2, dit gaat prima. Voor momenten dat er geen contact wordt gemaakt wil je een veiligheid inbouwen die alle spanning tegenhoudt. Deze veiligheid noem je een ‘pull-down-weerstand’ en plaats je in de vorm van een 10 kOhm weerstand tussen GND en digitale aansluiting 2.

7. Plaats een piëzo-zoemer op het breadboard (of er naast) en steek de zwarte en rode draad in twee afzonderlijke stroken van het breadboard.
8. Plaats een draadbrug tussen de rode draad van de piëzo-zoemer en de digitale I/O 3 op de Arduino.
9. Plaats een draadbrug tussen de zwarte draad van de piëzo-zoemer en de blauw gemarkeerde strook op het breadboard.

   // Je sluit de rode draad van de piëzo-zoemer via een draadbrug aan op de Arduino en niet op 5V omdat je de zoemer wilt aansturen met de Arduino op basis van de input van het reedcontact.


DIGITAL (PWM~)
TX -> 1
RX <- 0
L
TX
RX
Arduino UNO
ON
Atmel ATmega328P
POWER
ANALOG IN
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5

Arduino alarmsysteem | de sketch

Je gaat nu de sketch schrijven om alle verschillende onderdelen van de hardware aan te sturen. Open hiervoor de Arduino software op je computer. Er opent vanzelf een lege sketch.

We gaan nu stap-voor-stap de verschillende onderdelen van de sketch bespreken.

Je hebt twee aansluitingen gebruikt in je opstelling. Om de sketch begrijpelijk te houden is het makkelijk om deze aansluitingen een naam te geven. Deze aansluiting verandert niet (het is een constante) en heeft een numerieke waarde (integer). De piëzo-zoemer is bijvoorbeeld aangesloten op aansluiting 3.

Je maakt hier ook een waarde aan om te toestand van de zoemer aan te geven.

```
// Benoem de noodzakelijke elementen van deze sketch
const int zoemer = 3;
const int reed = 2;
int zoemerWaarde = LOW;
```

Je gaat nu een waarde aanmaken waarin je opslaat wanneer de zoemer voor het laatst piepte. Deze waarde noem je een ‘unsigned long’. Met de waarde ‘long’ kunnen veel getallen worden opgeslagen (meer dan bij een ‘int’ het geval is). De toevoeging ‘unsigned’ maakt dat deze alleen positieve getallen opslaat, hierdoor is er ruimte voor nog meer getallen. Ter vergelijking; met een ‘int’ kun je maximaal 65 seconden opslaan voor het geheugen vol is. Een ‘unsigned long’ kan tot ca. 50 dagen opslaan. Dit maakt de ‘unsigned long’ waarde dus juist zo ideaal voor het werken met tijd.

```
unsigned long vorigeTijd = 0;
```

De volgende waarde is de interval. Je wilt namelijk dat de piëzo-module zoemt, even stil is, en dan weer gaat zoemen. De tijd hier tussenin sla je op als de interval. In dit geval houden we 1000 milliseconden aan (oftewel; een periode van 1 seconde).

```
const long interval = 1000;
```

Wat nu volgt is de setup(), hier geef je eenmalig aan wat de juiste instellingen voor deze sketch zijn.

Geef als eerste aan dat de aansluiting die je in de vorige stap hebt aangeduid als zoemer dient als uitgang (output) in deze sketch. Het reedcontact wordt gebruikt als ingang (input).

```
// Hier geven we eenmalig aan wat de instellingen voor deze sketch moeten zijn
void setup() {
  pinMode(zoemer, OUTPUT);
  pinMode(reed, INPUT);
}
```

De sketch komt nu aan bij het loop() gedeelte waar daadwerkelijke acties zich keer op keer herhalen.

Als eerste maak je hier een waarde aan genaamd ‘huidigeTijd’. Hiermee sla je de tijd in milliseconden sinds het starten van de loop (millis) op om te gebruiken als nulmeting tijdens de sketch. Daarnaast maak je een waarde aan genaamd ‘Reedwaarde’. Dit is de digitale waarde (aan of uit) die wordt uitgelezen bij het reedcontact. Hiermee bepaal je tijdens de sketch of de zoemer aan of uit moet.

```
// Hier geven we aan welke stappen de Arduino moet afwerken in de sketch
void loop() {
  unsigned long huidigeTijd = millis();
  int reedWaarde = digitalRead(reed);
```

Je begint nu met het creëren van een voorwaarde. Namelijk; als (if) de reedWaarde aan is ‘HIGH’. Als aan deze voorwaarde wordt voldaan, dan wordt er geen spanning gezet op de zoemer. Dit doe je door de digitale uitgang op ‘LOW’ te zetten. Je wil in dit geval niets doen omdat de magneet nog gewoon tegen het reedcontact aan ligt.

```
if(reedWaarde == HIGH){
  digitalWrite(zoemer, LOW);
 }
```

Als het reedcontact niet is ingeschakeld, dan is deze uit (else if). Het alarm moet dus afgaan. Daarvoor ga je rekenen. Als de huidige tijd min de vorige tijd groter of gelijk is aan de interval. Oftewel, is de intervaltijd sinds het starten van de loop verstreken? Als dit zo is, dan wil je de vorige tijd overschrijven met de huidige tijd. Je reset zo als het waarde de starttijd om de volgende loop weer opnieuw te beginnen en de gewenste interval tussen het zoemen te krijgen.

```
else if (huidigeTijd - vorigeTijd >= interval) {
    vorigeTijd = huidigeTijd;
```

Als de zoemer nu uit staat (LOW), dan moet de zoemer aan (HIGH). Is dit niet het geval, oftewel; de zoemer staat nu aan, dan moet de zoemer dadelijk uitgezet worden. Je voert de actie hier nog niet daadwerkelijk uit, je maakt enkel een waarde aan.

```
if (zoemerWaarde == LOW) {
  zoemerWaarde = HIGH;
} else {
  zoemerWaarde = LOW;
}
```

Haal de waarde uit de vorige stap op en geef deze waarde door aan de zoemer. Het komt er op neer dat je in deze stap de zoemer aan of uit zet waardoor deze gaat zoemen met daartussen de intervaltijd.

```
    digitalWrite(zoemer, zoemerWaarde);
  }
}
```

Het haakje } dient als afsluiting van de loop(). Je hebt deze namelijk ook geopend met een haakje {.

Arduino alarmsysteem| uitvoeren van de sketch

Sluit nu de Arduino via de USB-kabel aan op de computer. Je zult zien dat het ON-lampje gaat branden. Nu kun je de sketch uploaden naar de Arduino via de upload-knop in de software. De RX/TX-lampjes gaan even knipperen, vervolgens zie je in de software staan dat de upload is geslaagd. Het alarmsysteem is klaar voor gebruik!

Er gebeurt in eerste instantie niets. Maar haal de magneet nu eens weg. Je zult zien dat het reedcontact wordt verbroken en dat de zoemer wordt ingeschakeld door de Arduino.

Je hebt nu zelf een eenvoudig alarmsysteem gebouwd. Ditzelfde principe vind je in de praktijk vaak terug als deur- en of raamcontact in volwaardige alarmsystemen.

Project: Arduino nachtlampje

In dit project ga je een nachtlampje bouwen. Deze gaat automatisch branden zodra het donker wordt. Om zelf te bepalen wanneer jij het donker genoeg vindt, voeg je ook hier een instelling voor toe.

Benodigde onderdelen:
- 1 Witte LED
- 1 Weerstand 68 Ohm (blauw/grijs/zwart)
- 1 Lichtsensor (LDR)
- 1 Weerstand 10 kOhm (bruin/zwart/oranje)
- 1 Potmeter
- 6 Draadbruggen
- 1 Breadboard

Arduino nachtlampje | de schakeling

Op de volgende pagina vind je een duidelijk schema van de schakeling.

1. Plaats een draadbrug tussen de 5V-aansluiting van je Arduino en de rood gemarkeerde rij op het breadboard. Het maakt niet uit welke draadbrug je hiervoor kiest.
2. Plaats een draadbrug tussen de GND-aansluiting van je Arduino en de blauw gemarkeerde strook op het breadboard.

   // De eerste stap bij het bouwen van vrijwel iedere schakeling is het aansluiten van de voeding op het breadboard. De 5V- en GND-aansluiting van je Arduino dienen in dit project als voeding, deze worden doorverbonden naar het breadboard waardoor ook hier spanning op staat.

3. Plaats de witte LED op het breadboard, let daarbij goed op de polariteit (+/-) van de LED en verbind de pootjes tussen twee stroken op het breadboard.
4. Plaats een draadbrug tussen de anode (+) van de LED en digitale I/O 2 op de Arduino.

   // Je sluit de anode van de LED via een draadbrug aan op de Arduino en niet op 5V omdat je de LED wilt aansturen met de Arduino op basis van input (de lichtsensor).

5. Plaats een weerstand van 68 Ohm (blauw/grijs/zwart) tussen de kathode (-) van de LED en de blauwe GND-strook van het breadboard.

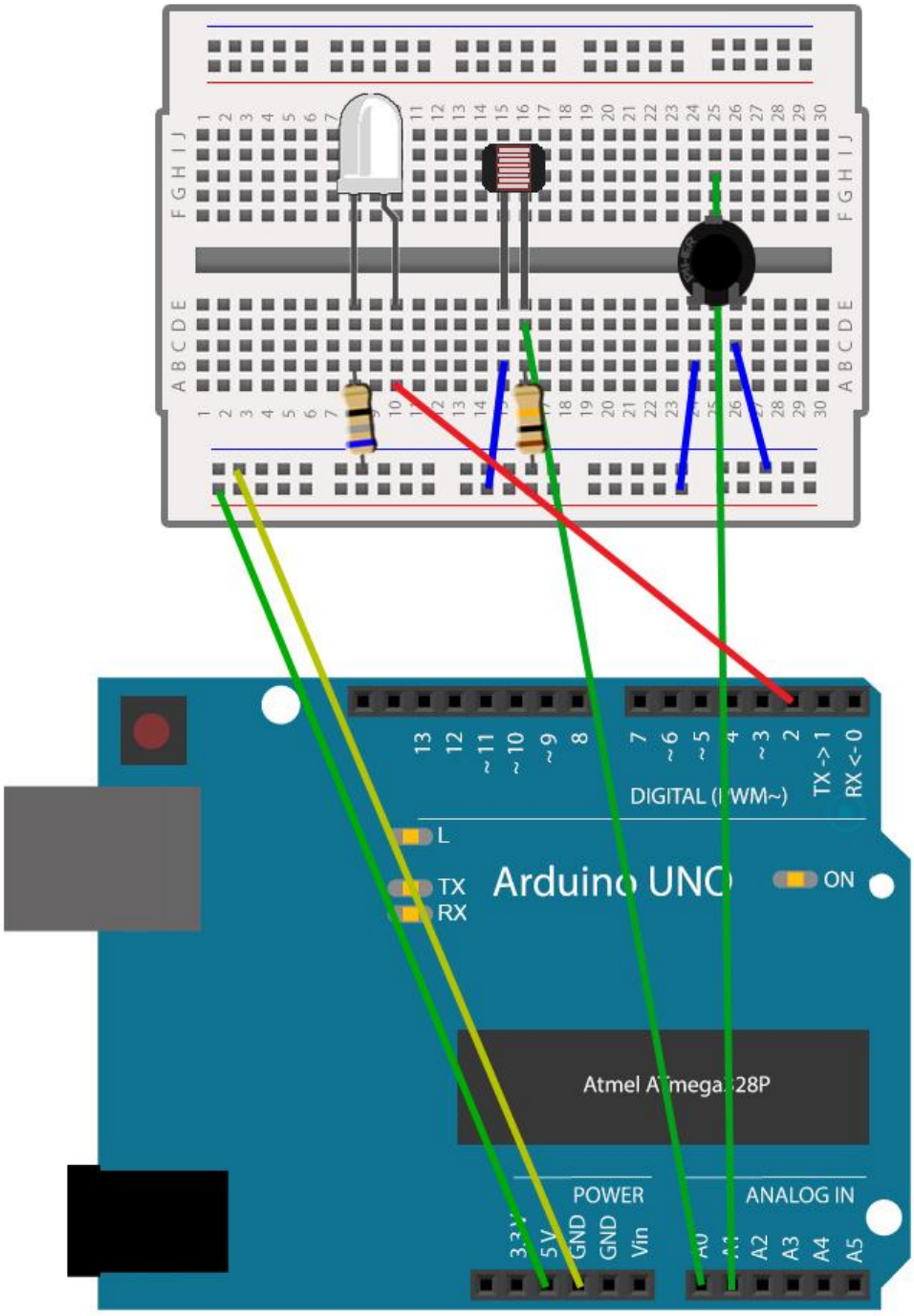
   // Deze LED heeft een voorschakelweerstand van 68 Ohm nodig om niet teveel spanning over de LED te laten lopen (deze is geschikt voor maximaal 3,6V).

6. Plaats de lichtsensor tussen twee stroken van het breadboard, polariteit speelt hier geen rol.
7. Plaats een draadbrug tussen de 5V van het breadboard en de linkerpin van de lichtsensor.
8. Plaats een weerstand van 10 kOhm (bruin/zwart/oranje) tussen de rechterpin van de lichtsensor en de GND-strook op het breadboard.
9. Plaats een draadbrug tussen de rechterpin van de lichtsensor en de A0 op je Arduino.

   // De lichtsensor gaat samen met de weerstand functioneren als spanningsdeler. Met deze spanningsdeler kun je de uitgangsspanning regelen en zo de bijbehorende waarde uitlezen vanaf de lichtsensor.

10. Plaats de potmeter over het midden van je breadboard. Zorg ervoor dat de zijde met twee pinnen aan de kant van de LED’s zitten. De enkele pin wordt aan de overzijde in het breadboard geplaatst.
11. Plaats een draadbrug tussen de linkerpin van de potmeter en de 5V van het breadboard.
12. Plaats een draadbrug tussen de rechterpin van de potmeter en de GND van het breadboard.
13. Plaats een draadbrug tussen de achterste pin van de potmeter en de A1van de Arduino.

// Een potmeter is een spanningsdeler, maar dan in één behuizing. Door te draaien aan de as wijzig je de uitgangsspannning. De uitgangsspanning loopt in dit geval richting de analoge A1-aansluiting van de Arduino.



Arduino nachtlampje | de sketch

Je gaat nu de sketch schrijven om alle verschillende onderdelen van de hardware aan te sturen. Open hiervoor de Arduino software op je computer. Er opent vanzelf een lege sketch.

We gaan nu stap-voor-stap de verschillende onderdelen van de sketch bespreken.

Je hebt drie aansluitingen gebruikt in je opstelling. Om de sketch begrijpelijk te houden is het makkelijk om deze aansluitingen een naam te geven. Deze aansluiting verandert niet (het is een constante) en heeft een numerieke waarde (integer). De LED is bijvoorbeeld aangesloten op aansluiting 2.

```
// Benoem de elementen die je in deze sketch wilt gebruiken
const int led = 2;
const int sensor = A0;
const int potmeter = A1;
```

Wat nu volgt is de setup(), hier geef je eenmalig aan wat de juiste instellingen voor deze sketch zijn. Met de code Serial.begin(9600); open je een seriële verbinding met de computer. Via deze verbinding kun je straks alle relevante waarden uitlezen op het beeldscherm van je computer. De toevoeging (9600) staat voor de snelheid (de baudrate) waarmee de communicatie verloopt en is een veelgebruikte waarde.

Verder geef je hier aan dat de aansluiting die je in de vorige stap hebt aangeduid als LED dient als uitgang (output) in deze sketch. De sensor en potmeter worden gebruikt als ingang (input).

```
// Hier geven we eenmalig aan wat de instellingen voor deze sketch moeten zijn
void setup() {
  Serial.begin(9600);
  pinMode(led, OUTPUT);
  pinMode(sensor, INPUT);
  pinMode(potmeter, INPUT);
}
```

De sketch komt nu aan bij het loop() gedeelte waar daadwerkelijke acties zich keer op keer herhalen.

Als eerste maak je hier een waarde aan genaamd ‘drempelWaarde’. Dit is de analoge waarde (tussen 0 en 1024) die wordt uitgelezen bij de uitgang van de potmeter. Deze waarde gebruik je om aan te geven of het donker genoeg is. Via de Serial.print code toon je deze waarden op het beeldscherm van je computer. Een waarde tussen “aanhalingstekens“ wordt letterlijk als tekst overgenomen. De “ | “ gebruik je hier alleen als scheidingsteken voor een betere leesbaarheid op de computer.

```
// Hier geven we aan welke stappen de Arduino moet afwerken in de sketch
void loop() {
  int drempelWaarde = analogRead(potmeter);
  Serial.print("Drempelwaarde: ");
  Serial.print(drempelWaarde);
  Serial.print(" | ");
```

Je maakt vervolgens een waarde aan genaamd ‘sensorWaarde’. Dit is de analoge waarde (tussen 0 en 1024) die wordt uitgelezen vanaf de lichtsensor. Ook deze waarde ga je straks weergeven op het beeldscherm van je computer.

```
int sensorWaarde = analogRead(sensor);
Serial.print("Sensorwaarde: ");
Serial.print(sensorWaarde);
Serial.print(" | ");
```

In de vorige stappen worden de invoerwaarden (de input) van je schakeling uitgelezen. Deze ga je nu gebruiken om een concrete actie uit te voeren (de output).

Je begint met het creëren van een voorwaarde. Namelijk; als (if) de gemeten sensorWaarde kleiner is dan de gemeten drempelWaarde. Als aan deze voorwaarde wordt voldaan, dan wordt er spanning gezet op de LED-aansluiting. Dit doe je door deze digitale uitgang op 'HIGH' te zetten.

```
if(sensorWaarde < drempelWaarde){
  digitalWrite(led, HIGH);
```

Om inzicht te krijgen in wat er gebeurt wil je ook op het beeldscherm laten zien dat aan de voorwaarde wordt voldaan. Dit doe je in dit geval door aan te geven dat het nachtlampje aan staat. Let op; je gebruikt hier Serial.println, de toevoeging ln(lees; el-en) geeft een regeleinde aan. Alles na deze Serial.println begint daarmee op een nieuwe regel.

```
  Serial.println("Nachtlampje Aan");
}
```

Je hebt nu een voorwaarde gecreëerd. Het kan dan ook gebeuren dat hier niet aan wordt voldaan. Dit doe je met een de voorwaarde 'else'. Als de sensorwaarde dus niet lager is dan de drempelWaarde, dan wil je geen spanning op de LED-uitgang. Dit doe je door de digitale uitgang op 'LOW' te zetten.

Ook hier wil je via de computer zien wat er gebeurt. Gebruik daarom ook hier de Serial.println toevoeging voor de seriële communicatie.

```
else {
  digitalWrite(led, LOW);
  Serial.println("Nachtlampje Uit");
}
```

Je bent nu aan het einde van de loop() beland. De Arduino gebruikt in deze sketch een aantal analoge waarden. Deze worden intern omgezet naar digitale waarden om goed verwerkt te kunnen worden. Dit verloopt via een ADC (Analog-to-Digital-Converter). Om deze ADC zijn werk goed te laten doen en om knipperen van de LED te voorkopen voegen we een vertraging toe. Als je deze op 1000 milliseconden zet wacht de sketch hier een seconde en begint daarna weer van voor af aan.

Het haakje } dient als afsluiting van de loop(). Je hebt deze namelijk ook geopend met een haakje {.

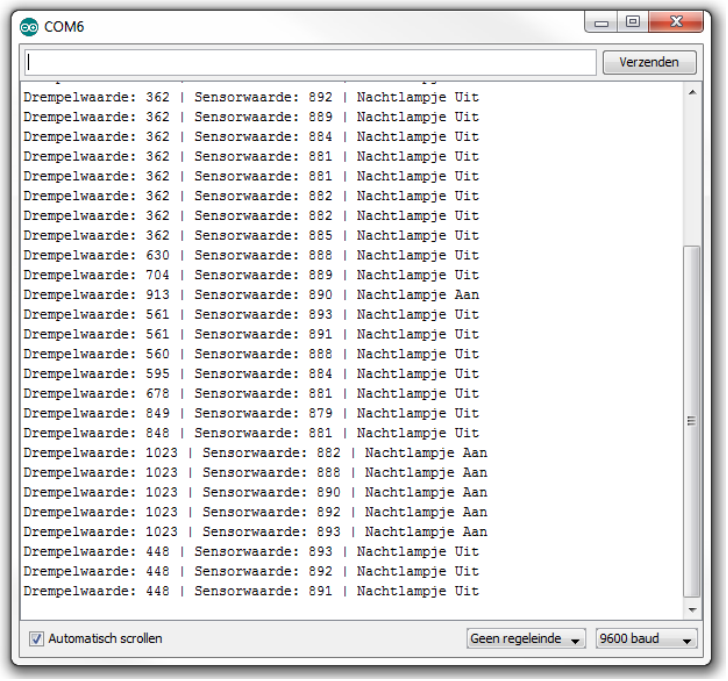
```
delay(1000);
  }
```

Arduino nachtlampje | uitvoeren van de sketch

Sluit nu de Arduino via de USB-kabel aan op de computer. Je zult zien dat het ON-lampje gaat branden. Nu kun je de sketch uploaden naar de Arduino via de upload-knop in de software. De RX/TX-lampjes gaan even knipperen, vervolgens zie je in de software staan dat de upload is geslaagd. Het nachtlampje is klaar voor gebruik!

Open als eerste de Serial Monitor via de Arduino software (Hulpmiddelen→ Seriële Monitor). Dit hulpmiddel laat zien wat er in je project gebeurt, dat hebben we immers zelf aangegeven in de sketch.

Draai nu eens aan de potmeter. Je zult zien dat de drempelwaarde verandert. Zodra de gemeten hoeveelheid licht op de lichtsensor onder de drempelwaarde zakt, gaat het nachtlampje aan. Test dit zelf ook maar eens door je hand boven de lichtsensor te houden. Je zult zien dat de waarde verandert.



Je hebt nu zelf een nachtlampje gebouwd. Je kunt hetzelfde principe natuurlijk ook gebruiken als schemerschakelaar in verschillende andere projecten!

Project: Arduino thermometer

In dit project ga je een thermometer bouwen waarbij je via drie gekleurde LED's kunt zien hoe warm het is. Wordt het te warm of koud, dan gaan er zelfs LED's knipperen.

Benodigde onderdelen:
- 1 Rode LED
- 1 Groene LED
- 1 Blauwe LED
- 2 Weerstanden150 Ohm (bruin/groen/bruin)
- 1 Weerstand 100 Ohm (bruin/zwart/bruin)
- 1 Temperatuursensor (LM35)
- 8 Draadbruggen
- 1 Breadboard

Arduino thermometer | de schakeling

Op de volgende pagina vind je een duidelijk schema van de schakeling.

1. Plaats een draadbrug tussen de 5V-aansluiting van je Arduino en de rood gemarkeerde rij op het breadboard. Het maakt niet uit welke draadbrug je hiervoor kiest.
2. Plaats een draadbrug tussen de GND-aansluiting van je Arduino en de blauw gemarkeerde strook op het breadboard.

   // De eerste stap bij het bouwen van vrijwel iedere schakeling is het aansluiten van de voeding op het breadboard. De 5V- en GND-aansluiting van je Arduino dienen in dit project als voeding, deze worden doorverbonden naar het breadboard waardoor ook hier spanning op staat.

3. Plaats de rode, groene en blauwe LED's op het breadboard, let daarbij goed op de polariteit (+/-) van de LED en verbind de pootjes tussen twee stroken op het breadboard.
4. Plaats een draadbrug tussen de anode (+) van de LED's en digitale I/O's op de Arduino. De blauwe LED verbind je met aansluiting 4, de groene LED met aansluiting 3 en de rode LED met aansluiting 2.

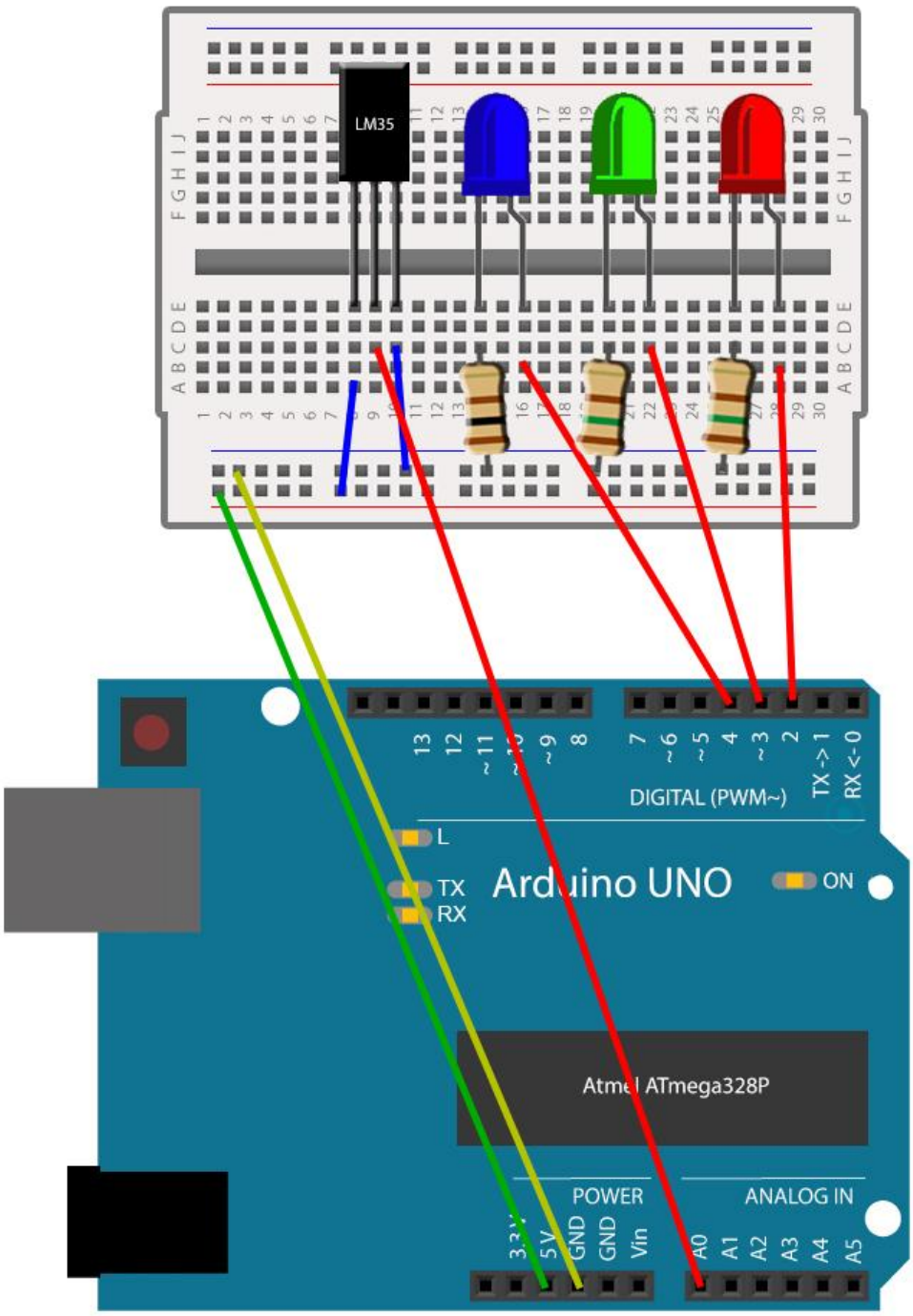
   // Je sluit de anode van de LED via een draadbrug aan op de Arduino en niet op 5V omdat je de LED wilt aansturen met de Arduino op basis van input (de temperatuursensor).

5. Plaats een weerstand van 150 Ohm (bruin/groen/bruin) tussen de kathode (-) van de groene LED en de blauwe GND-strook van het breadboard. Plaats ook een 150 Ohm weerstand tussen de kathode (-) van de rode LED en de blauwe GND-strook van het breadboard. Plaats nu een weerstand van 100 Ohm (bruin/zwart/bruin) tussen de kathode (-) van de blauwe LED en de blauwe GND-strook van het breadboard.

   // De LED's hebben een voorschakelweerstand nodig om niet teveel spanning over de LED te laten lopen (de rode en groene LED's zijn geschikt voor max.2,2V, de blauwe LED voor 3,1V).

6. Plaats de drie pootjes van de temperatuursensor in het breadboard. Vanaf de platte kant gezien zijn dit van links naar rechts gezien de plus-, V^{out} en de min.
7. Plaats een draadbrug tussen de plus van de sensor (links) en de rode 5V-strook van het breadboard.
8. Plaats een draadbrug tussen de min van de sensor (rechts) en de blauwe GND-strook van het breadboard.
9. Plaats een draadbrug tussen de V^{out} van de sensor (midden) en de analoge A0 aansluiting op de Arduino.

   // De uitgangsspanning van de sensor (V^{out}) verandert zodra de temperatuur verandert.


LM35
DIGITAL (PWM~)
TX -> 1
RX <- 0
L
TX
RX
Arduino UNO
ON
POWER
ANALOG IN
GND
GND
Vin
A0
A1
A2
A3
A4
A5

Arduino thermometer | de sketch

Je gaat nu de sketch schrijven om alle verschillende onderdelen van de hardware aan te sturen. Open hiervoor de Arduino software op je computer. Er opent vanzelf een lege sketch.

We gaan nu stap-voor-stap de verschillende onderdelen van de sketch bespreken.

Je hebt vier aansluitingen gebruikt in je opstelling, drie LED's en een sensor. Om de sketch begrijpelijk te houden is het makkelijk om deze aansluitingen een naam te geven. Deze aansluiting verandert niet (het is een constante) en heeft een numerieke waarde (integer).

Maak ook een element voor de gewenste temperatuur (basisTemperatuur). Hier kun je later in de sketch mee gaan rekenen. Het betreft hier een waarde genaamd 'float', dit is een waarde met decimalen. In dit geval 20.0 graden Celsius.

```
// Benoem de elementen die je in deze sketch wilt gebruiken
const int rodeLed = 2;
const int groeneLed = 3;
const int blauweLed = 4;
const int sensor = A0;
const float basisTemperatuur = 20.0;
```

Wat nu volgt is de setup(), hier geef je eenmalig aan wat de juiste instellingen voor deze sketch zijn. Met de code Serial.begin(9600); open je een seriële verbinding met de computer. Via deze verbinding kun je straks alle relevante waarden uitlezen op het beeldscherm van je computer.

Verder geef je hier aan dat de LED-aansluitingen (pin 2 tot en met 4) die je in de vorige stap hebt aangeduid dienen als uitgang (output) in deze sketch. De hier gebruikte schrijfwijze noem je een 'for loop'. Wanneer je meerdere I/O's tegelijkertijd wilt benoemen kun je met deze schrijfwijze wat tekst (en dus ruimte/geheugen) besparen. Standaard zijn deze LED's uit (LOW).

```
// Hier geven we eenmalig aan wat de instellingen voor deze sketch moeten zijn
void setup() {
  Serial.begin(9600);
  for(int pinNumber = 2; pinNumber <5; pinNumber++){
    pinMode(pinNumber,OUTPUT);
    digitalWrite(pinNumber, LOW);
  }
 }
```

De sketch komt nu aan bij het loop() gedeelte waar daadwerkelijke acties zich keer op keer herhalen.

Als eerste maak je hier een waarde aan genaamd 'sensorWaarde'. Dit is de analoge waarde (tussen 0 en 1024) die wordt uitgelezen bij de uitgang van de sensor. Deze waarde gebruik je verderop om de temperatuur te berekenen. Via de Serial.print code toon je deze waarden op het beeldscherm van je computer. Een waarde tussen "aanhalingstekens" wordt letterlijk als tekst overgenomen. De " | " gebruik je hier alleen voor een betere leesbaarheid op de computer.

```
// Hier geven we aan welke stappen de Arduino moet afwerken in de sketch
void loop() {
  int sensorWaarde = analogRead(sensor);
  Serial.print("Sensorwaarde: ");
  Serial.print(sensorWaarde);
  Serial.print(" | ");
```

Je gaat nu een kleine berekening uitvoeren. Je weet namelijk welke waarde er wordt uitgelezen op de sensor (een waarde tussen 0 en 1024), maar niet wat de bijbehorende spanning is (een waarde tussen 0 en 5 volt). In de sketch reken je dat als volgt om. Toon ook hier de waarde weer via de seriële verbinding op de computer.

```
float voltage = (sensorWaarde/1024.0) *5.0;
Serial.print("Volt: ");
Serial.print(voltage);
Serial.print(" | ");
```

De volgende stap is ook weer een berekening. Met behulp van de spanning kun je namelijk de temperatuur berekenen (het is een decimale waarde, dus een float). De temperatuursensor kent een lineair verband tussen de temperatuur en spanning; voor elke graad wordt de spanning met 10mV verhoogd. Ook deze waarde ga je weergeven op de computer.

```
float temperatuur = voltage * 100;
Serial.print("graden Celcius: ");
Serial.println(temperatuur);
```

Ga nu een voorwaarde creëren; als de (berekende) temperatuur lager is dan de basisTemperatuur -2. Dit komt dus neer op een temperatuur lager dan 18 graden Celsius. Wordt aan deze voorwaarde voldaan, dan moet de blauwe LED gaan knipperen, de andere LED's blijven uit. Je doet dit door de aansluiting van de blauwe LED op HIGH te zetten en – na een delay van ½ seconde – weer op LOW.

```
if(temperatuur < basisTemperatuur-2){
  digitalWrite(blauweLed, HIGH);
  delay(500);
  digitalWrite(blauweLed, LOW);
  digitalWrite(groeneLed, LOW);
  digitalWrite(rodeLed, LOW);
}
```

Maak nu nog een aantal vergelijkbare voorwaarden en laat de bijbehorende LED's knipperen. De eerste voorwaarde betekent 'als de temperatuur hoger of gelijk is aan 18 graden én lager dan 20 graden. De karakters && betekenen 'én' in de Arduino programmeertaal.

```
else if(temperatuur >= basisTemperatuur-2 && temperatuur < basisTemperatuur){
   digitalWrite(blauweLed, HIGH);
   digitalWrite(groeneLed, LOW);
   digitalWrite(rodeLed, LOW);
}
else if(temperatuur >= basisTemperatuur && temperatuur < basisTemperatuur+2){
   digitalWrite(rodeLed, LOW);
   digitalWrite(groeneLed, HIGH);
   digitalWrite(blauweLed, LOW);
}
else if(temperatuur >= basisTemperatuur+2 && temperatuur < basisTemperatuur+4){
   digitalWrite(rodeLed, HIGH);
   digitalWrite(groeneLed, LOW);
   digitalWrite(blauweLed, LOW);
}
```

Creëer nog een laatste voorwaarde. Als de temperatuur hoger is dan 24 graden Celsius. Wordt hier aan voldaan, dan moet de rode LED gaan knipperen.

```
else if(temperatuur > basisTemperatuur+4){
   digitalWrite(blauweLed, LOW);
   digitalWrite(groeneLed, LOW);
   digitalWrite(rodeLed, HIGH);
   delay(500);
   digitalWrite(rodeLed, LOW);
}
```

Je bent nu aan het einde van de loop() beland. De Arduino gebruikt in deze sketch een aantal analoge waarden. Deze worden intern omgezet naar digitale waarden om goed verwerkt te kunnen worden. Dit verloopt via een ADC (Analog-to-Digital-Converter). Om deze ADC zijn werk goed te laten doen en om knipperen van de LED te voorkopen voegen we een vertraging toe. Als je deze op 500 milliseconden zet wacht de sketch hier een ½ seconde en begint daarna weer van voor af aan.

Het haakje } dient als afsluiting van de loop(). Je hebt deze namelijk ook geopend met een haakje {.

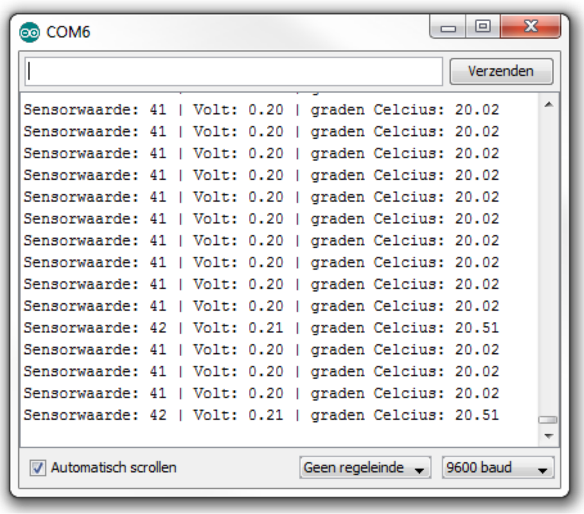
```
delay(500);
}
```

Arduino thermometer| uitvoeren van de sketch

Sluit nu de Arduino via de USB-kabel aan op de computer. Je zult zien dat het ON-lampje gaat branden. Nu kun je de sketch uploaden naar de Arduino via de upload-knop in de software. De RX/TX-lampjes gaan even knipperen, vervolgens zie je in de software staan dat de upload is geslaagd. De thermometer is klaar voor gebruik!

Open als eerste de Serial Monitor via de Arduino software (Hulpmiddelen→ Seriële Monitor). Dit hulpmiddel laat zien wat er in je project gebeurt, dat hebben we immers zelf aangegeven in de sketch.

Kijk nu eens naar de gemeten temperatuur, je zult zien dat de bijbehorende LED op het breadboard gaat branden. Test dit zelf ook maar eens door je vingers voorzichtig om de sensor te plaatsen. Je zult zien dat de waarde verandert.



Je hebt nu zelf een thermometer gebouwd. Je kunt hetzelfde principe natuurlijk ook gebruiken om temperatuur te meten in verschillende andere projecten!

Uitbreidingen voor Arduino

Met de onderdelen uit deze starterkit maak je op vrij eenvoudige wijze kennis met Arduino. Er komt echter een moment dat je meer wilt. Dit kun je oplossen door zelf alle benodigde onderdelen te kiezen. Een andere optie is het toevoegen van een Arduino Shield. Een shield is een soort board wat je bovenop de Arduino plaatst en waarmee je functionaliteit toevoegt.

We hebben een aantal populaire uitbreidingen voor je Arduino op een rijtje gezet.

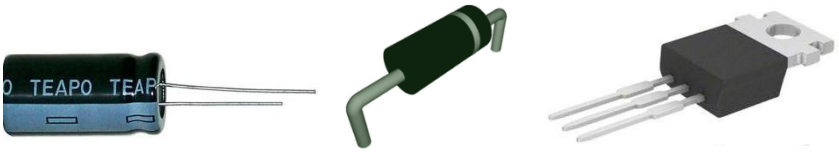
Ethernet/WiFi-shield
Je kunt heel veel interessante projecten bouwen zonder verbinding met het internet. Maar in sommige gevallen is het handig om het project toch van afstand te kunnen benaderen. Het toevoegen van een Ethernet of WiFi-shield biedt de oplossing in dit geval.

Relaiskaart
Wanneer je apparatuur op 230V/AC wilt aansturen dan is een relais bijna onvermijdelijk (je Arduino board levert namelijk slechts 5V=/40mA). Er zijn relaiskaarten verkrijgbaar die je vrij eenvoudig op een uitgang van de Arduino kunt aansluiten. Deze relaiskaarten zijn speciaal ontworpen voor boards als de Arduino.

Servo/DC-motor
Met een servo of DC-motor kun je beweging toevoegen aan je Arduino-projecten. Een servo heeft hierbij een beperkte uitslag (bijvoorbeeld te gebruiken om een luik te openen). Een DC-motor draait volledig rond en kan dit op verschillende toerentallen (bijvoorbeeld te gebruiken als aandrijving).

Display
Je hebt bij de projecten uit deze starterkit een aantal keer de serial monitor gebruikt. In sommige gevallen wil je juist de output weergeven op een klein display. Dit kan een klein display zijn met één kleur, maar ook volledige OLED-displays zijn mogelijk.

Losse componenten
De tot nu toe genoemde uitbreidingen zijn eenvoudig te gebruiken in combinatie met een Arduino. Toch heb je vaak nog ondersteunende componenten nodig om alles goed te laten werken. De weerstanden uit deze starterkit hebben namelijk ook zo hun functie. Bij gebruik van een servo heb je bijvoorbeeld een condensator nodig om spanningspieken op te vangen.



Voor ieder project zijn er zo wel specifieke componenten te bedenken die van belang zijn. Wanneer je het leuk vindt om je verder te verdiepen in elektronica, dan is dit natuurlijk een goede reden!

Tot slot

Je bent nu aan het einde van deze handleiding beland. Je weet nu genoeg van Arduino om zelf aan de slag te gaan. Hopelijk hebben we je iets kunnen leren. Maar belangrijker nog; hopelijk heb je inspiratie gekregen om alles in de praktijk te brengen in je eigen projecten!

Heb je aanvullingen en/of opmerkingen voor ons? Wij horen het graag van je!

https://www.facebook.com/Conrad.nl

https://twitter.com/conradnl

https://www.conrad.nl/contact

Conrad Electronic Benelux B.V.
www.conrad.nl