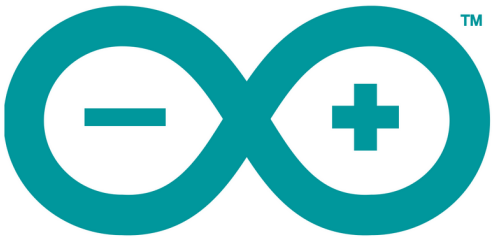
microControllers
programmeren
met ARDUINO



Frans Killian

Frans Killian V 1.0 juni 2015

Voorwoord

Welkom in de wereld van de moderne digitale elektronica!
Microcontrollers zijn overal. Elektronische schakelingen van vandaag zijn bijna altijd opgebouwd rond een microcontroller. Een moderne auto heeft minstens 10 microcontrollers aan boord.

Zelf microcontrollers programmeren kan bij het technasiumvak O&O, maar ook voor NLT of natuurkunde is het een goede manier om met fysische informatica bezig te zijn.
Het systeembord dat in de jaren 1980 door CMA werd geïntroduceerd is misschien wel achterhaald. Een simpele microcontroller van een paar euro kan heel veel meer dan tientallen systeemborden samen. Het is bovendien veel eenvoudiger, leuker en goedkoper.
Het is voor een ontwerper niet meer nodig om met discrete poortschakelingen te werken.
Als je wilt leren fietsen hoef je toch ook niet te weten hoe het rubber van de fietsbanden wordt gevulkaniseerd?

In de jaren 1980 heb ik een aantal instrumenten ontworpen en gebouwd met behulp van discrete poortschakelingen. Vele uren ontwerpen, vele tientallen TTL IC's met OR, AND, NOR, NAND en XOR poorten, binaire tellers, shift registers, flip-flops, converters, multiplexers, drivers enzovoort, waarvoor prints moesten worden ontworpen om de honderden pootjes op de juiste manier met elkaar te verbinden. Wat een puzzels!
Microcontrollers bestonden toen al wel, maar de programma's moesten in de dure chips worden gebrand (eenmalig) met dure hardware, dure software en moeilijke programmeertalen.
Nu is dat allemaal anders. Sinds een aantal jaren ontwerp ik de meeste instrumenten die ik op school gebruik met het Arduino-platform.

Het Arduino materiaal is open-source. Arduino gebruikt een eenvoudige programmeertaal. Sinds de introductie in 2005 is de populariteit almaar toegenomen. Op het internet is dan ook heel veel te vinden.
De meeste Arduino blogs, boeken en tutorials gaan uit van projecten en gaan er bovendien vaak van uit dat je al wat ervaring hebt met elektronica. Helaas is er maar weinig te vinden in het Nederlands.

Met deze methode leer je stap voor stap een microcontroller te programmeren met een Arduino UNO. In het Nederlands.
Steeds als er iets nieuws is geleerd, wordt het toegepast in een opdracht.
Er is aandacht voor het vertalen van ‘menselijke’ ontwerpeisen via logische stappen naar een werkend programma en prototype.
Om goed te kunnen begrijpen hoe sensoren werken wordt een klein beetje elektronica basiskennis zoals de wet van ohm en de spanningsdeler behandeld.
Ook is er aandacht voor het tekenen en lezen van elektronische schema’s. Dit zijn belangrijke vaardigheden die bij veel andere bronnen niet of nauwelijks aan de orde komen. Een Fritzing breadboard diagram ziet er misschien wel leuk uit, maar een standaard schema is vele malen overzichtelijker. Zeker als de schakeling wat ingewikkelder wordt.

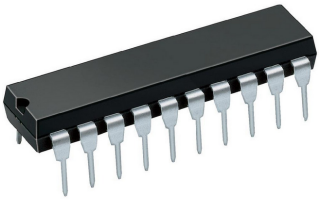
Frans Killian

Inhoud

1 Microcontrollers

1.1 Een microcontroller, wat is dat?

Een microcontroller is één enkele elektronische chip die complexe taken kan uitvoeren aan de hand van een programma wat in de chip is opgeslagen. Op de microcontroller-chip zijn meerdere onderdelen samengebracht, zoals een microprocessor, een RAM-geheugen, een A/D converter, een klok en een programma-geheugen. Eigenlijk is de microcontroller een heel klein computertje op één enkele chip.



Met zijn metalen contactpinnen staat de microcontroller in contact met de buitenwereld.
Een aantal van die contactpinnen zijn ingangen, waarop je sensoren kunt aansluiten zoals een lichtsensor, temperatuursensor, druksensor, bewegingssensor, een draaiknop, een schakelaartje, een IRsensor, GPS-ontvanger enzovoort. Met sensoren kan de microcontroller de buitenwereld waarnemen.
In het programma staat beschreven wat de microcontroller moet doen. Zo kan de microcontroller via de uitgangscontacten allerlei dingen besturen. Op de uitgangen kun je dingen aansluiten als LED's, lampen, een pomp, een display, een verwarmingsapparaat, een luidsprekertje, een motor, een IR-led, een robot-arm, een relais enzovoort. Dit noem je actuatoren.

1.2 De wasmachine

Een mooi voorbeeld van een toepassing van een microcontroller is de wasmachine.
Op de ingangen van deze microcontroller staan de bedieningsknoppen, zodat de microcontroller weet wat je met de wasmachine wilt doen. Op de ingangen staan ook de sensoren, zoals een druksensor die de microcontroller laat weten hoe hoog het water in de trommel staat, een bewegingssensor zodat de microcontroller het centrifugeren kan stoppen als de wasmachine te hard trilt, een temperatuursensor waarmee de microcontroller meet of hij het verwarmingselement moet aan- of uitzetten en een lichtsensor waarmee de microcontroller 'kijkt' hoe vuil het water is.
Op de uitgangen staan de elektrische kraan waarmee de microcontroller water in de trommel kan laten lopen, het verwarmingselement om het water te verwarmen, de motor waarmee de trommel kan draaien, de pomp waarmee vuil water wordt weggepompt, een display en een elektrisch slot wat ervoor zorgt dat je de wasmachine niet kan openen tijdens het wassen.

Het programma in de microcontroller laat alles mooi samenwerken zodat je wasgoed schoon wordt. In het programma is ook een klok geprogrammeerd. Via het display laat de microcontroller zien hoe lang het wassen nog duurt, welke temperatuur je hebt ingesteld enzovoort.

1.3 Microcontrollers zijn overal!

Microcontrollers vindt je overal. Bijna elk elektrisch apparaat heeft één of meer microcontrollers. De wasmachine, een horloge, klok of wekker, een thermostaat, de afstandbediening van de televisie, telefoons, of een wenskaart die 'happy birthday' speelt als je hem open vouwt. Een moderne auto heeft minstens 10 microcontrollers aan boord. Zelfs in een OV-chipkaart zit een microcontrollertje, zo groot als een suikerkorreltje.

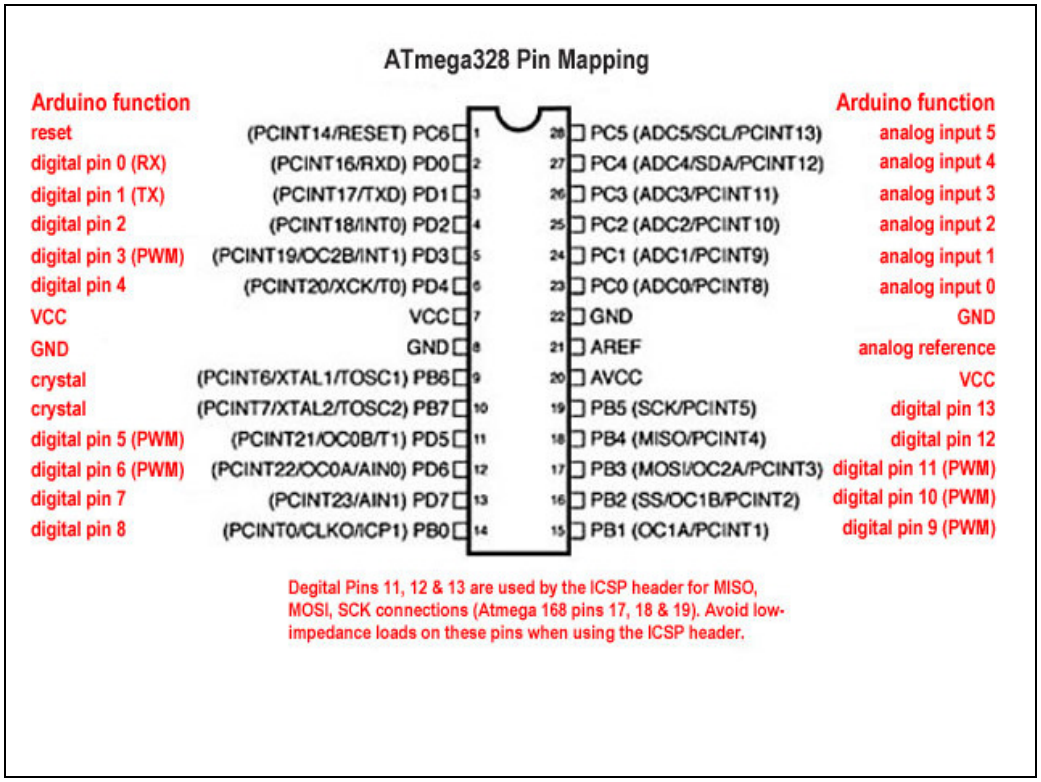
Vragen en opdrachten

1. Tel het aantal microcontrollers in je slaapkamer.
2. En hoeveel microcontrollers zijn er (ongeveer) in het hele huis?
3. Google op 'microcontroller' en bekijk de afbeeldingen.
4. Beschrijf wat een microcontroller in een kamerthermostaat allemaal moet doen.
5. Wat is het verschil tussen een microprocessor en een microcontroller?
6. Laat een oude papieren OV-chipkaart een tijdje weken in water zodat je het papier kan verwijderen. Je houd dan een dun plastic folie over waarin je de microcontroller kunt zien. De metalen banen die rondlopen zijn de antenne.

Als je zelf wel eens technisch ontwerpt is het erg handig als je ook zelf een microcontroller kan programmeren. Deze cursus is bedoeld om je hierbij op weg te helpen.

1.4 De ATmega328

De microcontroller die je zelf gaat programmeren en gebruiken is de ATmega328. Deze chip heeft 28 pootjes. Zes van die pootjes zijn analoge ingangen. Hierop kun je sensoren aansluiten.
De ATmega328 heeft 14 digitale in/uitgangen. Je kunt in het programma voor elk van deze pootjes aangeven of het een ingang of een uitgang moet zijn. Met een digitale uitgang kan de microcontroller iets aan en uit zetten.
Hieronder zie je een 'plattegrond' van de in- en uitgangen van de ATmega328

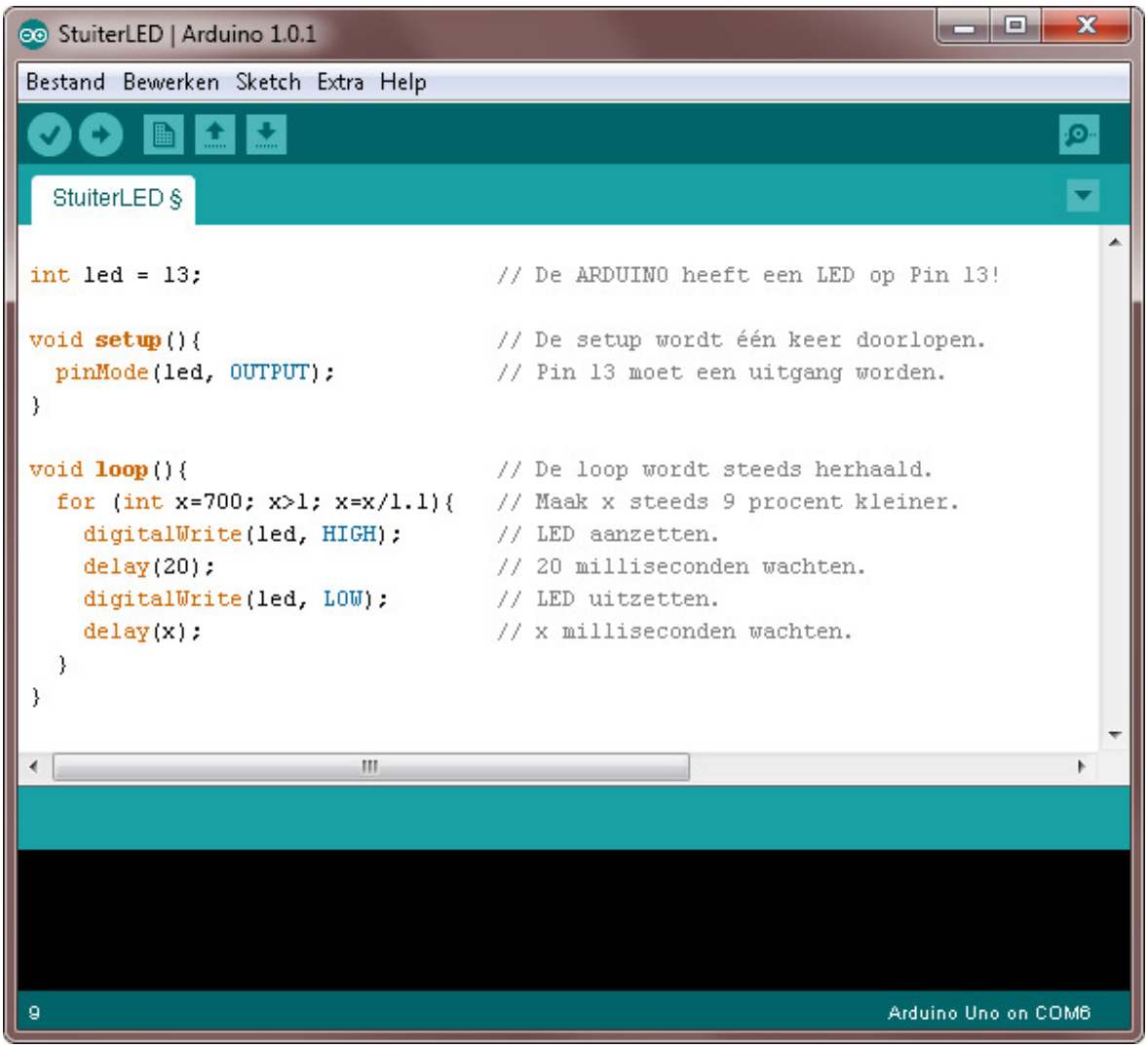


1.5 De Integrated Development Environment (IDE)

Een programma zoals dat in een microcontroller zit bestaat uit binaire code. Dat zijn alleen maar enen en nullen. Een mens kan geen programma schrijven in binaire code. Om die binaire code te maken heb je een speciaal computerprogramma nodig, een software-ontwikkelomgeving of IDE. Met deze software kun je een programma voor een microcontroller schrijven in een programmeertaal die voor een mens (de programmeur) te begrijpen is. De IDE heeft een editor om het programma te schrijven, een debugger om fouten in het programma op te zoeken en een compiler die het door jou geschreven programma vertaalt in de binaire code die in de microcontroller moet worden geladen.
Naast de IDE software heb je ook hardware nodig waarmee je de code vanuit de IDE in het geheugen van de microcontroller kunt laden.

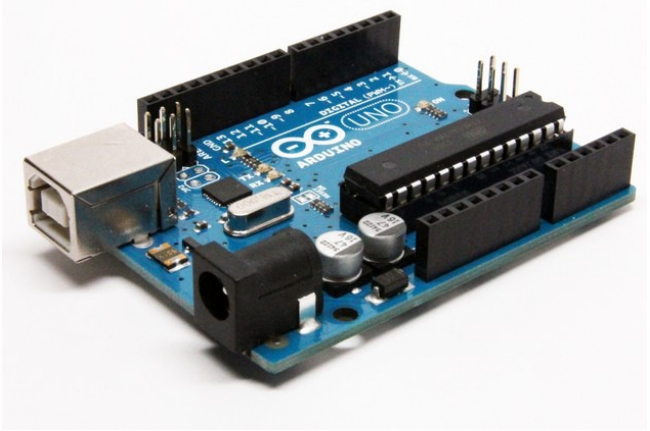
1.6 Arduino

Arduino maakt sinds 2005 een IDE en de benodigde hardware om microcontrollers te programmeren en om er prototypes mee te bouwen. Het uitgangspunt van de Arduino ontwerpers was dat iedereen het moest kunnen gebruiken. Kunstenaars, knutselaars, ingenieurs, studenten, scholieren en hobbyisten.
De Arduino hardware is open-source en het Arduino IDE programma is gratis. Je kunt het programma downloaden van http://www.arduino.cc .



Een eenvoudig programma in de Arduino IDE

De Arduino UNO hardware bestaat uit een board waarop de microcontroller in een voetje geprikt zit. De in- en uitgangen van de microcontroller zijn met heel kleine stekkertjes bereikbaar, zodat je gemakkelijk zonder te solderen een prototype kan bouwen en uitproberen. Verder zit er elektronica op het board om het programma van de IDE naar de ATmega328 chip te kunnen sturen. Het Arduino board sluit je aan op een USB poort van je computer.



Het Arduino UNO board

Als je op de groene pijl  van de IDE klikt wordt het programma in de editor gecontroleerd op fouten en als er geen fouten in zitten wordt het gecompileerd (omgezet in binaire code voor de microcontroller) en via de USB kabel ge-upload naar de ATmega328 op het Arduino board.

Het geheugen van de ATmega328 chip is een flash-geheugen. Dat betekent dat je zo vaak als je wilt een ander programma kunt uploaden.
Een Arduino UNO board inclusief de microcontroller kost ongeveer € 25,00. De losse ATmega328 microcontroller kost ongeveer € 5,00.
Als de microcontroller geprogrammeerd is kun je de USB kabel er uit halen en kan de Arduino zelfstandig zonder computer het programma uitvoeren. Je hebt dan natuurlijk wel een voeding van 9 volt nodig, of een USB voeding van 5 volt.
Je kunt zelfs de ATmega328 chip voorzichtig uit het voetje halen en los van het Arduino board gebruiken. Hoe dat moet wordt beschreven in hoofdstuk 7.2.

2 Programmeren

2.1 De Arduino sketch

Een programmeertaal moet je leren, net als duits of engels. Als je je niet goed aan de regels houdt begrijpt de IDE-editor niet wat je bedoelt en werkt het programma verkeerd of helemaal niet. Ook als je een hoofdletter of een komma vergeet kan dat grote gevolgen hebben. Alleen de spaties in de editor zijn niet zo belanglijk.
De Arduino IDE heeft een uitgebreide help functie. Via Help/Reference kun je alle belangrijke onderdelen van de programmeertaal (de syntax) vinden.
De Arduino programmeertaal is redelijk simpel. Een programma wat geschreven is in de Arduino programmeertaal noem je een sketch. Via de IDE Bestand/Voorbeelden kun je veel voorbeelden van sketches vinden. Op het internet zijn ook heel veel voorbeelden te vinden.

2.2 De setup- en loop-functies

Een sketch moet altijd een setup- en een loop-functie hebben.
In de setup functie van de sketch worden de instellingen van pinnen en dergelijke gemaakt. De setup wordt maar één keer doorlopen, direct nadat je de microcontroller hebt aangezet.
De loop wordt steeds herhaald, zolang de microcontroller aan staat.
Een minimale sketch ziet er dus zo uit:

```
void setup(){
}

void loop(){
}
```

Deze sketch doet niks want er staan nog geen opdrachten in.
Tussen de accoladen, { en } worden de programmaregels met opdrachten geschreven. Voor elke { staat verderop in de sketch een }. Een sketch bevat dus altijd evenveel { als }. De sluit-accolade staat recht onder de eerste letter van de functie. Dit is niet noodzakelijk, maar je kunt dan goed zien waar de accolade bij hoort.

2.3 Meer functies

Meerdere opdrachten die bij elkaar horen en samen één functie hebben kun je tussen accoladen zetten en een naam geven. Je kan zelf zoveel functies programmeren als je wilt. Dit is erg handig als je in het programma regelmatig dezelfde reeks dingen moet doen, zoals sensoren uitlezen, berekeningen maken of iets naar een display schrijven.
Met de opdracht `berekenX();` laat je het programma naar de functie `void berekenX(){…}` springen en worden de opdrachten tussen de accoladen van die functie één keer uitgevoerd. Daarna gaat het programma verder waar het gebleven was.

2.4 // Commentaarregel

Een // (dubbele slash) en alles wat er achter staat op die regel wordt genegeerd door de compiler. Hiermee kun je je sketch voorzien van commentaar. Kijk maar naar het voorbeeld op bladzijde 9. Dit is erg handig voor jezelf, maar ook voor andere programmeurs die jouw sketch willen gebruiken. En voor deze cursus! Gebruik altijd commentaarregels in je sketch!!!

2.5 /* Commentaarblok */

Als je meerdere regels commentaar hebt kun je iedere regel beginnen met een dubbele slash, maar je kunt ook de eerste regel beginnen met /* , en de laatste regel eindigen met */.
Je kunt ook delen van een programma tijdelijk uitzetten door er een commentaarregel of commentaarblok van te maken. Zo kun je binnen één sketch verschillende dingen uitproberen.

2.6 pinMode

De 14 digitale pinnen van de Arduino kunnen ingang of uitgang zijn. Voordat je de pinnen kan gebruiken moet je eerst aangeven of de betreffende pinnen ingangen of uitgangen moeten zijn. Dit doe je met de pinMode opdracht. De pinMode opdrachten worden meestal in de setup-functie gezet:

```
pinMode(6, OUTPUT);           // Maak van pin 6 een uitgang.
pinMode(LED, OUTPUT);         // Maak van pin LED een uitgang.
pinMode(8, INPUT);            // Maak van pin 8 een ingang.
pinMode(x, OUTPUT);           // Maak van pin x een uitgang.
pinMode(schakelaar, INPUT);   // Maak van pin schakelaar een ingang.
```

Je kunt een pin zelf elke naam geven. Als je een pin '`LED`' noemt, of '`groeneLED`' of '`plustoets`' of '`deurslot`', dan kun je in de sketch meteen zien wat de ingang of uitgang doet.
Als je een pin een naam geeft moet je die naam altijd definiëren aan het begin van de sketch. Omdat de pinnen altijd een geheel getal zijn definieer je ze als integer. Een integer (int) is een geheel getal.

```
int A=2;          // int betekent: geheel getal.
int LED=12;       // LED is een geheel getal en is 12.
int groeneLED=4;  // groeneLED is geheel getal 4.
int plustoets=5;  // de plustoets wordt aangesloten op pin 5.
int deurslot=10;  // het deurslot op uitgang 10.
```

De getallen A, LED, groeneLED, plustoets en deurslot in de voorbeelden noem je variabelen.

2.7 Variabelen

Een variabele is een waarde die in een geheugenplaats wordt opgeslagen. Je kunt een variabele zelf een naam geven, bijvoorbeeld: a, x, LED, motor2, druktoets, RODElamp, enz. De naam mag geen spaties of leestekens bevatten en mag ook niet beginnen met een cijfer.
Er zijn verschillende soorten variabelen.
De soorten variabelen die je het meest gebruikt zijn: **`int`**, **`long`** en **`float`**.
Je kunt een variabele elke waarde geven die je wilt, zolang het maar binnen het bereik van die soort variabele is. In het programma kun je variabelen veranderen met sensoren of met berekeningen.

Vragen en opdrachten

7. Zoek uit wat de variabelen **`int`**, **`long`** en **`float`** precies zijn.
8. Waarom kun je voor het definiëren van pinnen beter **`int`** gebruiken, en niet **`float`**?

2.8 digitalWrite

Met de opdracht digitalWrite kun je een digitale uitgang hoog of laag maken en dus iets aan of uit zetten:

```
digitalWrite(6, HIGH);        // Maak uitgang 6 hoog (5 volt).
digitalWrite(6, LOW);         // Maak uitgang 6 laag (0 volt).
digitalWrite(LED, HIGH);      // Maak uitgang LED hoog (LED aan zetten).
digitalWrite(Motor2, HIGH);   // Zet motor2 aan.
digitalWrite(deurslot, LOW); // Doe het deurslot dicht.
```

Let op de hoofdletters en kleine letters. DigitalWrite, digitalwrite of Digitalwrite werkt niet!
Elke opdracht moet altijd worden afgesloten met een puntkomma. Dan weet de compiler dat er een nieuwe opdracht begint.

2.9 delay

Delay betekent wacht. Achter de opdracht delay staat tussen haakjes hoe lang het programma moet wachten in milliseconden. Één milliseconde is een duizendste van een seconde.

```
delay(1000);      // Wacht een seconde.
delay(1800000);   // Wacht een half uur.
delay(x);         // Wacht x milliseconden.
delay(p*1000);    // Wacht p seconden.
```

Let op! Als x in de opdracht **`delay`**`(x);` een variabele van het type **`int`** is, dan kan de delay ten hoogste ongeveer 32 seconden zijn. Als de delay langer moet worden moet je x als een **`long`** of een **`unsigned long`** definiëren.

2.10 Knipperlicht 1

De volgende sketch laat een LED knipperen op digitale pin 13 van de Arduino. Op het Arduino UNO board is al een LED aanwezig, aangesloten op pin 13 en gemerkt met een L. Je hoeft zelf dus geen extra LED aan te sluiten.

Opdracht 9: Knipperlicht 1

Sluit de Arduino UNO met een USB-kabel aan op de computer. Start het Arduino IDE programma en typ het volgende programma in de editor:

```
//////////  KNIPPERLICHT 1  ////////////////////////////////////////////////

int LED = 13;                  // LED komt op pin 13.

void setup(){                  // De setup wordt maar één keer doorlopen.
  pinMode(LED, OUTPUT);        // Maak van pin LED (pin 13 dus) een uitgang.
}                              // Setup wordt afgesloten emt een }.

void loop(){                   // De loop blijft herhalen.
  digitalWrite(LED, HIGH);     // Zet de LED aan.
  delay(500);                  // Wacht een halve seconde.
  digitalWrite(LED, LOW);      // Zet de LED uit.
  delay(500);                  // Wacht een halve seconde.
}                              // Loop wordt afgesloten met een }.
```

Staan alle haakjes, accoladen en puntkomma's op de goede plaats?
Upload het programma naar het Arduino board met de upload-knop. Werkt het?
Kan je de LED twee keer zo snel laten knipperen?
Onderzoek hoeveel milliseconden je de LED minimaal moet laten flitsen om het te kunnen zien.

Let op: de definitie van de variabele `int LED = 13;` moet vóór de setup staan. Dan geldt het voor het hele programma. Als je deze definitie in de setup zet geldt hij alléén voor de setup en niet meer voor de loop. Het gevolg is dat de loop de variabele LED niet herkent en dus werkt het programma niet!
Je kunt de definitie ook in de loop zetten, maar dan kan je LED alléén in de loop gebruiken en dus werkt de pinMode (LED, ...) opdracht in de setup niet.

2.11 Knipperlicht 2

Deze sketch is een variatie op knipperlicht 1. Hierbij wordt een variabele in de loop veranderd door een berekening.

Opdracht 10: Knipperlicht 2

Bestudeer de volgende sketch en voorspel wat de microcontroller zal gaan doen.

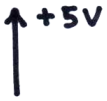
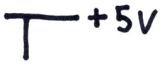
```
//////////  KNIPPERLICHT 2  ////////////////////////////////////////////////

int LED = 13;                 // LED komt op pin 13.
int k = 0;                    // Geheel getal k = 0.

void setup(){                 // De setup wordt maar één keer doorlopen.
  pinMode(LED, OUTPUT);       // Maak van pin LED (pin 13 dus) een uitgang.
}                             // Setup wordt afgesloten emt een }.

void loop(){                  // De loop blijft herhalen.
  digitalWrite(LED, HIGH);    // Zet de LED aan.
  delay(k);                   // Wacht k milliseconden.
  digitalWrite(LED, LOW);     // Zet de LED uit.
  delay(k);                   // Wacht k milliseconden.
  k = k + 5;                  // De waarde van k wordt met 5 verhoogd.
}                             // Loop wordt afgesloten met een }.
```

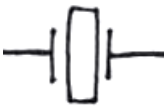
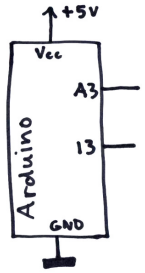
Upload de sketch en controleer of jouw voorspelling juist was.
Kan je de LED ook steeds sneller laten knipperen?

3 Elektronica

3.1 Elektronische schakelingen tekenen

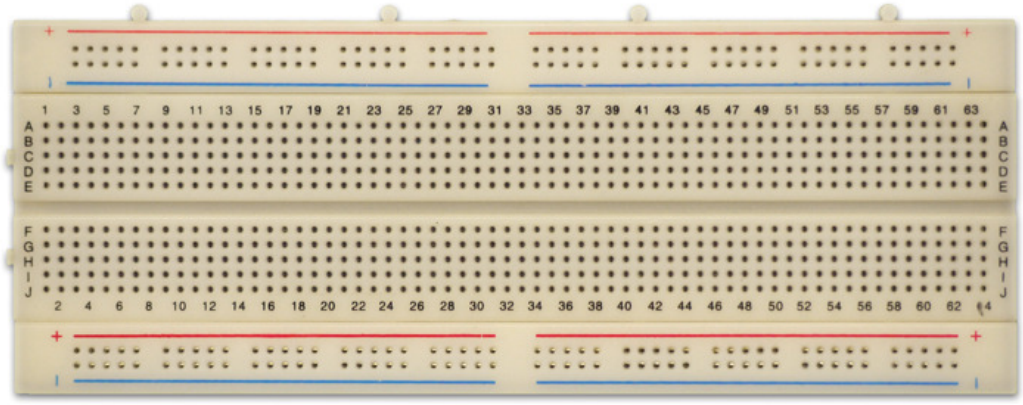
Tot nu toe heb je nog geen schakelingen met de Arduino hoeven bouwen. Als je de ingangen en uitgangen van de Arduino gaat verbinden met sensoren, LED's en andere onderdelen, dan doe je dat volgens een schema. In het schema zijn alle onderdelen met speciale symbolen overzichtelijk getekend. De volgende onderdelen en symbolen heb je regelmatig nodig en moet je kunnen tekenen:

		Verbindingsdraad
		Kruisende verbindingsdraden maken geen contact met elkaar!
		Deze draden maken wel contact met elkaar.
		Zo teken je meerdere draden die contact maken. Dus geen kruisingen, alleen driesprongen.
		Ground of GND Deze draad is verbonden met Ground. Ground betekent 0 volt. Lage spanningen teken je liefst onderaan in het schema.
		5 V Een verbinding naar de voedingsspanning kan zo worden aangegeven. Hoge spanningen teken je liefst bovenaan in het schama.
		Dit is een andere manier om de voedingsspanning aan te geven.
		Schakelaar Met een schakelaar kun je een stroomkring onderbreken of sluiten.
		Drukschakelaar (of pulsschakelaar) Een drukschakelaar maakt alleen contact als hij ingedrukt is.
		Reed relais Een reed relais is een schakelaartje wat contact maakt als je een magneet in de buurt houdt.
		Weerstand De waarde kun je er bij schrijven, bijv: 220Ω of 10k. Je kunt ze ook nummeren: R1, R2, R3 enz. en in een tabel de waarden schrijven.

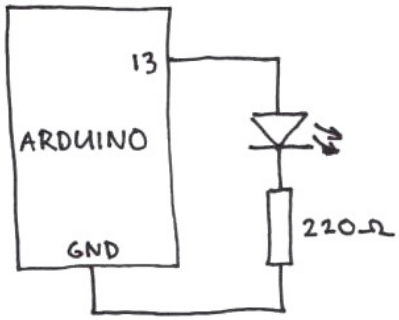
		In Engeland en de USA gebruikt men dit symbool vaak voor een weerstand. Wij gebruiken dit symbool niet, maar als je een engels schema moet lezen is het handig om te weten.
		Potentiometer (of potmeter) Dit is een weerstand met een aftakking. Je kunt er aan draaien en zo de plaats van de aftakking instellen.
		LDR (Light Dependent Resistor) Deze weerstand reageert op licht. Hoe meer licht er op schijnt, des te lager de weerstand.
		Variabele weerstand Deze weerstand kan afhankelijk zijn van temperatuur, druk, vochtigheid, enz. Zet er bij waarvan hij afhankelijk is, bijv: °C, druk, enz
		Diode Een diode laat de stroom maar in één richting door, de richting van de pijl. Met een diode kan je van wisselstroom gelijkstroom maken.
		LED Een LED (Light Emitting Diode) is een diode die licht uitzendt als er stroom door gaat. Omdat een LED een diode is werkt hij niet als je hem verkeerdom aansluit.
		Kristal De microcontroller heeft een kristal van 16 Mhz nodig om zijn klok goed te laten lopen.
		Een ingewikkelder onderdeel met meerdere aansluitingen kun je tekenen als een rechthoek, met daarin de naam en de aansluitingen vermeld.

3.2 Schakelingen bouwen op een breadboard

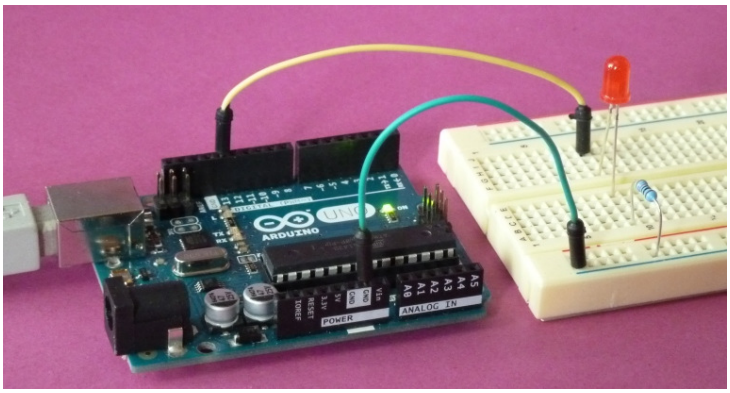
Een breadboard is een kunststof plaatje met rijen gaatjes. Onder de gaatjes zitten geleidende metalen strips in de vorm van veerklemmen, die contact maken met de pootjes van de elektronische componenten of snoertjes die je er in prikt. Met een breadboard kun je prototypes van elektronische schakelingen bouwen zonder te solderen.
Bij de kolommen 1 t/m 64 zijn de gaten ABCDE en de gaten FGHIJ via geleidende klemmen met elkaar verbonden. De gaatjes van de rode (+) en de blauwe (-) rijen zijn over de gehele breedte met elkaar verbonden.



Als je een tekening kan lezen en maken, dan kan je de schakeling maken op het breadboard.



Pin 13 van de arduino is via een LED en een weerstand verbonden met GND.



Zo kan het er in het echt uitzien als je het hebt gebouwd.

Een LED heeft een plus-kant (ook wel anode genoemd) en een min-kant (ook wel kathode). Als je een LED verkeerdom aansluit werkt hij niet. De lange poot is de plus, de korte poot is de min.
We willen een rode LED aansluiten op pin 13 van de Arduino.
De lange poot van de LED wordt aangesloten op digitale uitgang 13 van de Arduino.
De korte poot wordt aangesloten op een poot van een weerstand van 220Ω.
De andere poot van de weerstand wordt aangesloten op GND. GND betekent Ground en is 0 Volt.
Het maakt niet uit of de LED vóór of na de weerstand zit. De weerstand is nodig om ervoor te zorgen dat er niet te veel stroom door de LED gaat. Als er teveel stroom door de LED gaat kan hij doorbranden of (erger nog) de microcontroller gaat kapot!

Opdracht 11: Knipperlicht 3

Je gaat een rode LED aansluiten op pin 11 van de Arduino, en een groene LED op pin 9.
Teken eerst het schema.
Sluit de lange blauwe (–) strips van het breadboard aan op een GND pen van de Arduino.
Sluit de LED's en de weerstanden aan volgens jouw schema.
Pas het programma 'knipperlicht 2' aan zodat beide LED's knipperen. Als de rode LED aan is moet de groene LED uit zijn en andersom.

3.3 Verkeerslichten 1

De gemeente wil een veilige oversteekplaats voor voetgangers laten maken. Een brug of een tunnel is te duur, dus hebben burgemeester en wethouders besloten dat er een verkeerslicht moet komen. De verkeersdeskundige van de gemeente heeft laten meten hoeveel auto's er dagelijks voorbij komen en hoeveel voetgangers dagelijks moeten oversteken. Op grond van de metingen heeft hij het volgende berekend:
Het licht voor de auto's moet een halve minuut aaneengesloten groen zijn.
Het licht voor de voetgangers moet 8 seconden groen zijn, daarna 8 seconden groen knipperen voordat het weer rood wordt.

Als het licht voor de auto's rood is geworden mag het licht voor de voetgangers niet direct op groen springen. Dit mag pas nadat het autolicht 5 seconden rood is.
Voordat het autolicht rood wordt, moet het eerst 5 seconden op oranje staan.
Voordat het autolicht op groen springt moet het voetgangerslicht 5 seconden op rood staan.
Als het voetgangerslicht groen knippert moet het groene licht afwisselend 0,5 seconden uit en 0,5 seconden aan zijn.

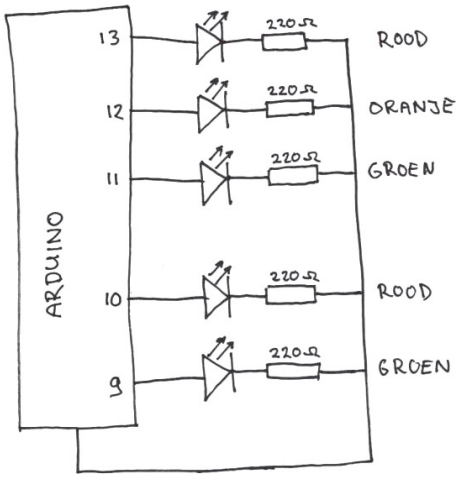
Bovenstaande ontwerpeisen zijn geschreven in 'mensentaal'. Het is handig om hiervan een waarheidstabel te maken voordat je gaat programmeren. In de waarheidstabel staat in de goede volgorde van alle lichten aangegeven of ze aan (1) of uit (0) staan, en hoe lang.
Als je begint met het moment waarop het licht voor de auto's groen wordt, dan kan het begin van de waarheidstabel er zo uit zien:

waarheidstabel verkeerslichten

autolichten			voetgangerslichten		tijdsduur
rood	oranje	groen	rood	groen	
0	0	1	1	0	30 s
0	1	0	1	0	5 s

Opdracht 12: Verkeerslichten 1

Maak een waarheidstabel voor de verkeerslichten van één hele cyclus.
Programmeer de verkeerslichten op de pinnen van de Arduino volgens het schema hiernaast.
Maak een prototype met een breadboard.
Upload de sketch.
Werkt het? Sla je sketch op, je hebt hem later misschien weer nodig!



3.4 Spanning stroom en weerstand

I Een elektrische stroom bestaat uit elektronen die zich verplaatsen. Hoe meer elektronen zich verplaatsen, des te groter de stroom. Stroom wordt aangegeven met de letter I. De stroomsterkte wordt uitgedrukt in ampère (A). Als er een stroom van 1 A door een draad loopt, dan stromen er per seconde 6.241.510.000.000.000.000 elektronen door die draad. Je kunt dan zeggen I = 1A.

V Er gaat pas een stroom lopen als de elektronen door de draad geduwd worden. De elektrische spanning duwt de elektronen door de draad. Spanning wordt aangegeven met de letter V en uitgedrukt in volt (V). Een AA batterij heeft een spanning van ongeveer 1,5 V. De spanning van het stopcontact is 230 V. Hoe hoger de spanning, des te groter de stroom I. Bij een spanning van 0 volt worden er geen elektronen geduwd en loopt er geen stroom.

R Elektronen stromen niet overal even gemakkelijk doorheen. Koper is een geleider, hier gaan de elektronen gemakkelijk doorheen maar plastic is een isolator. Hier krijg je de elektronen niet doorheen geduwd. Hoe moeilijk de elektronen ergens doorheen kunnen stromen noemen we de weerstand en geven we aan met de letter R. De eenheid van weerstand is de ohm (Ω). Hoe groter de weerstand, hoe moeilijker de elektronen er doorheen gaan. Een koperdraad heeft geen weerstand oftewel: R = 0Ω. De weerstand van een isolator zoals lucht, glas of plastic is oneindig. In de elektronica worden weerstanden gebruikt om verschillende stromen en spanningen te maken.

3.5 De wet van Ohm: V = I x R

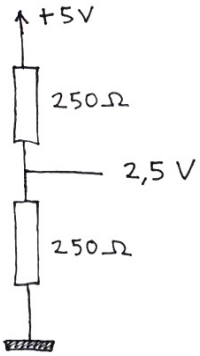
Stroom, spanning en weerstand hebben met elkaar te maken. Als je een weerstand aansluit op een spanning, dan gaat er een stroom lopen. Je kunt uitreken hoeveel stroom er gaat lopen met een formule. Deze formule heet de wet van Ohm:

V = I x R (spanning in Volt = stroom in Ampère x weerstand in Ohm)

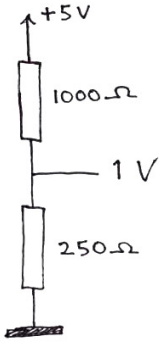
Als **V = I x R** dan geldt ook: **I = V / R** en **R = V / I**

Als je een weerstand van 100Ω aansluit op een spanning van 5 volt, dan gaat er een stroom lopen van: I = V / R = 5 / 100 = 0,05 A .
0,05 A = 50 mA (milli-ampère).
Een elektrotechnisch ingenieur gebruikt bijna dagelijks de wet van Ohm.

3.6 De spanningsdeler



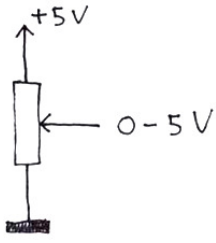
Hoe kun je een elektrische spanning maken? Met een elektrische stroom door weerstanden!
Stel, je sluit twee weerstanden van 250 Ω in serie aan op een spanning van 5 V. De totale weerstand is dan 500 Ω. Er gaat dan een stroom lopen van I = V / R dus 5V / 500 Ω = 0,01 A.
Door beide weerstanden loopt dezelfde stroom: 0,01 A.
Door weerstand 1 loopt een stroom van 0,01 A. De weerstand is 250 Ω. De spanning over die weerstand is dan: V = I x R dus 0,01 x 250 = 2,5 V.
De spanning over weerstand 1 = 2,5 V. De spanning over weerstand 2 is ook 2,5 V. De spanning over twee gelijke weerstanden wordt in twee gelijke stukken verdeeld. Twee weerstanden in serie noemen we een spanningsdeler.
Als je twee verschillende weerstanden neemt, dan verhouden de spanningen zich als de weerstanden.



Stel je wilt uitgaande van een spanning van 5 V een spanning van 1 V maken met een spanningsdeler. De spanningen over de weerstanden verhouden zich dan als 1:4. Dan moeten de weerstanden zich ook verhouden als 1:4.
Je zou dus weerstanden kunnen nemen van 250 Ω en 1000 Ω. Of weerstanden van 470 Ω en 1880 Ω.
Je kunt met de wet van Ohm uitrekenen dat het klopt!

3.7 De potmeter

Een potentiometer (of potmeter) is een weerstand met een aftakking. Door aan de as te draaien kun je de aftakking verplaatsen. De potmeter is dus een spanningsdeler waarvan je de weerstandverhouding kunt instellen.
Als je een potmeter met de buitenste twee poten op 0 V en 5 V aansluit, dan kun je op de aftakking elke gewenste spanning tussen 0 V en 5 V krijgen door aan de as te draaien. De aftakking vindt je op de middelste poot. Deze poot noem je de loper.



4 Sensoren

4.1 Analoge sensoren

Analoge sensoren zijn vaak variabele weerstanden. De weerstand verandert door een natuurkundige grootheid zoals de temperatuur, druk of de hoeveelheid licht. Als je met zo'n sensor en een gewone weerstand een spanningsdeler maakt kun je de spanning laten veranderen door een veranderende grootheid.

Een LDR is een voorbeeld van zo'n sensor. LDR betekent Light Dependent Resistor (licht afhankelijke weerstand). Als je een LDR in het licht houdt is de weerstand laag. Hoe feller het licht, des te lager de weerstand. Als je de LDR in het donker houdt is de weerstand juist heel hoog. Dus als je een spanningsdeler maakt met een LDR en een 'vaste' weerstand, dan is de spanning tussen beiden afhankelijk van de lichtintensiteit.
In de schakeling hiernaast geldt: Hoe meer licht, des te hoger spanning U.

4.2 De ingangen van de microcontroller

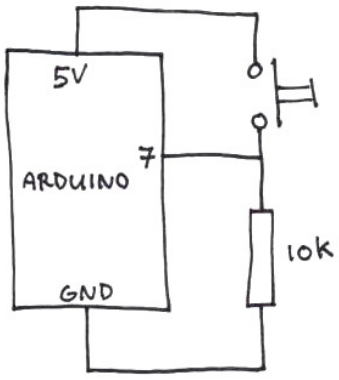
De ATmega328 microcontroller heeft 6 analoge ingangen en maximaal 14 digitale ingangen. Via deze ingangen kan de chip de buitenwereld waarnemen. Dit doet hij door middel van elektrische spanningen. Sensoren en schakelaars geven informatie aan de microcontroller in de vorm van een elektrische spanning.

Een digitale ingang kan een spanning van 5 volt of 0 volt onderscheiden. 0 volt op een digitale ingang noemen we LOW, 5 volt noemen we HIGH. Op een digitale ingang kun je een digitale sensor aansluiten of een schakelaar die de ingang op de juiste manier met 0 volt of 5 volt verbindt. Een digitale waarde op een ingang kun je uitlezen met de opdracht `digitalRead`.

De analoge ingangen van de microcontroller kunnen ook alle tussenliggende spanningen tussen 0 en 5 volt 'meten'. Het bereik van 0 - 5 volt wordt door de A/D converter (Analoog naar digitaal omzetter) van de ATmega328 chip verdeeld in 1023 gelijke stapjes, van 0 t/m 1023. Zet je een spanning van 5 volt op de analoge ingang, dan is de analoge waarde 1023. Bij 0 volt is de analoge waarde 0. Bij 1 volt is de waarde dus 1/5 x 1023 = 205. De analoge waarde kun je uitlezen met de opdracht `analogRead`.

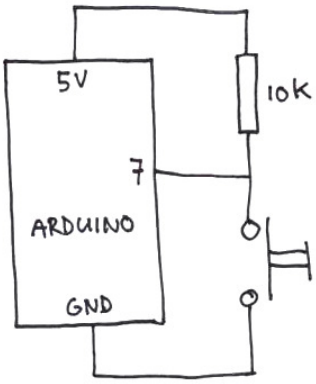
4.3 Een schakelaar als input

Een schakelaar kun je beschouwen als een digitale sensor. Hij heeft twee waarden: gesloten of open (ingedrukt of niet ingedrukt). Als je de schakelaar indrukt is de weerstand 0Ω, als je hem niet indrukt is de weerstand oneindig. Verbindt je een digitale ingang via een schakelaar met 5V, dan kun je een digitaal 'HIGH' signaal krijgen door de schakelaar in te drukken. Maar wat nu als je de schakelaar niet indrukt? Om er voor te zorgen dat de spanning dan 0V (LOW) wordt moet je de digitale ingang ook nog via een weerstand verbinden met de 0V (GND). Hiervoor kun je een weerstand van 10.000 Ω nemen. 10.000 Ω = 10 kΩ (kilo-ohm) of kortweg 10k. Bestudeer de schakeling. Eigenlijk is dit ook een spanningsdeler. De schakelaar in de spanningsdeler heeft een weerstand van 0 Ω of oneindig Ω. Dan is de spanning op digitale ingang 7 dus 0 V (niet ingedrukt) of 5 V (wel ingedrukt).

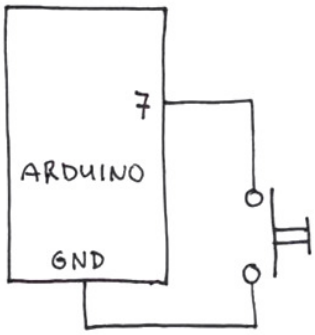


4.4 Pullup en pulldown

De weerstand van 10k in het vorige voorbeeld wordt ook wel een pull-down weerstand genoemd. Hij trekt de spanning naar de GND als de schakelaar geopend is. Je kunt een schakelaar ook zo aansluiten dat de ingang van de Arduino juist laag wordt als je de schakelaar indrukt. De weerstand en de schakelaar moeten dan worden verwisseld. In deze situatie wordt de weerstand ook wel pull-up genoemd.



Schakelaar met pullup weerstand



Met ingebouwde pullup

De ATmega328 heeft zelfs pullup weerstanden ingebouwd! Met de volgende opdracht kun je een digitale ingang via de ingebouwde pullup weerstand met de +5V verbinden:

```
pinMode (pin, INPUT_PULLUP); // Ingang pin krijgt zo een interne pullup.
```

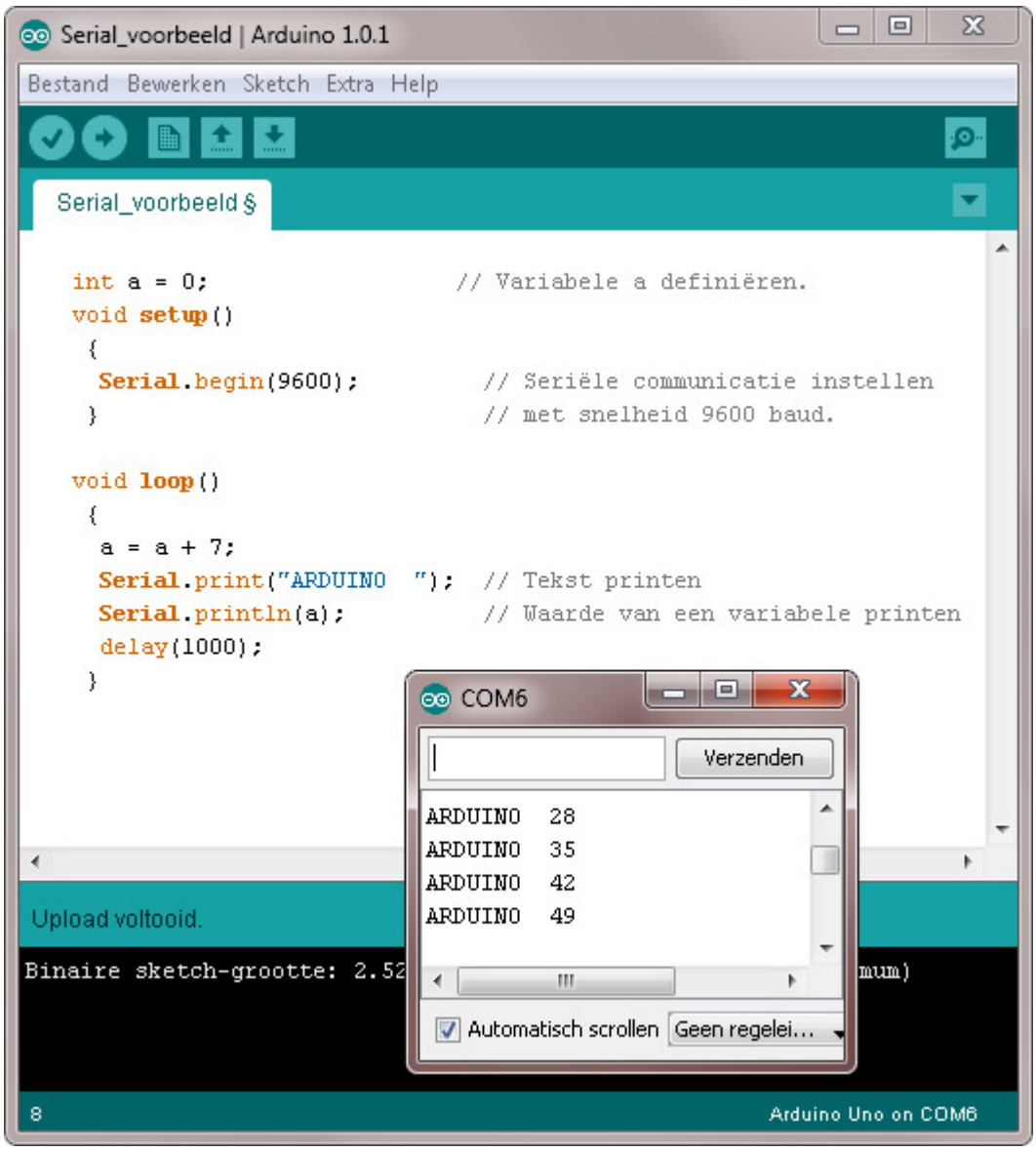
Je hebt dan geen extra weerstand meer nodig. Je hoeft de interne weerstand in de microcontroller niet te tekenen. Je kunt de interne pullup niet bij een uitgang gebruiken.

4.5 Seriële communicatie

Als je met sensoren werkt is het handig als je tijdens het programmeren van je prototype kunt zien welke waarden de microcontroller aan de analoge ingangen 'ziet'.
Zolang de ATmega328 chip op het Arduino-board zit kun je via het programma ook gegevens naar de computer terug sturen. De belangrijkste opdrachten hiervoor zijn:

```
Serial.begin(9600);       // Seriële communicatie instellen.
                          // Dit zet je in de setup.
                          // Tussen haakjes staat de baudrate (snelheid).
Serial.print("ARDUINO"); // Print de tekst ARDUINO naar de computer.
Serial.print(a);          // Print de waarde van variabele a.
Serial.println(a);        // Print de waarde van variabele a en
                          // ga naar de volgende lijn.
```

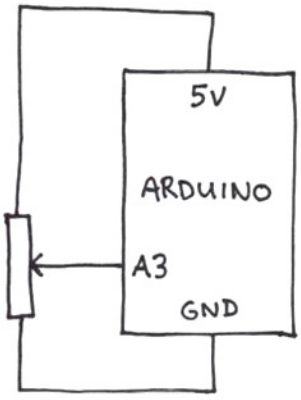
Je kunt de seriële communicatie op de computer zien via het menu Extra/Seriële monitor. Je opent dan een nieuw venster waarin de verstuurde gegevens worden 'geprint'.
De Seriële data worden ontvangen en verzonden via digitale pinnen 0 en 1. Deze pinnen kun je dus niet voor andere dingen gebruiken tijdens de seriële communicatie!



Seriële communicatie en het monitor-venster.

Opdracht 13: De potmeter

Sluit een potmeter aan volgens onderstaand schema en typ de sketch in de IDE:




```
int x = 0;

void setup(){
  Serial.begin(9600);
}

void loop(){
  x = analogRead(3);
  Serial.println(x);
  delay(100);
}
```

Upload de sketch en bekijk via de seriële monitor hoe je de waarde van de analoge ingang kunt veranderen met de potmeter.

Opdracht 14: Licht meten

Vervang de potmeter in opdracht 13 door een LDR en een weerstand van 10 kΩ (zie hoofdstuk 4.1). Onderzoek met de seriële monitor hoe je de analoge ingang kunt veranderen met licht.

Opdracht 15: Potmeter en LED

Breid de schakeling van opdracht 13 uit met een LED en een weerstand van 220Ω op één van de digitale uitgangen.
Teken het schema.
Schrijf een sketch waarmee je de knipperfrequentie van de LED kan regelen met de potmeter.

4.6 Sensoren ijken

Om een schemerschakelaar te kunnen maken hoef je alleen maar te weten wat de waarde van de analoge ingang is bij één bepaalde lichtintensiteit. De schemerschakelaar hoeft alleen maar te reageren als die ene waarde wordt bereikt. Deze waarde kun je meten met een proefopstelling zoals in opdracht 14.
Maar wat moet je doen als je meerdere waarden wilt gebruiken?

De spanning van 0 tot 5 volt op een analoge ingang van de microcontroller wordt door de A/D convertor in 1023 stapjes verdeeld, van 0 tot 1023. Elk stapje is dus 5 / 1023 = 0,00488 volt. Met een eenvoudige berekening kun je uit de analoge waarde berekenen hoeveel volt er op de ingang staat. Zo kun je van de Arduino een voltmeter maken die tot maximaal 5 volt kan meten:

```
V = (analogRead(3)/1023)*5;  // V = spanning in volt op pin A3
```

Stel je hebt een lineaire temperatuursensor die je wilt ijken in ºC. Omdat het een lineaire sensor is heb je maar twee waarden nodig om te ijken. Je moet dus eerst de analoge waarden bij twee bekende temperaturen meten, bijvoorbeeld ijswater en kokend water.
Stel je meet een analoge waarde van 78 in ijswater en 844 in kokend water.
Om de analoge waarde om te rekenen naar ºC moet je er eerst 78 van aftrekken.
Dan moet je het bereik van 100º verdelen in 844-78=766 stapjes, dus vermenigvuldigen met 100/766. De totale berekening wordt dan:

```
grC = (analogRead(3)-78)*(100/766);  // Sensor op pin A3 ijken in grC
```

4.7 map

Om het ijken van lineaire sensoren gemakkelijk te maken is de opdracht map uitgevonden. In de map opdracht zet je tussen haakjes de waarde die je wilt mappen, het bereik er van in een lage en een hoge waarde, en naar welk bereik het moet worden omgerekend. Dus:

y = map(x, van lage waarde, van hoge waarde, naar lage waarde, naar hoge waarde);.

De temperatuursensor waarvoor we een formule hebben gemaakt zou je dus ook zo kunnen ijken:

```
grC = map(analogRead(3), 78, 844, 0, 100);  // Sensor ijken in grC
```

Met de map opdracht kunnen alleen lineaire sensoren worden geijkt. Een niet-lineaire sensor ijken kan wel met een formule, maar dat is een stuk lastiger. We gaan er hier niet verder op in.

Vragen en opdrachten

16. Je kunt van de Arduino een millivoltmeter maken met de map opdracht. Hoe ziet die map opdracht er precies uit?
17. Je wilt van een (lineaire) potmeter een hoekmeter maken waarmee je een hoek in graden kunt meten. De potmeter heeft van begin tot eind een draaihoek van 280 graden. Hoe ziet de map opdracht er uit?

5 Vergelijken

5.1 if

Met de if opdracht kan je het programma beslissingen laten nemen. If betekent als.
Bijvoorbeeld: Als de schakelaar ingedrukt wordt, doe dan dit. Of: Als de ingestelde tijd nog niet bereikt is doe dan dat. Of: Als de temperatuur te laag is, doe dan de verwarming aan. Enzovoort.
De voorwaarde staat tussen haakjes achter if. Daarna staat tussen accolades wat moet worden gedaan als aan de voorwaarde wordt voldaan. Zo kan het er in de sketch uit zien:

```
if (x < y){                  // Als x kleiner is dan y.
  digitalWrite(7,HIGH);      // Doe dan wat tussen de accoladen staat.
}                            // In dit voorbeeld uitgang 7 hoog maken.
```

5.2 Vergelijkingen

De if-opdracht wordt gebruikt in combinatie met een vergelijking. Bijvoorbeeld: als x gelijk is aan y of: als x groter is dan y enzovoort. Hier zie je de mogelijke vergelijkingen:

```
x == y    // x is gelijk aan y.
x != y    // x is ongelijk aan y.
x < Y     // x is kleiner dan y.
x > y     // x is groter dan y.
x <= y    // x is kleirer of gelijk aan y.
x >= y    // x is groter of gelijk aan y.
```

5.3 else

Als niet aan de voorwaarde in de if-vergelijking wordt voldaan kun je eventueel met de opdracht else iets anders laten doen.

```
if (temp >= 30){             // Als temp is groter of gelijk aan 30.
  digitalWrite(7,LOW);       // Maak pin 7 laag.
  delay(1000);               // En wacht een seconde.
}
else{                        // Anders (dus als temp < 30).
  digitalWrite(7,HIGH);      // Doe wat tussen de { en } staat.
  delay(1000);
}
```

5.4 for

Met de for opdracht kan een deel van het programma een bepaald aantal malen worden herhaald. Een deel dat herhaald wordt noemen we een loop. Met een variabele wordt bijgehouden hoe vaak er wordt herhaald.
Achter for staan tussen haakjes achtereenvolgens de beginwaarde van de variabele die de teller bijhoudt, een vergelijking die bijhoudt of de loop opnieuw moet worden doorlopen en een berekening met de teller die bepaalt hoe er moet worden geteld.

```
for (int x=0; x<100; x=x+1){  // Begin bij  x = 0.
                              // Ga door zolang x < 100.
                              // Maak x na elke doorloop 1 hoger.
}                             // Doe wat tussen { en } staat.
```

In dit voorbeeld worden de opdrachten tussen { en } 100 keer uitgevoerd.
Opdracht 18: Bestudeer het programma op bladzijde 9.
Begrijp je nu hoe het werkt?

5.5 while

De while opdracht kijkt net als de if opdracht of er wordt voldaan aan een vergelijking. De opdrachten tussen de accoladen worden herhaald zolang er aan de voorwaarde wordt voldaan.

```
while (x < 100){   // Zolang variabele x < 100.
}                  // Herhalen wat tussen { en } staat.
```

Met de while opdracht kun je het programma laten wachten tot een knop wordt ingedrukt. De knop is aangesloten met een interne pullup. Tussen de accoladen staat niets, dus zolang de drukknop niet is ingedrukt (HIGH) gebeurt er niets. Zodra de knop wordt ingedrukt is de ingang LOW en wordt niet meer voldaan aan de vergelijking dus gaat het programma verder:

```
while (digitalRead(drukknop) == HIGH){}  // Wacht tot drukknop LOW wordt.
```

5.6 millis()

Als je de ATmega328 aan zet begint er in de chip een teller te lopen. Deze teller wordt elke milliseconde met één verhoogd. Je kunt de tellerstand opvragen met de opdracht **`millis`()**. Met deze opdracht is het mogelijk om tijden te programmeren, afhankelijk van gebeurtenissen, zoals een kookwekker, een vertragingsschakelaar, een stopwatch of een reactietijd-tester.

```
int Tnu = millis();           // Maak Tnu gelijk aan millis().
int Teind = Tnu + 60000;      // Maak Teind 1 minuut na Tnu.
while (Tnu < Teind){          // Zolang Teind nog niet bereikt is.
  digitalWrite (LED, HIGH);   // LED aanzetten.
  Tnu = millis();             // Tijd bijhouden.
}
digitalWrite (LED, LOW);
```

In dit voorbeeld wordt een LED een minuut lang aangezet. Het is wel ingewikkelder dan delay, maar het voordeel is dat je tijdens de loop ook andere dingen kunt doen, zoals sensoren (bijvoorbeeld drukknoppen) uitlezen.

Vragen en opdrachten

19. millis() is een variabele van het type **`unsigned long`**. Hoeveel dagen moet de Arduino aan staan voordat de geheugenplaats van millis() vol is?

5.7 Om te proberen: Een reactietijdmeter

```
/* ///////////  Reactietest  ////////////////////////////////////////////
Open de seriële monitor in het menu extra.
Druk op de knop om te starten.
Wacht tot de LED aan gaat.
Druk dan zo snel mogelijk weer op de knop.
*/

int LED = 7;           // Sluit een LED aan op pin 7.
int drukknop = 8;      // Sluit een drukknop aan op pin 8.
long Tbegin = 0;       // Definiëer variabele Tbegin.
long Teind = 0;        // Definiëer variabele Teind.
int Treactie = 0;      // Definiëer variabele Treactie.

void setup(){
  pinMode (LED, OUTPUT);
  pinMode (drukknop, INPUT);
  Serial.begin(9600);        // Seriële communicatie aan.
}

void loop(){
  digitalWrite(LED, HIGH);
  while (digitalRead(drukknop) == LOW){}  // Wacht op drukknop.
  digitalWrite (LED, LOW);                // Starten: LED uit.
  delay (random(4000, 7000));             // Wacht tussen 4 en 7 s.
  digitalWrite(LED, HIGH);                // LED aan.
  Tbegin = millis();                      // Begintijd.
  while (digitalRead(drukknop) == LOW){}  // Wacht op drukknop.
  Teind = millis();                       // Eindtijd.
  digitalWrite(LED, LOW);                 // LED weer uit.
  delay (1000);
  Treactie = Teind - Tbegin;              // Reactietijd berekenen.
  Serial.print("reactietijd: ");
  if(Treactie < 100){                     // Als je te vroeg drukt.
    Serial.println("VALS !!!");           // Tekst printen
  }
  else{                                   // Goed gedrukt.
    Serial.print(Treactie);               // Variabele printen.
    Serial.println(" ms");                // Tekst printen
  }                                       // Daarna naar volgende regel.
}
```

5.8 Verkeerslichten met aanvraag

Het verkeerslicht uit opdracht 12 is dom. Het houdt geen rekening met het verkeersaanbod. Het voetgangerslicht springt steeds weer op groen, ook als er geen voetgangers zijn. Dit geeft onnodige vertraging voor de auto's. Bovendien zorgt het afremmen en optrekken van de auto's voor meer lawaai en vervuiling.
Je gaat het verkeerslicht uitbreiden met een drukknop voor de voetgangers.
De eisen zijn grotendeels dezelfde als bij opdracht 12, dus misschien is het handig om je sketch van opdracht 12 te gebruiken als uitgangspunt.
Er komt één nieuwe eis bij: Het voetgangerslicht wordt alleen groen na een aanvraag met een drukknop.

Opdracht 20: Verkeerslichten met aanvraag

Programmeer verkeerslichten als bij opdracht 12, maar nu met een drukknop voor de voetgangers. Maak een prototype met 6 LED's (rood, oranje en groen voor de auto's, rood en groen voor de voetgangers en een witte of blauwe LED die aangeeft dat een aanvraag is gedaan) en een drukschakelaar.
Teken eerst het schema voordat je gaat bouwen en programmeren.

5.9 random

De reactietijdmeter uit hoofdstuk 5.7 wacht een willekeurige tijd tussen de 4 en 7 seconden voordat de LED aan gaat. Om een willekeurig getal te maken heb je de random opdracht nodig:

```
random(b);                  // Willekeurig getal tussen 0 en b.
random(a, b);               // Willekeurig getal tussen a en b.
D = random(1, 7);           // Maak D willekeurig getal 1, 2, 3, 4, 5 of 6.
delay(random(4000, 7000)); // Wacht een willekeurige tijd tussen 4 en 7 seconden.
```

Elke keer als je de Arduino opstart wordt dezelfde reeks random getallen gegenereerd. Als je wilt dat de getallenreeks elke keer als je de Arduino aan zet anders is moet je met een willekeurig getal beginnen. Dit is de randomseed. Een veelgebruikte methode om deze randomseed te maken is door een niet gebruikte analoge ingang te lezen. Als een pin van een analoge ingang niet gebruikt wordt zit er altijd een beetje ruis op. Dat betekent dat de analoge waarde op elk moment willekeurig anders is. Je hoeft maar één keer een randomseed aan te maken, bijvoorbeeld in de setup:

```
randomSeed(analogRead(0)); // Een willekeurige beginwaarde voor de randomgenerator.
```

5.10 switchcase

De if-opdracht uit hoofdstuk 5.1 kun je zo vaak herhalen als je wilt maar soms is de switchcase opdracht handiger om te gebruiken.

Vragen en opdrachten

21. Zoek uit hoe de switchcase-opdracht werkt.

Opdracht 22: Dobbelsteen

Ontwerp, programmeer en bouw een prototype van een originele elektronische dobbelsteen.

6 Actuatoren

6.1 Geluid

De ATmega328 chip kan ook geluiden maken. Dit doe je met de opdracht tone.
Om het signaal hoorbaar te maken moet je een luidsprekertje aansluiten tussen de digitale uitgangspin en GND.

```
tone(9, 880);          // Maak op pin 9 een toon van 880 Hz
tone(9, freq, 1000);   // Maak op pin 9 een toon van freq Hz gedurende 1s.
tone(L, freq, T);      // Maak op pin L een toon van freq Hz gedurende T ms.
noTone(speaker);       // Zet de toon op pin speaker uit.
```

Met deze opdracht kun je dus allerlei geluiden maken. Een simpel piepje, een sirene of een complete melodie.

```
////  Sirene  ////////////////////////////////////////////

int speaker = 9;                  // Luidspreker op pin 9.

void setup(){
  pinMode(speaker, OUTPUT);
}

void loop(){
  for (int i=200; i<2000; i++){  // Van 200Hz naar 2000Hz.
    tone(speaker, i);
  }
  for (int i=2000; i>200; i--){  // Van 2000Hz naar 200Hz.
    tone(speaker, i);
    delay(1);                    // Langzaam naar beneden.
  }
}
```

Vragen en opdrachten

23. In de sirene sketch hierboven staat in de for-loop `++`. Dit is een zogenaamde compound operator. `i++` betekent hetzelfde als `i=i+1`. Er bestaan nog meer van dat soort compound operators. Zoek ze op.

6.2 Array

Een array is een geïndexeerde lijst van meerdere variabelen. Dat betekent dat elke variabele in de lijst ook een nummer heeft. Je kunt een array een naam geven, net als één enkele variabele:

```
int mijnlijst[] = {2004, 112, 3, 62, 577, 2015}; // Deze array bevat 6 variabelen.
```

De indexnummering begint bij 0. Variabele 5 in de array mijnlijst is dus 2015.
Je kunt variabelen uit de lijst halen met het indexnummer tussen de [ ].

```
a = mijnlijst[0]; // a krijgt de waarde 2004.
x = mijnlijst[1]; // x krijgt de waarde 112.
y = mijnlijst[5]; // y krijgt de waarde 2015.
```

Je kunt variabelen in een array ook veranderen (anders zouden het ook geen variabelen zijn!).

```
mijnlijst[3] = 66 // Zet het getal 66 op indexplaats 3 van mijnlijst.
mijnlijst[c] = 88 // Zet het getal 88 op indexplaats c van mijnlijst.
mijnlijst[8] = 88 // Werk nooit buiten het bereik van de array!!!
```

Pas op met indices die buiten de lijst vallen. Je schrijft dan een getal op een geheugenplaats buiten de lijst. Deze geheugenplaats wordt door de microcontroller misschien ergens anders voor gebruikt. Er kunnen dan allerlei onverwachte dingen gebeuren dus pas op!
Arrays zijn handig om een melodie te programmeren. Probeer maar:

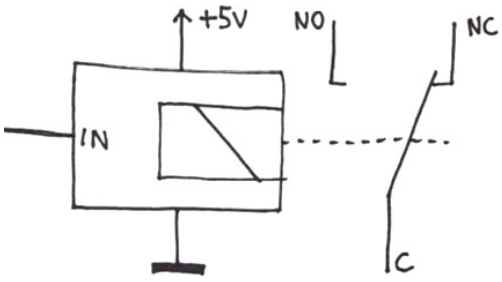
```
///// Melodietje ////////////////////////////////////////

int speaker = 9;  // Luidsprekertje op pin 9
int frequentie[] = {524, 392, 392, 440, 392, 494, 524};
int duur[] = {400,200,200,400,800,400,400};

void setup(){
  pinMode (speaker, OUTPUT);
  for (int i = 0; i<7; i++){
    tone(speaker, frequentie[i], 65);
    delay(duur[i]-65);
  }
}

void loop(){} // loop is verplicht, ook al is hij leeg!
```

Als je op de resetknop van de Arduino drukt speelt het melodietje nogmaals af.

6.3 Relais

Microcontrollers kunnen maar een klein beetje stroom leveren aan de uitgangen. Ongeveer 40 mA. Je kunt er een paar LEDjes op laten branden maar geen 100. Als je iets wilt aansluiten dat meer stroom nodig heeft dan kun je een relais gebruiken. Een relais is een schakelaar die je niet met je vinger bedient, maar met een klein stroompje, bijvoorbeeld uit de microcontroller.

Een relais-module

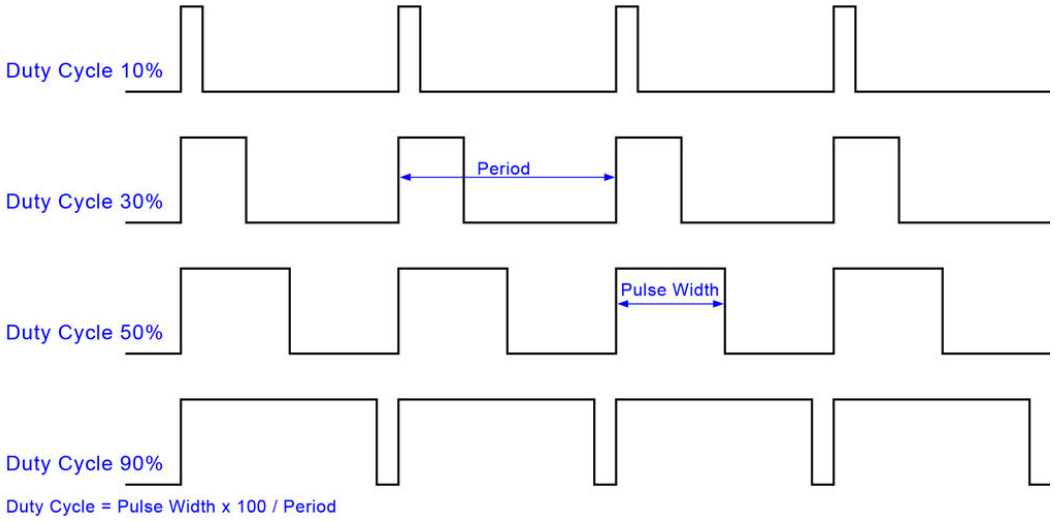


Schema van de relais- module

Dat er een écht schakelaartje in het relais zit kun je horen. Het ding maakt een tikje als het aan- of uitschakelt. Omdat de schakelcontacten van het relais elektrisch gezien los staan van de signaal-ingang kun je ook andere spanningen schakelen, zoals bijvoorbeeld 230 Volt wisselstroom. De motor, het verwarmingselement en de pomp van een wasmachine worden allemaal via relais bestuurd. Dat zijn de tikjes die je af en toe uit de wasmachine hoort komen.
De drie schakelcontacten zijn vaak met schroefklemmen uitgevoerd zodat je dikke kabels kunt vastschroeven. Het middelste contact is meestal de common, afgekort C. De andere twee zijn de NC (normal closed) en de NO (normal open). Als de stuurspanning op de ingang van het relais LOW (0 volt) is, is het C contact verbonden met het NC contact. Als de ingang HIGH (5 volt) is zijn de C en NO contacten met elkaar verbonden.

6.4 analogWrite en PWM

Als je een LED wilt dimmen of een motortje langzamer laten draaien, dan kun je dit doen door de spanning te verlagen. Maar de microcontroller kan alleen maar 0 volt of 5 volt aan een uitgang geven. Toch is er een truckje om met een microcontroller LED's te dimmen en motortjes langzamer te laten draaien. De microcontroller doet dit door een digitale uitgang heel snel achter elkaar aan en uit te zetten. Dit gebeurt met een frequentie van ongeveer 490Hz (490 keer per seconde aan en uit). De tijd dat de uitgang HOOG of LAAG is kun je variëren. We noemen dit pulsbreedte-modulatie of PWM (Pulse Width Modulation). De tijd dat de spanning hoog is noemen we de pulsbreedte. De totale tijd waarin de spanning één keer hoog en één keer laag is noemen we de periode. Bij de Arduino UNO is de PWM periode iets meer dan 2 milliseconden.
Is de pulsbreedte de helft van de periode, dan is de gemiddelde spanning ook de helft van 5 volt.



Met de opdracht analogWrite kun je de pulsbreedte instellen van 0 tot 255.

```
analogWrite(pin, 0);   // Maak uitgang pin 0 Volt.
analogWrite(pin, 255); // Maak uitgang pin 5 Volt.
analogWrite(pin, 127); // Maak de gemiddelde spanning 2,5 Volt.
analogWrite(pin, a);   // Gemiddelde spanning: (a/255)*5 Volt.
```

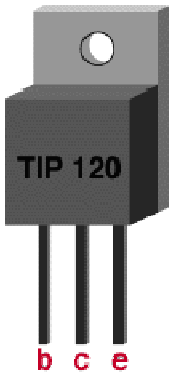
Als je een LED aansluit op een PWM signaal van 127, dan lijkt hij minder fel te branden. In werkelijkheid knippert hij. Dit gaat zó snel (490 keer per seconde) dat wij dat niet kunnen zien.
De PWM functie is niet op alle pinnen beschikbaar. Bij de Arduino UNO kun je alléén de digitale uitgangen 3, 5, 6, 9, 10 en 11 met pulsbreedte moduleren. Deze uitgangen herken je aan het ~ symbooltje op het Arduino board.

```
////////  Theelichtje   ////////////////

void setup(){
  pinMode(3, OUTPUT);   // LED op pin 3.
}

void loop(){
  analogWrite(3, random(100, 255));
  delay(random(50, 500));
}
```

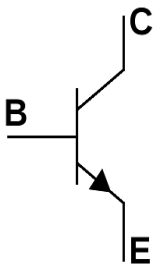
6.5 Motor

Een piepklein motortje (5Volt, max. 40 mA) kun je direct aansluiten op een digitale uitgang van de microcontroller. Een sterkere motor kun je aanschakelen met een relais. Wil je de snelheid van de motor kunnen regelen met PWM, dan heb je niks aan een relais. Het relais kan niet 490 keer per seconde schakelen. Hij is veel te traag. We gebruiken hiervoor een transistor. Een transistor kan duizenden malen per seconde 'schakelen'.
Een transistor is eigenlijk een versterkertje dat een klein stroompje kan versterken tot een grote stroom. Er zijn duizenden soorten transistoren. Het type wat wij gebruiken, de TIP120, kan gelijkstroommotoren tot wel 60 V en 5 A aan! Als er meer dan 1 A stroom door de motor (en dus ook door de transistor) loopt, dan moet je de transistor koelen met een koellichaam. Dat is een stuk aluminium met gleuven waardoor de warmte aan de lucht kan worden afgestaan.

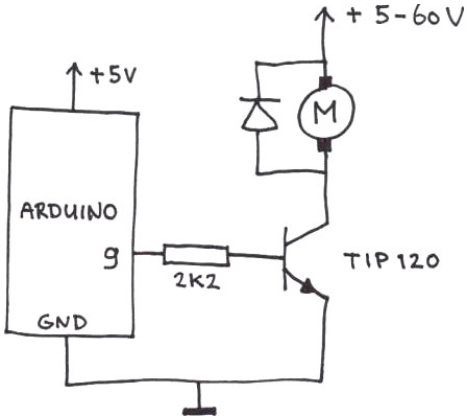
Zo werkt de transistor: De kleine stroom komt van de microcontrollerpin en wordt via een weerstand van 2,2k aangesloten op pin B (basis) van de transistor. De grote stroom loopt via de motor (het cirkeltje met de M) door de transistor van pin C (collector) naar pin E (emittor).



De TIP120 transistor



Het symbool van de transistor



De motor via een TIP120 aangesloten op de Arduino

Parrallel over de motor is een diode aangesloten. De diode is met de min-kant aangesloten op de plus van de voedingsspanning, dus gaat er normaal geen stroom lopen door de diode. Als de motor wordt uitgeschakeld (en dat gebeurt 490 keer per seconde) dan geeft de motor een spanningspiek met de plus aan de kant van de transistor. Hierdoor kan de transistor stuk gaan. De diode beschermt de transistor door de stroom terug naar de + te laten lopen in plaats van door de transistor.

6.6 Servomotor

Een servomotor draait niet volledig rond, maar kan maximaal 180 graden draaien. Je kunt de hoek waarover hij draait instellen, en zo de as in een bepaalde positie zetten. Je kunt met een servomotor bijvoorbeeld het stuur van een modelauto, het roer van een modelboot of -vliegtuig besturen, maar ook bijvoorbeeld de stand van een camera onder een quadcopter drone besturen.
In de servo zit een elektromotor en een aantal tandwielen om de motor te vertragen en zodoende de kracht van de servo-arm te vergroten. Op de as van de servo zit tevens een potmeter gemonteerd, zodat de stand van de arm kan worden doorgegeven aan de interne elektronica. De elektronica laat het motortje draaien tot de juiste stand bereikt is.
De servomotor heeft drie aansluitingen. De plus (meestal rood), de min (GND, meestal zwart) en de stuur-ingang (meestal wit of geel).
De stuuringang van de servo kan direct op een digitale pin van de Arduino worden aangesloten.

Een servomotor of servo

Een robotarm met 5 servo's

Sluit een servo aan op pin 3 van de Arduino en probeer de volgende sketch:

```
////////   Servo test   ////////////////////////////////

#include <Servo.h>
Servo servo3;  // Definieer een servo en noem hem servo3.

void setup(){
  servo3.attach(3);   // servo3 komt op pin 3.
}

void loop(){
  servo3.write(0);   // Zet servo3 in positie 0 graden.
  delay(1000);
  servo3.write(90);  // Zet servo3 in positie 90 graden.
  delay(1000);
  servo3.write(180); // Zet servo3 in positie 180 graden.
  delay(1000);
  servo3.write(random(180)); // servo3 in random positie.
  delay(3000);
}
```

Als je een servo gebruikt kun je niet tegelijk PWM signalen sturen via pinnen 9 en 10. Ook al staat de servo op andere pinnen. Je kunt met de Arduino UNO maximaal 12 servo's tegelijk aansturen. Wil je meer servo's gebruiken, dan kun je een Arduino Mega gebruiken. Met de Mega kun je 48 servo's aansturen.
Niet alle servomotoren werken goed met de Arduino.

Vragen en opdrachten

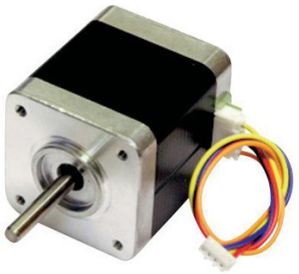
24. Ontwerp een sketch en een schakeling waarmee je door aan een potmeter te draaien een servo kunt instellen tussen 0 en 180 graden. Teken eerst het schema voordat je gaat bouwen.

6.7 Stappenmotor

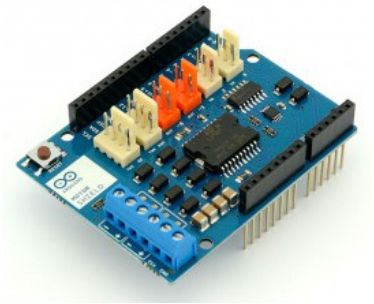
Een stappenmotor heeft twee paar magneten en spoelen die in de juiste volgorde aan en uit geschakeld moeten worden om de stappenmotor een stapje te laten draaien. Een stapje is altijd precies een bepaalde hoek, bijvoorbeeld 1,8 graden. Laat je de stappenmotor 100 stappen maken, dan draait de as dus over een hoek van 180 graden. Laat je de motor 2000 stappen maken, dan draait de as precies 10 omwentelingen.

In een printer wordt een stappenmotor gebruikt om de printkop naar de juiste positie boven het papier te brengen. Een 3D-printer heeft meestal 4 stappenmotoren, drie voor de x, y en z posities en één om het plastic door de extrusiekop te persen. Een industriële robotarm heeft zware stappenmotoren om de arm zeer precies te manoeuvreren.

Om een stappenmotor te besturen met de Arduino heb je een speciaal IC - een H-bridge - nodig wat tussen de Arduino en de stappenmotor moet. Via de Arduino IDE Help>Referece>Libraries>stepper kun je gedetailleerde informatie en voorbeelden vinden. Als dit te moeilijk is kun je de hardware ook als een complete motor-shield kopen. In hoofdstuk 7.1 kun je meer lezen over shields.



Een stappenmotor



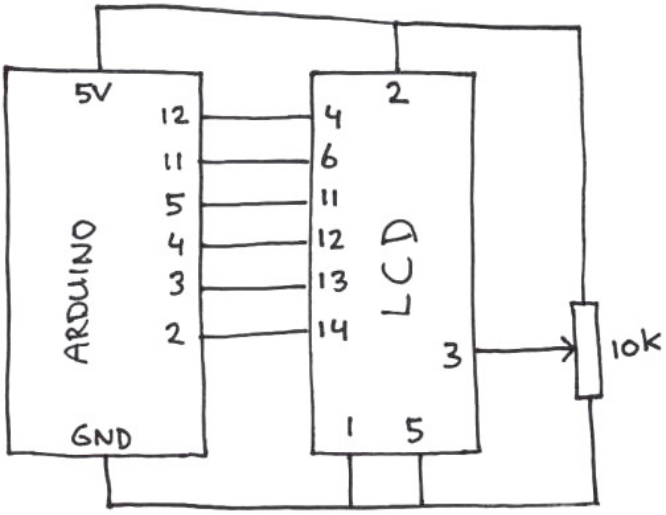
Het Arduino motor-shield

6.8 Liquid Crystal Display

Wil je tekst en gegevens zien zonder de Arduino via een USB kabel met de computer te verbinden, dan kan je een LCD aansluiten. Arduino's LiquidCrystal library ondersteunt displays die werken volgens het Hitachi HD44780 protocol. Er zijn veel displays die werken met dit protocol. Je herkent ze aan de 16 aansluitingen. De meest voorkomende displays zijn die met 2 regels van 16 karakters en die met 4 regels van 20 karakters.
Het display heeft 6 digitale signalen van de Arduino nodig. Je moet dus 6 digitale uitgangen van de Arduino gebruiken om de LCD aan te sluiten.

Een LCD display met 4 regels van 20 karakters

Sluit een LCD met twee regels van 16 karakters aan volgens het volgende schema:



Met de potmeter kun je het contrast regelen. De meeste displays hebben ook LED's ingebouwd voor de achtergrondverlichting. Deze vind je meestal op pinnen 15 en 16. Kijk maar in de datasheet hoe je die precies moet aansluiten.

Pin No.	Symbol	Level	Description
1	VSS	0V	Ground.
2	VDD	+5.0V	Power supply for logic operating.
3	V0	--	Adjusting supply voltage for LCD driving.
4	RS	H/L	A signal for selecting registers: 1: Data Register (for read and write) 0: Instruction Register (for write), Busy flag-Address Counter (for read).
5	R/W	H/L	R/W = "H": Read mode. R/W = "L": Write mode.
6	E	H/L	An enable signal for writing or reading data.
7	DB0	H/L	This is an 8-bit bi-directional data bus.
8	DB1	H/L	
9	DB2	H/L	
10	DB3	H/L	
11	DB4	H/L	
12	DB5	H/L	
13	DB6	H/L	
14	DB7	H/L	
15	LED+	+5.0V	Power supply for backlight.
16	LED-	0V	The backlight ground.

Tabel uit de datasheet met pinfuncties van de display

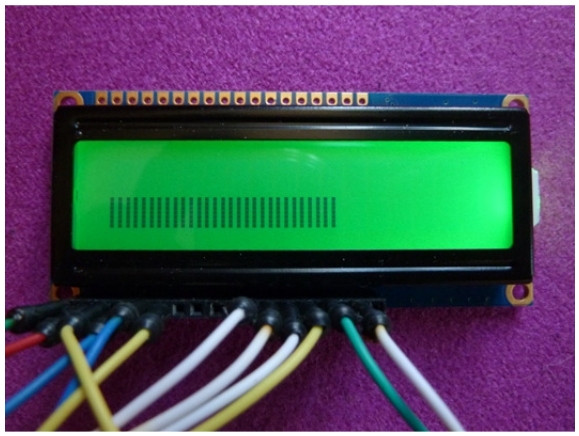
Met de volgende sketch kun je testen of het display werkt.

```
//////////  LCD test  ////////////////////////////////////////////////////

#include <LiquidCrystal.h>              // De standaard library voor LCD's.
LiquidCrystal lcd(12, 11, 5, 4, 3, 2); // De Arduino pinnen die worden aange-
                                        // sloten op (RS, E, D4, D5, D6, D7)
void setup(){
  lcd.begin(16, 2);             // Display met 2 regels van 16 karakters
  lcd.print("SECONDEN AAN:");   // Tekst schrijven naar het display
}

void loop(){
  lcd.setCursor(0, 1);          // Cursor naar kolom 0, rij 1 (onderste regel)
  lcd.print(millis()/1000);     // Een variabele printen.
}
```

Elk karakter van het display bestaat uit 7 rijen van 5 pixels. Je kunt ook je eigen karakters maken, zoals een smiley of een rechthoekje.
Je kunt maximaal 8 eigen karakters definiëren.

```
////////  Je eigen LCD karakter maken   ////////////////////////////////////

#include <LiquidCrystal.h>                // Gebruik de LCD library.
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);    // LCD(RS, E, D4, D5, D6, D7) op
                                          // Arduino pin (12, 11, 5, 4, 3, 2).
byte hokje[] ={       // Definieer de byte array hokje.
  B11111,             // Elke byte variabele vormt een rijtje pixels van het
  B10001,             // 5*7 pixel rooster van het karakter. Door de variabelen
  B10001,             // onder elkaar te zetten kun je goed zien welke pixels
  B10001,             // van het karakter aan (1) of uit (0) staan.
  B10001,
  B10001,
  B11111
};

void setup(){
  lcd.createChar(1, hokje);    // Maak karakter 1 met de array hokje.
  lcd.begin(16, 2);            // Display van 2 regels met 16 karakters.
  lcd.write(1);                // Karakter 1 printen.
}

void loop(){}
```

Een analoge waarde van een sensor kun je op het display weergeven met een rij blokjes van 0 tot 16 blokjes. Je kunt de resolutie ook verhogen naar 48 streepjes. Hiervoor moet je eerst drie karaktertjes maken: met één streepje, met twee streepjes en met drie streepjes.



Opdracht 25

Open de sketch **`streepjes.ino`**.
Sluit een 16*2 display aan volgens de aangegeven pinnen.
Sluit een potmeter aan tussen GND en 5V.
Sluit de loper aan op pin A0 van de Arduino.
Upload de sketch.
Draai aan de potmeter om het aantal streepjes op het display te veranderen.
Bestudeer de sketch. Snap je hoe het werkt?

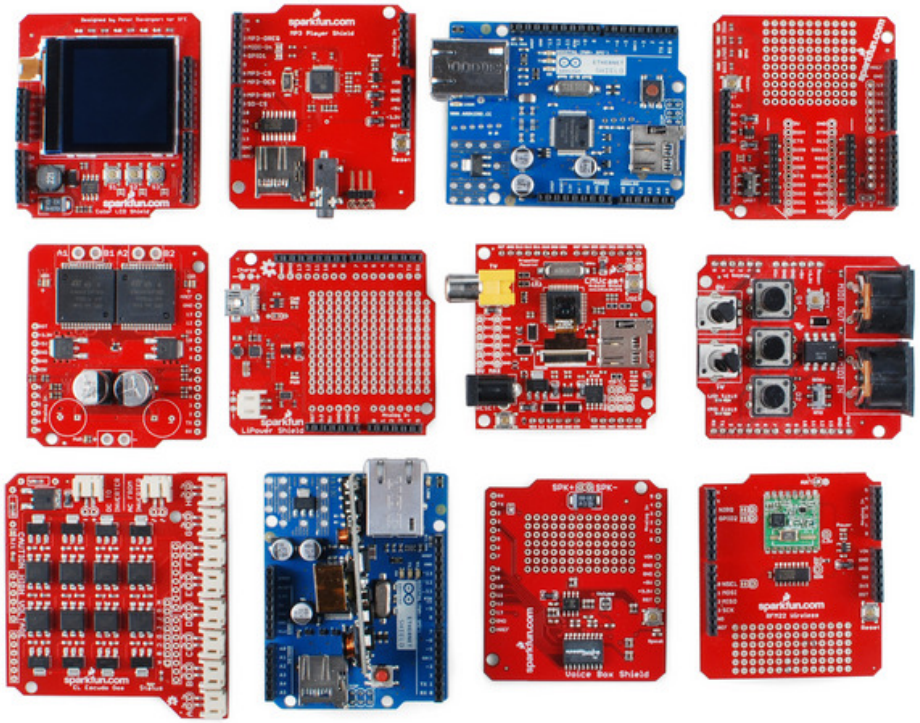
6.9 Libraries

De servomotor, de stappenmotor en het LCD maken allemaal gebruik van libraries. Een library is een stuk programma wat iemand al voor je heeft geschreven. Je kunt het programma in je eigen sketch opnemen met de opdracht **`#include <>`**. Om de LCD te kunnen gebruiken moet je de LiquidCrystal library gebruiken: **`#include <LiquidCrystal.h>`**.
Zoek via de Arduino IDE Help>Referentie>Libraries de LiquidCrystal library op. Je ziet dat er heel veel opdrachten met betrekking tot de LCD zijn. Je kan je eigen karakter maken, of je kan tekst laten scrollen.
Een library voegt dus allerlei mogelijke opdrachten toe aan de IDE.
Er zijn veel meer libraries dan de standaard libraries die in de IDE te vinden zijn. Je kunt ze op het internet vinden. Als je zo'n library wilt gebruiken moet je hem eerst downloaden en in de libraries folder van de Arduino map zetten en daarna de IDE opnieuw opstarten.

7 Bouwen

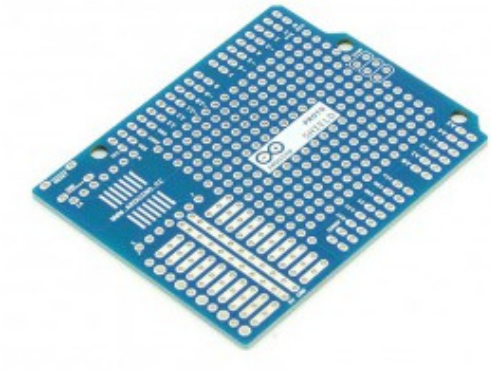
7.1 Shields

Een shield is een board met hardware voor een bepaalde toepassing wat je direct op de Arduino kunt stapelen. Op die manier zijn direct alle pinnen van de Arduino verbonden met de hardware van de shield. Er zijn veel shields te koop, zoals een ethernetshield, GPS-shield, motorshield, MP3-shield, WIFI-shield, LCD-shield, TFT-shield, RFID-shield, SD-shield, GSM-shield, enzovoort.

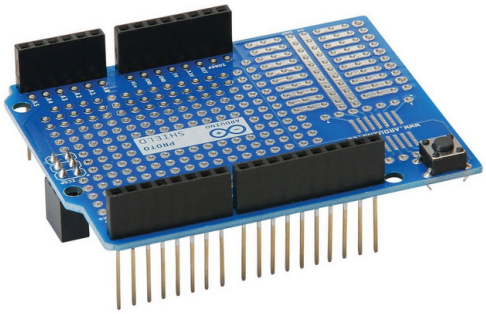


Er zijn veel verschillende shields te koop

Een proto-shield is een leeg shield waar je zelf je hardware prototype op kunt solderen. Je moet er wel eerst de pinnen op solderen. Deze pinnen noem je headers. Je kunt een speciale set headers voor je protoshield kopen. Deze headers hebben niet alleen pinnen, maar ook bussen aan de bovenkant zodat je er stekkertjes in kan prikken of zelfs meerdere shields op elkaar stapelen.



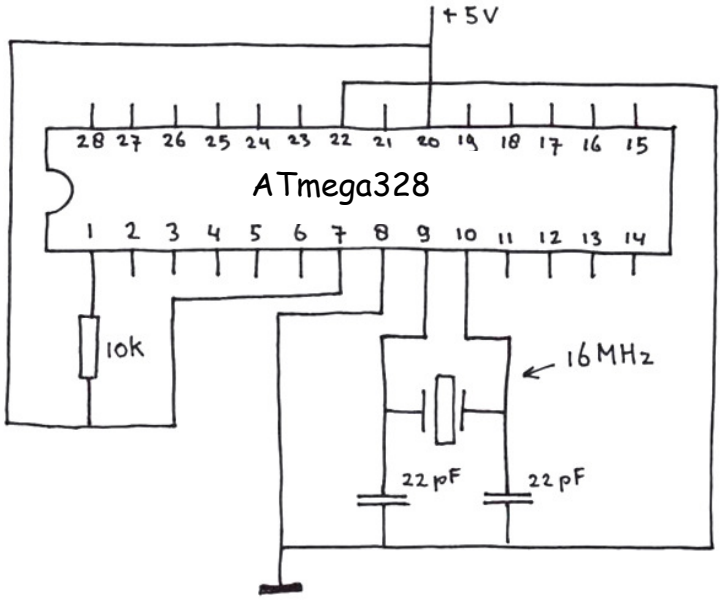
Het Arduino UNO proto-shield



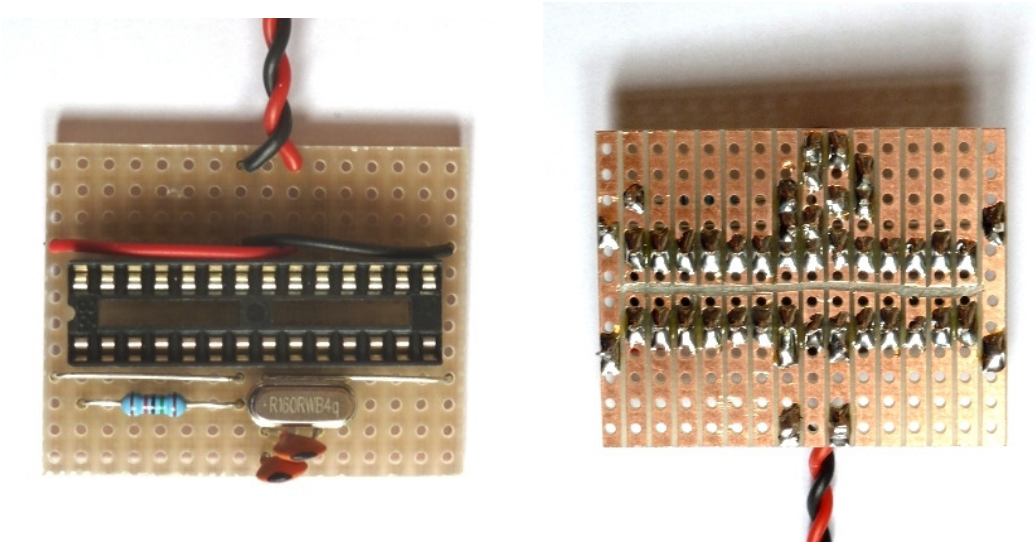
Het Arduino proto-shield met gesoldeerde headers

7.2 Barebone

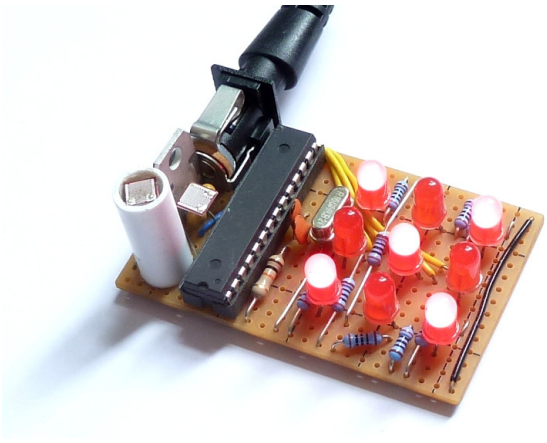
De ATmega328 microcontroller werkt ook zelfstandig, buiten het Arduino board. Na het uploaden van de sketch kun je de microcontroller heel voorzichtig uit het voetje halen. Pas op dat je de pootjes niet krom buigt!
Om het IC buiten het Arduino board te laten werken heb je ook een IC-voetje, een 16 MHz kristal, 2 keramische condensatortjes van 22 pF en een weerstand van 10k nodig. Deze onderdeeltjes zijn samen met een ATmega328 als barebone-kit te koop voor ongeveer € 7,50. De ATmega328 in de barebone-kit is voorzien van de Arduino-bootloader. Deze bootloader is een stukje software wat in de microcontoller geladen is om hem met de Arduino IDE te kunnen laten werken.
Verder heb je natuurlijk een 5 Volt DC voeding nodig.
De pinnen van de ATmega328 zijn genummerd van 1 tot en met 28. Tussen de pinnen 1 en 28 is een inkeping gemaakt.
Op bladzijde 8 kan je zien op welke pinnen de analoge en digitale in- en uitgangen te vinden zijn.
Zo sluit je het kristal, de 2 condensators, de weerstand en de voedingsspanning op het IC aan:



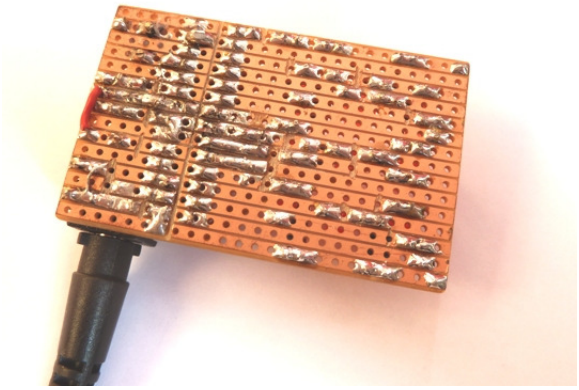
Soldeer niet aan de pinnen van het IC, maar gebruik altijd een IC voetje. Soldeer het voetje (met de inkeping aan de goede kant) op een stuk rasterboard met een 2,54 mm lijnenraster van 16 koperbanen breed. Dan heb je aan beide kanten naast het IC een extra baan voor de GND en de +5V.
Onderbreek de koperbanen onder het IC voetje door ze met een scherp mes door te snijden, of met een multitool door te slijpen.



Barebone (zonder de ATmega328) op een stuk 2,54 mm lijnraster board



Een barebone-dobbelsteen



Je moet wel een beetje kunnen solderen!

7.3 Project ideeën

Op het internet zijn honderden Arduino projecten te vinden. Compleet met bouwbeschrijvingen, sketches Fritzing tekeningen en Youtube tutorials.
In de praktijk wil je jouw ontwerp meestal nét iets anders maken, bijvoorbeeld omdat je andere onderdelen hebt of andere ontwerpeisen.
Als je dit boek hebt doorgewerkt ben je in staat om je eigen ontwerp te maken en projecten van anderen aan te passen aan jouw situatie. Hieronder vindt je dus geen compleet uitgewerkte projecten maar alleen wat ideeën die je zelf kunt uitwerken.
Je hebt zelf vast veel meer ideeën.

AA Alkaline batterij tester

Met een batterijtester kun je meten hoeveel energie er nog in een batterij zit. Als een batterij leeg raakt wordt de spanning lager tijdens het gebruik. Sluit je de batterij aan op een analoge ingang, dan kan je de spanning meten.
Maar als een batterij geen stroom hoeft te leveren is de spanning van een (lege) batterij veel hoger dan wanneer hij wel stroom moet leveren. Daarom moet de batterij tijdens het testen stroom leveren. Parrallel aan de te testen batterij moet een weerstand van 4,7Ω (0,5W) worden aangesloten. De batterij moet eerst 5 seconden stroom leveren voordat de spanning wordt gemeten.
Met een prototype kun je uitzoeken welke spanning een batterij geeft als hij vol, bijna vol, half vol, bijna leeg of helemaal leeg is.

Diodetester

Een diode laat de stroom maar in één richting door. Het streepje op de diode geeft de min aan (zie foto en symbool op blz. 14). Soms is het streepje maar moeilijk te zien. Een LED is ook een diode. De korte poot is de min, maar wat als de pootjes even lang zijn afgeknipt?
Met een diodetester kun je testen of een diode nog goed is, en wat de plus- en de min-kant zijn.

Vertragingsschakelaar

Als je 's middags je fiets in de fietsenkelder zet en je vergeet het licht van de kelder achter je uit te doen, dan brandt het licht tot je de volgende morgen weer je fiets pakt (als je dan niet wéér vergeet om het licht uit te doen). Om te voorkomen dat het licht onnodig brandt kun je een vertragingsschakelaar inbouwen.
Met een enkel drukknopje doe je het licht aan en uit. Als het licht vijf minuten aan is en je hebt nog niet op het knopje gedrukt, dan gaat het vanzelf weer uit.

Badkamerventilator

Tijdens het douchen ontstaat er veel waterdamp. De lucht in de badkamer wordt heel erg vochtig. Direct na het douchen is de badkamer gevuld met vochtige lucht. Het kan wel een half uur duren voordat het ergste vocht met een ventilator weggezogen is. De badkamerventilator zou eigenlijk pas een half uur na het douchen uit moeten. Jij denkt er na een half uur niet meer aan om hem uit

te zetten, dus moet je een automatische badkamerventilator maken. Een soort vertragingsschakelaar dus.

Automatisch buitenlicht

Je hebt het vast wel eens gezien. Loop je ergens langs een voordeur, gaat er ineens een lamp aan. Een automatisch buitenlicht gaat aan als er iemand in de buurt komt. Het werkt met een PIR-sensor (Passieve Infra-Rood sensor). De sensor geeft een signaal als er een verandering is in het infra rood licht. Natuurlijk hoeft het buitenlicht alleen aan te gaan als het buiten donker is!

Stroboscoop

Een stroboscoop is een knipperlicht waarvan je de frequentie kunt instellen. Leuk voor een feestje, maar je kan het ook als wetenschappelijk instrument gebruiken.
Een stroboscoop kan worden gebruikt om snelle bewegingen te analyseren. Als je wilt zien hoe een naaimachine werkt kun je de snel bewegende naald met de stroboscoop belichten. Als de frequentie van de srtroboscoop gelijk is aan de frequentie van de naald, dan lijkt het alsof de naald stil staat. Als de frequentie van de stroboscoop een klein beetje lager is dan die van de naald, dan lijkt het alsof de naaimachine heel langzaam beweegt. Op die manier kun je goed zien hoe hij beweegt. Een wetenschappelijke stroboscoop heeft natuurlijk ook een display waarop je de frequentie kan aflezen. Op die manier kun je de stroboscoop gebruiken om bijvoorbeeld het toerental van een motor te meten.
Helmaal mooi wordt het als je een superfelle witte LED neemt (een CREE-LED bijvoorbeeld). Zo'n LED gebruikt teveel stroom om direct te worden aangesloten op een uitgang van de controller dus moet je hem aansluiten via een sterke transistor (bijvoorbeeld een TIP120, zie blz. 32).

Kwismaster

De kwismaster stelt een vraag. De kandidaat die het eerst op de knop drukt mag de vraag beantwoorden. Maar wie drukt het eerst op de knop? Dat kan de Arduino razendsnel en eerlijk vaststellen. Bij de kandidaat die het eerst drukt gaat een lampje branden. Bij de andere kandidaten werkt de knop nu niet meer (of mag de kandidaat die als tweede gedrukt heeft antwoorden?). Misschien kun je ook een toetergeluid programmeren als iemand drukt. En de quizmaster moet natuurlijk een resetknop hebben voor de volgende vraag.

Plantenbegieter

Vergeet jij ook altijd je planten water te geven? Met twee roesvrijstalen pinnen op een vaste afstand (bijvoorbeeld 1 cm) in de grond geprikt kun je meten hoe vochtig de aarde is. Vochtige aarde heeft een lage weerstand, droge aarde heeft een hoge weerstand.
De automatische plantenbegieter meet een paar keer per dag de weerstand. Is de weerstand te hoog, dan krijgt de plant een scheutje water.
Maak het systeem wel veilig, zodat je geen overstromingen krijgt....

Huisdiervoederautomaat

Je gaat op vakantie, maar er is niemand die voor je huisdier kan zorgen.
Ontwerp een machine waarmee je jouw huisdier twee weken kan verzorgen.

Inbraakalarm

Met een automatisch buitenlicht kun je inbrekers afschrikken. Als ze dan toch een raam of een deur openbreken, dan kan je een alarm laten afgaan.
De ramen en deuren kunnen worden voorzien van sensoren, bijvoorbeeld een reed-relais op de deurpost en een magneetje op de deur. Mischien heb je zelfs een PIR-sensor tot je beschikking. Je moet het alarm wel zelf kunnen uitzetten. En als je niet thuis bent is het wel fijn voor de buren als het alarm niet langer dan een minuut afgaat.

Kilometerteller

Een kilometerteller op een fiets werkt met een magneet en een reed-relais. De magneet is aan een spaak bevestigd, het reed-relais is aan de vork bevestigd. Elke keer als de magneet langs het reed-relais komt schakelt het relais eventjes aan. Elke keer als het reed-relais schakelt is het wiel dus één keer rond gedraaid. Uit de omtrek van het wiel en het aantal pulsen van het reed relais kun je de afgelegde afstand uitrekenen. Op een display kun je de afgelegde afstand aflezen.

Snelheidsmeter

Snelheid is afstand gedeeld door tijd.
Deel de omtrek van het wiel in mm door de tijd die het wiel nodig heeft om één keer rond te draaien in milliseconden. Je krijgt dan een snelheid in mm/ms. Dat is hetzelfde als m/s.

Reken de snelheid desgewenst om in km/u.

Elektronische schuifmaat
Met een lineaire (schuif-)potmeter kun je een elektronische schuifmaat maken.

Windsnelheidsmeter
Een molentje draait in de wind. De snelheid waarmee het draait is een maat voor de windsnelheid. Met een lichtsluis kun je zonder contact met de bewegende delen het aantal omwentelingen per tijdseenheid meten. Hieruit kan je de windsnelheid berekenen.

Windrichtingmeter
Op de as van een windvaan monteer je een magneetje. Rond de as met de magneet plaats je 4 of 8 reed-relais. Afhankelijk van de windrichting wordt één van de reed-relais gesloten.
Een windvaantje wappert meestal wat heen en weer. Je kunt een hogere resolutie krijgen door in je sketch steeds de gemiddelde windrichting over een periode van bijvoorbeeld 30 seconden te berekenen.

Alcohol blaastest
De alcohol die iemand drinkt wordt via de maag in het bloed opgenomen. Via het bloed verspreidt de alcohol zich over de waterige weefsels van het lichaam zoals de hersenen en de longen.
Via de longen wordt een gedeelte van de alcohol uitgeademd. Hoe meer alcohol in het bloed, des te meer ook in de adem. Bij een promillage van 0,5 is de concentratie in de adem 220 µg/L.
Een TGS822 of TGS2020 gassensor reageert op alcoholdamp. Hoe hoger de alcoholconcentratie, des te lager de weerstand. Met zo'n sensor kun je dus meten hoeveel alcohol in de adem zit, en dit omrekenen naar een hoeveelheid alcohol in het bloed.
Je kunt de blaastest ijken met een AAC longsimulator. (http://www.nano2.nl/longsimulator.pdf)

Index