Verwerken en beslissen

21st Dec, 2024

created in Curvenote

Keywords NLT, schakelmodule, digitale technologie

1 Leerdoelen

Kennis 4.1 Je kunt in grote lijnen beschrijven hoe een computerchip gemaakt wordt en welke materialen daarvoor gebruikt worden 4.2 Je kunt uitleggen wat 'programmeren' van een computerchip inhoudt en welke rol transistoren daarin hebben. 4.3 Je kunt uitleggen dat computers een computertaal hebben die uit enen en nullen bestaat en dat programmeertaal hierin omgezet moet worden door een compiler

2 Vaardigheden

4.4 Je kunt een programma en bijbehorende drivers voor de Arduino (of vergelijkbare microcontroller) downloaden, bewerken, uploaden en testen. 4.5. Je kunt de belangrijkste commando's voor de Arduino die je nodig hebt voor deze module gebruiken en uitleggen waar ze voor bedoeld zijn. 4.6 Je kunt uitleggen dat programmeertalen (zoals Matlab, C++, Python) ontworpen zijn voor specifieke toepassingen en dat ze ieder sterke en zwakke punten hebben. 4.7 Je bent in staat bronnen te vinden waarmee je verschillende programmeertalen kunt leren.

Digitale technologie bestaat niet zonder microchips. Na de uitvinding van de transistor is de electronica in een snel tempo kleiner en krachtiger geworden. We geven je in dit gedeelte een idee hoe die chips gemaakt en geprogrammeerd worden. Als je meer wilt weten kun je terecht in bijlage 3 in de moduledatabase.

Digitale apparaten verwerken informatie en beslissen wat er met die informatie moet gebeuren. Dat gebeurt allemaal in chips, waarvan een processor er ook een is. Dat is de hardware. Informatie kan opgeslagen worden, doorgestuurd of naar actuatoren (scherm, printer, luidspreker etc.). De chip kan een vaste bedrading hebben, waardoor telkens dezelfde bewerking wordt gedaan. Een programmeerbare chip kan telkens andere taken of bewerkingen uitvoeren. De instructies hiervoor zijn gecodeerd in de software. We kijken hoe die chips gemaakt worden.

2.1 4.1 Fabricage van electronica

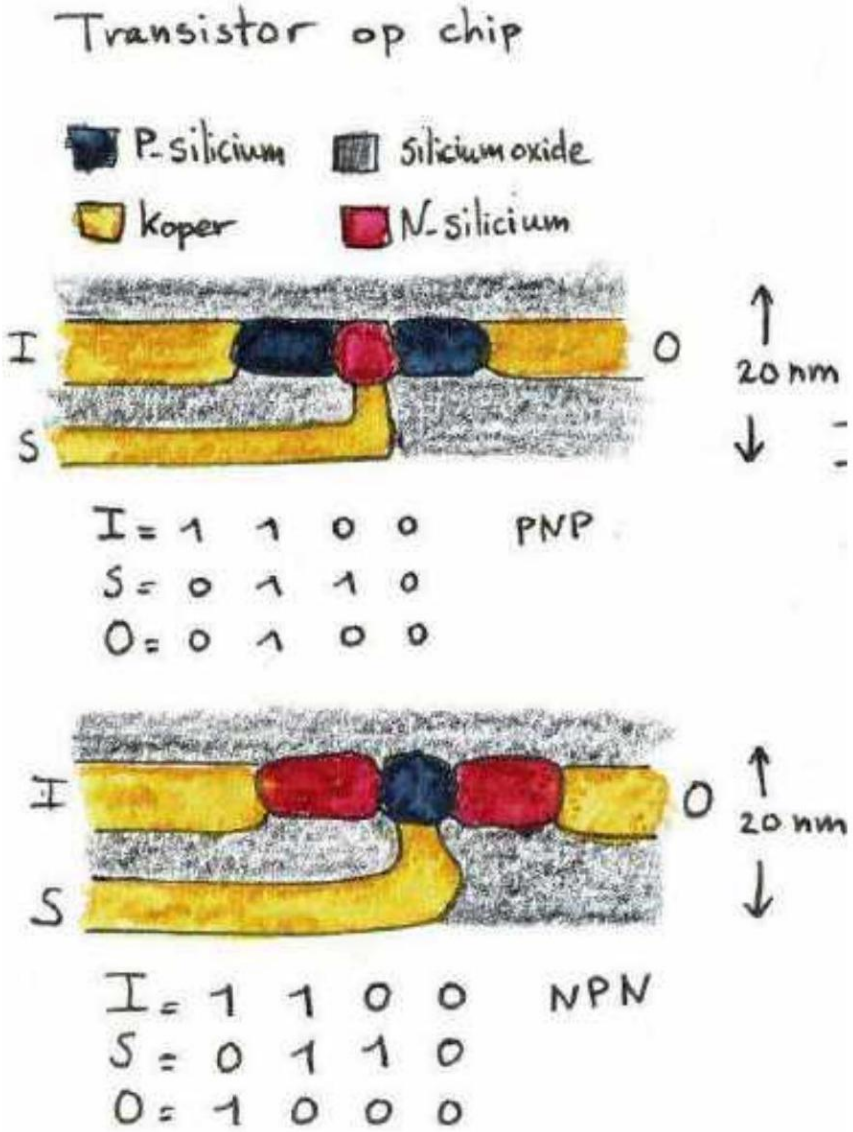
Computerchips worden met tientallen tot honderden tegelijk gemaakt op ronde schijven zuivere silicium van 1 mm dik (wafers). Dat gebeurt in superschone ruimten (cleanrooms). Op de siliciumwafer worden tientallen superdunne laagjes aangebracht: metaal, siliciumoxide en mengsels van silicium en andere elementen. Dit is de Metaal, Oxide, Silicium (MOS) techniek, waarmee het overgrote deel van de computerchips en sensoren gemaakt worden. De siliciumwafer is aan het begin 1 mm dik. Ieder laagje is zo'n 30 nm dik. De wafer is na al die laagjes maar 2000 nm dikker, dus $1,002\ \mathrm{mm}$. Een computerchip is meestal niet groter dan $1 \times 1\ \mathrm{cm}$, dus er kunnen tientallen tot honderden chips op een enkele wafer gemaakt worden. De wafer wordt daarna in kleine stukjes ('chips') gezaagd. Ze worden dan bevestigd in houders met metalen pinnetjes om ze op een printplaat te kunnen aansluiten.

3 Onderdeeltjes op nanoschaal

Figuur 4.1 Een productieruimte voor computerchips (clean room)

De patronen in laagjes van de chip vormen transistoren, verbindingsdraadjes en diodes. De lijntjes en blokjes van metaal, oxide en silicium hebben een breedte en dikte van enkele tot tientallen nanometers (nm , een miljoenste millimeter). Stel dat je met verf heel precies een lijn wilt schilderen, dan doe je afplakband (maskeertape) langs de randen. Op een wafer is het allemaal zo klein, dat daarvoor een speciale techniek gebruikt wordt: lithografie. De wafer krijgt een dun laagje UVgevoelige lak. Wanneer deze lak droog is, kan die weer zacht gemaakt worden met UV-licht. Om het UV-licht precies op de goede plek te krijgen gebruikt men een 'masker'. Een masker is een doorschijnende zwart-wit afbeelding met de verbindingen en onderdeeltjes voor de chip. De afbeelding van het masker wordt met UV op een stukje van $1 \times 1\ \mathrm{cm}$ op de lak geprojecteerd. Dat gebeurt voor alle chips op een wafer achter elkaar in een chipmachine. Elk laagje op de chip

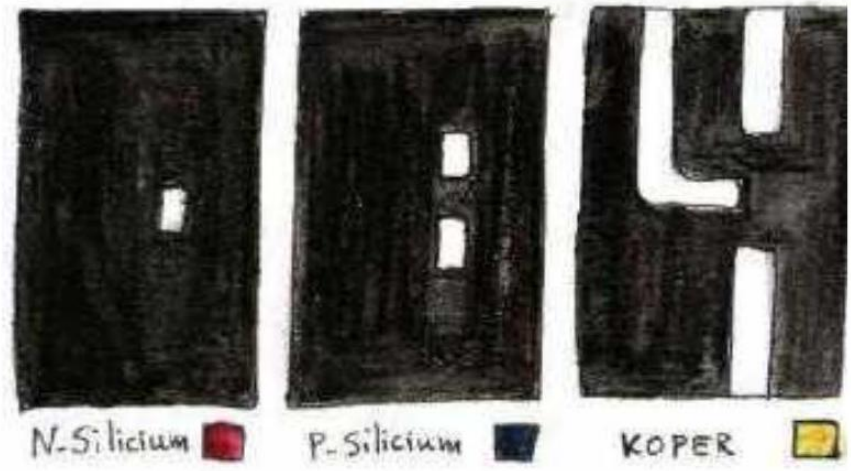
heeft een eigen masker, dus de machine moet een hele serie maskers na elkaar op dezelfde wafer projecteren.



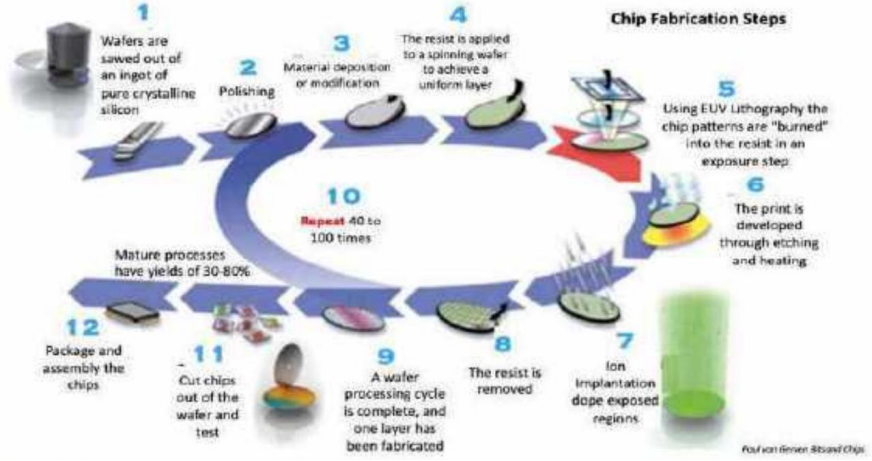
Figuur 4.2 Laagjes metaal, silicium (met n of p toevoegingen) en de verschillende maskers die voor elke laag nodig zijn om een transistor te maken. De structuren zijn enkele nanometers breed.

Realiseer je goed: er zitten tientallen chips op een wafer en elke wafer krijgt tientallen laagjes die tot op de nanometer precies op elkaar moeten zitten. Een wafer moet dan heen en weer geschoven worden tijdens het belichten, naar andere machines voor spoelen, opdampen van metaal en lakken en dan weer terug voor de volgende belichting en je wilt graag heel veel chips achter elkaar kunnen maken.

De machines die dat snel en goed kunnen zijn extreem ingewikkeld en kostbaar. Het Nederlandse bedrijf ASML uit Veldhoven loopt voorop als het gaat

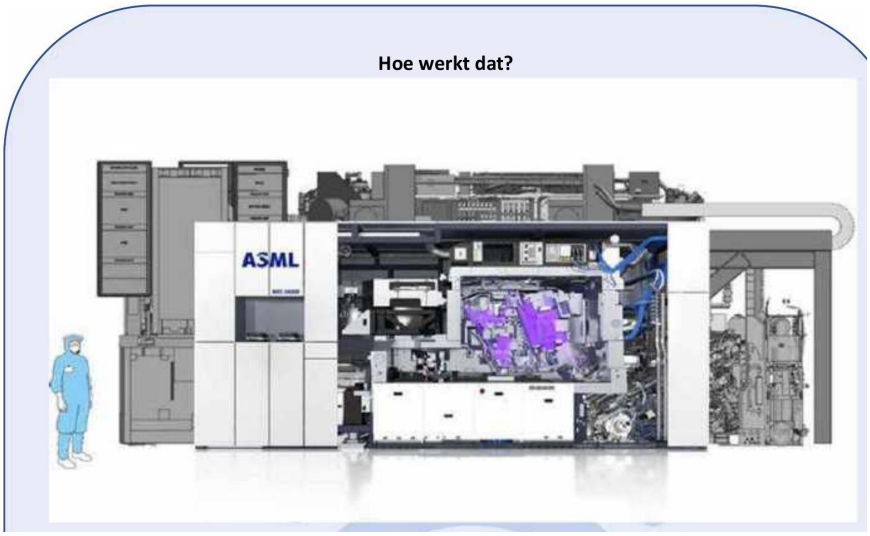


Figuur 4.3 Schema van klein stukje belichtingsmaskers in chipmachine voor onderdelen van één transistor



Figuur 4.4 Productiestappen voor een wafer met microchips om het maken van deze chipmachines. Ongeveer driekwart van alle chips wordt gemaakt met de machines van ASML (zie kader). Ze staan vooral in fabrieken in Taiwan, Zuid-Korea en de Verenigde Staten. Daar wordt het overgrote deel van de mi-

crochips gemaakt. Inbouwen van deze microchips in printplaten gebeurt op veel meer plekken in de wereld.



Voor het belichten van chips zijn uiterst precieze Lithografiemachines nodig. Dat zijn zeer dure en grote apparaten, die door 80% van alle chipfabrikanten gekocht worden bij het Nederlandse bedrijf ASML. Een masker met een afmeting van zo'n 30×30 cm wordt verkleind afgebeeld op de lak van de wafer. Voor het belichten wordt UV licht gebruikt. Hoe korter de golflengte van het licht, hoe kleiner de afbeeldingen gemaakt kunnen worden. De nieuwste machines die ASML maakt gebruiken EUV (Extended Ultraviolet, met een golflengte van 13,5 nm). De 'lamp' bestaat uit een krachtige laser die minuscule druppeltjes tin verhit die zo EUV gaan uitstralen. Daarmee kunnen lijntjes geprojecteerd worden die enkele nm uit elkaar liggen. Zo kunnen superkleine onderdelen op de chip worden gebouwd. Hoe kleiner de onderdelen, hoe sneller de chips kunnen schakelen. Elektronen hoeven minder ver te reizen, zodat energieverlies en warmteproductie minder zijn. Er kunnen méér bouwsteentjes op de chip gezet worden, zodat die meer bewerkingen kan uitvoeren en ook meer informatie kan opslaan. Volgens de wet van Moore verdubbelt elke 10 jaar de hoeveelheid transistoren op een chip. Met de nieuwste productietechnieken lukt het om die lijn vast te houden. Maar er is een grens aan: de afmeting van een atoom kun je niet kleiner maken. Kijk hier hoe een chipmachine van ASML werkt (link, url).

Buurten bij ASML - Wat is een chip? Hoe werkt een chipmachine? (https://youtu.be/5YJHwgoE3pE)

How it's made Microchips ASML (https://youtu.be/C4gYf-eaZTE) EUV: Lasers, plasma, and the sci-fi tech that will make chips faster | Upscaled (https://youtu.be/oliqVrKDtLc)

De chipmachine moet afbeeldingen kunnen maken waarbij lijntjes enkele nanometers uit elkaar liggen. Op die manier lukt het om de schakelingen op de chip op een fractie van een vierkante millimeter bij elkaar te persen. Met de nieuwste E-UV chipmachines kunnen draadjes en transistoren van enkele nanometers dikte gemaakt worden, waardoor er miljoenen bouwsteentjes op een vierkante millimeter passen. Klein is vooral belangrijk voor de snelheid: de elektronen in de chip leggen dan korte afstandjes af. Méér onderdelen betekent dat de chip meer bewerkingen tegelijk kan doen en meer geheugen heeft.

In de bijlagen in de moduledatabase is veel meer te lezen over technieken die bij de productie van chips gebruikt worden en de techniek achter transistoren en logische schakelingen.

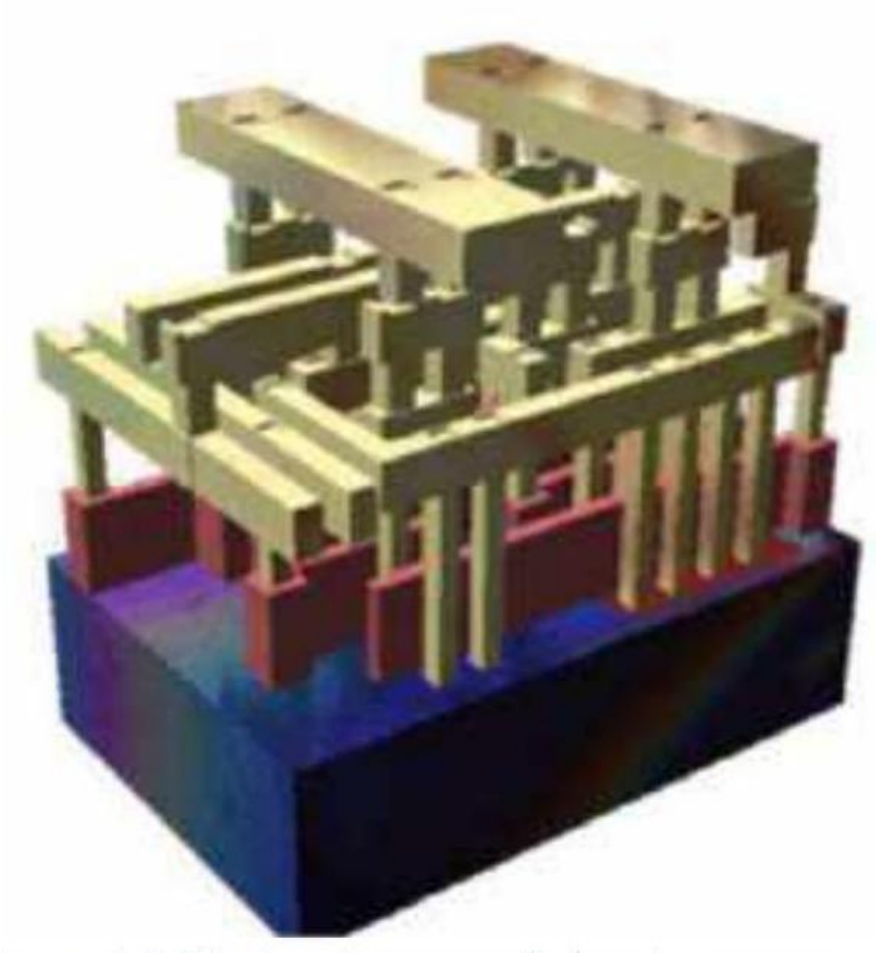
4 Bouwsteentjes op de chip

De ontwikkeling van de miniatuurversie van de transistor (elektronen-schakelaar, uitgevonden rond 1950) zoals die in je telefoon of computer zit laat zien dat het steeds sneller en kleiner kan. In een moderne microprocessors zitten miljoenen transistoren. MOS-techniek zorgt dat die op grote schaal en goedkoop te maken zijn. Met deze techniek worden ook diodes gemaakt, die zorgen dat elektronen maar in één richting kunnen lopen. Een bekende diode is de LED (Light Emitting Diode). Dat de elektronen hier maar in één richting doorheen kunnen kun je uitproberen. De LED geeft alleen licht als hij in een stroomkring in de juiste richting in aangesloten: met het lange

pootje op de pluspool. Ook de condensator, die elektronen kan opslaan en afgeven, is een belangrijke bouwsteen op de chip.

4.1 4.2 Programmeren van een chip

Chip-ontwerpers streven er naar om véél componenten bij elkaar te



Figuur 4.5 3D-structuur van de laagjes op een chip. De geleidende pilaartjes van metaal met alle componenten van de chip worden laag voor laag opgebouwd, tot een dikte van enkele micrometers. De blauwe onderkant is de silicium wafer. pakken op één chip. Via contactpuntjes aan de buitenkant van de chip staat die in verbinding met de pinnetjes van de behuizing. Zo staat de chip in verbinding met de andere componenten van het apparaat en krijgt de chip stroom. Hoe ingewikkelder de chip, hoe meer contactpinnetjes er zijn. Door de stroom op bepaalde pinnetjes aan en andere uit te zetten, kunnen de transistoren in een bepaalde stand gezet worden. Chips kunnen dat schakelen alleen doen als er spanning op de voedingspin staat.

Er bestaan twee belangrijke typen chips. Logische chips kunnen informatie verwerken en geprogrammeerd worden, zoals de centrale processor (CPU) van een computer. Bepaalde software is vast in het geheugen van de chip opgeslagen (firmware, embedded software) zodat die snel beschikbaar is. Geheugenchips houden informatie vast. Het snelle DRAM geheugen doet dat alleen als er spanning op de chip staat. Het trage NAND geheugen, zoals een flashkaartje, houdt informatie ook vast als er geen spanning op de chip staat.

Een chip verwerkt opdrachten in een vaste regelmaat: de kloksnelheid. Op iedere tik van de klok kan de computer één actie uitvoeren. In een computerchip zit een kristal dat met een bepaalde frequentie trilt (in de orde van MHz of GHz) en daarmee bepaalt de chip de tijdsduur en dus zijn kloksnelheid. Die kan bijvoorbeeld verlaagd worden om stroom te besparen. Hoe hoger de frequentie, hoe meer verwerkingsstappen een chip per seconde aan kan.

Hoe moet je het programmeren van een chip nu voorstellen? Programmeren is vergelijkbaar met omzetten van schakelaars. Dat gebeurt bijvoorbeeld in de cockpit van een vliegtuig. De piloten zetten handmatig schakelaars in een bepaalde stand voor vertrek, tijdens de vlucht en na de landing. In het vliegtuig gaan dan allerlei systemen aan of uit. Hoe de schakelaars moeten staan, hebben ze in een checklist staan. Je kunt dat vergelijken met het programma, de software.

Ook het programmeren van een chip is weinig anders dan het omzetten van schakelaars. De software programmeert de chip en zet ook de schakelaars om. Dat zijn in een chip de transistoren. Een pinnetje 'aan' schakelen doe je door er spanning op te zetten (bijvoorbeeld 3 V). 'Uit' is dan 0 V . Dat is in digitale waarden: 1 en 0.

Figuur 4.6 Het schakelbord in een cockpit vergeleken met de pinnen van een ATMEGA 32P microcontroller Een programma zorgt er voor dat een chip taken kan uitvoeren. De afzonderlijke stappen in een programma worden algoritmen genoemd. Als het beginpunt en eindpunt gelijk is, hoeven de tussenliggende stappen niet hetzelfde te zijn. Ofwel, in veel gevallen zijn er meerdere algoritmen te bedenken met dezelfde uitkomst. Een slimme volgorde van handelingen kan een sneller resultaat opleveren. Dat betekent dat het uit maakt hoe je het algoritme schrijft en welke algoritmen je voor het programma gebruikt. Dat is de kunst van het elegant programmeren. Terug naar de cockpit: het is wel handig om de knoppen niet kriskras door elkaar te bedienen, dat kost nodeloos extra tijd. Daarom is bij een goede checklist de volgorde zo efficiënt mogelijk.

4.1.1 Abstract

Op het niveau van de chip moet het algoritme vertellen wat de chip moet doen met de spanning op ieder van de pinnetjes. Maar programmeren door per pinnetje op te schrijven wanneer die 1 of 0 moet zijn, is voor een mens niet te doen. De reeksen 1 en 0 die de chip kan verwerken is machinetaal (machine code). Dat is voor mensen niet te volgen. Vandaar dat er programmeertalen zijn verzonnen die makkelijker te begrijpen zijn en eigenlijk een hele serie handelingen of algoritmen met één commando kunnen geven (assembler code). De computer rekent dit zelf om naar de algoritmen in machinetaal. Dat gebeurt door de compiler. De hogere orde programmeertalen zijn voor mensen beter begrijpelijk, maar staan verder af van de machinetaal.

We kunnen niet van buitenaf zien hoe de schakelaartjes in de computerchip staan. Daardoor kunnen we ons ook moeilijk voorstellen wat een programma met een chip doet en zijn we afhankelijk van informatie die we uit de chip terugkrijgen en vertalen naar iets begrijpelijks.

Er zijn tientallen programmeertalen ontwikkeld, die voor andere toepassingen bedoeld zijn. De Arduino-taal, Python of C++ zijn voorbeelden. Bepaalde talen zijn geschikt om apparaten aan te sturen, andere om plaatjes te bewerken of lange rijen getallen te bewerken.

Als een computerprogrammeur een programma maakt dat werkt, maar dat een ander niet kan begrijpen, is dat niet handig. Slordigheid kost heel veel rekentijd, fouten opsporen en verlies van kennis. Daarom is het plezierig om gebruik te kunnen maken van stukken code die goed werken, maar die je niet zelf hoeft te verzinnen. Er zijn allerlei databases met computercode beschikbaar. Programmeurs halen daar blokken code uit waarmee ze snel een werkend programma kunnen maken. Je hoeft van de code uit een blok niet precies te weten hoe die in elkaar

zit. Als die code ook nog eens als een (gekleurd) blokje wordt weergegeven, zie je echt 'blokken-taal' zoals je die bij veel populaire systemen tegenkomt.

4.1.2 Abstract

5 Programmeren van de microcontroller

Als je met een microcontroller gaat werken, zoals Micro:bit of Arduino, dan programmeer je in Arduino-taal. Jouw Arduino-programma wordt door de software in je computer (de compiler) omgezet naar machinetaal voor de Arduino en naar het bordje gestuurd. Daarmee gaat het bordje de taak uitvoeren. Zolang er geen nieuw programma komt blijft het bordje dezelfde taak uitvoeren. Als je het geheugen wist (dat kan met de RESET knop), is het programma verdwenen en doet het bordje niets meer. Opnieuw moet je vanuit je computer het programma in machinetaal kopiëren om het bordje te laten werken.

Over computerchips en programma's is veel meer te ontdekken. Bijvoorbeeld: de architectuur van processorchips met $4, 8, 16$, tot 64 bit of hoger, parallele of seriële processen, de opbouw van programmeertalen, hoe kunstmatige intelligentie via neurale netwerken geprogrammeerd wordt, opbouw van schakelingen, het bewerken van beelden of de manier waarop een sporthorloge zijn verzamelde data verwerkt en doorstuurt. Hiervoor kun je terecht in verdiepende (vervolg)modules.

5.1 4.3 Artificial Intelligence (AI)

Bij kunstmatige intelligentie (Artificial Intelligence, AI) maakt de software zelf allerlei instellingen die we niet kunnen zien. Dat gebeurt in programma's die kunstmatige neurale netwerken (Artificial Neural Networks, ANN) genoemd worden. Hierdoor kan de software leren door voortdurend patronen te vergelijken. De software kan zo bijvoorbeeld beelden herkennen, zoals bij de gezichtsherkenning van een fotocamera of het zoeken naar bepaalde vormen in afbeeldingen. Doordat de computer zelf de algoritmen aanpast, is het ook niet meer te achterhalen hoe die precies in elkaar zitten. Daarom verzinnen wetenschappers methoden om de algoritmen na te bouwen en te vergelijken met wat AI heeft gedaan. Hoe AI werkt, kun je in een vervolgmodule ontdekken (zie H8).

6 Kunstmatige Intelligentie: heel geschikt voor het leren herkennen van

beelden.

Zoek Wally! Een bekende tekenaar van plaatjesboeken voor kinderen tekende in iedere afbeelding van een boek het figuurtje Wally. Voor de lezers was het de uitdaging om die te ontdekken. We zijn als mensen visueel ingesteld, en heel behendig in het analyseren van beelden, maar het herkennen van wat we zien Vraagt oefening. Zeker als het ingewikkelde en niet-alledaagse beelden zijn. In een foto herkennen we snel dat er menselijke gezichten op staan. Maar het is een stuk lastiger om elke afgebeelde persoon te herkennen, daarvoor kennen we te weinig mensen. Computers die gezichten kunnen herkennen hebben de mogelijkheid om in grote databases te zoeken en razendsnel

vergelijkingen te maken tussen het afgebeelde gezicht en de afbeeldingen in de database. Door de verbeterde

Figuur 4.7 Deze lijkt op Wally gezichtsherkenning steeds meer toegepast. Beveiligingscamera's, toegangspoortjes, toegang tot je telefoon: AI met gezichtsherkenning is wijdverspreid.

Wat met gezichten kan, is ook mogelijk met alle andere patronen. Herkennen van planten, vogels, verkeersborden, de weg, obstakels, kwaadaardige cellen in een orgaan, pakketjes in een sorteercentrum of de boodschappen in een mandje: voor de computer is het allemaal even ingewikkeld. Voor ons is dat heel nuttig: in de zelfrijdende auto, een app die je vertelt welke vogel je ziet, robots die automatisch je boodschappen of pakketjes bezorgen. In medische centra wordt veel onderzoek gedaan naar het herkennen van afwijkingen in medische beelden. Is dit plekje een onschuldig bultje of een kwaadaardig gezwel? Pathologen moeten die Vraag elke dag tientallen keren beantwoorden als ze stukjes weefsel moeten onderzoeken tijdens operaties. Kan de computer artsen helpen om antwoord te geven op deze Vraag? Zal hen dat helpen om betere diagnoses te stellen?

7 Kunstmatige Intelligentie om tumoren te herkennen: het onderzoek van Daan Geijs

Een jaar of tien geleden zat Daan op de middelbare school. Hij had een natuurprofiel, maar volgde niet de juiste vakken om de studie te doen die hij graag wilde: biomedische technologie. Daarom is hij wiskunde B en natuurkunde gaan bijspijkeren om toch in Twente te gaan studeren. Inmiddels afgestudeerd als biomedisch ingenieur én met een lerarenopleiding natuurkunde, is hij aan de slag gegaan als promovendus bij de medische faculteit van de Radboud Universiteit. Hij werkt aan een promotieonderzoek om kunstmatige intelligentie (patroonherkenning door computers) in te zetten bij het opsporen van tumoren. Preciezer gezegd: om een systeem te ontwikkelen dat een patholoog ondersteunt bij het bekijken van weefselplakjes om te ontdekken of er kankercellen aanwezig zijn. Bedenk: van ieder verdacht plekje bij elke operatie moet een patholoog een stukje weefsel in plakjes snijden en beoordelen op de aanwezigheid van kankercellen. Dat is veel werk én het Vraagt de deskundigheid van een hoog opgeleide specialist. Kan je een computer leren om in foto's van die plakjes weefsel de kankercellen er uit te pikken? Kan een computer ingezet worden om de patholoog het vermoeiende routinewerk uit handen te nemen?

Figuur 4.8 Daan Geijs vertelt over de toepassing van digitale technologie in het ziekenhuis Daan heeft een aantal filmpjes opgenomen waarin hij vertelt over zijn opleiding, wat hij nu onderzoekt, wat kunstmatige intelligentie is en waar hij na zijn promotie zou kunnen gaan werken. Deze zijn terug te vinden op de leerlingsite (https://maken.wikiwijs.nl/171772/Schakelmodule Digitale Technologie online materiaal)

7.1 4.4 Maken en programmeren: digitale thermometer met Arduino

In deze opdracht ga je zelf ervaring opdoen met bouwen en programmeren van een Arduino (of vergelijkbaar device): een digitale (koorts)thermometer. Deze meet de temperatuur en toont de temperaturwaarde op een display. Misschien denk je: "wat heeft dat voor zin? Ik kan voor weinig geld ook zo'n ding kopen". Klopt, maar een thermometer is maar een voorbeeld van wat je allemaal met een Arduino kunt maken en doen. Misschien krijg je zelf een heel goed idee en wil je dat gaan maken! Het is dan handig om te weten hoe je dat aanpakt.

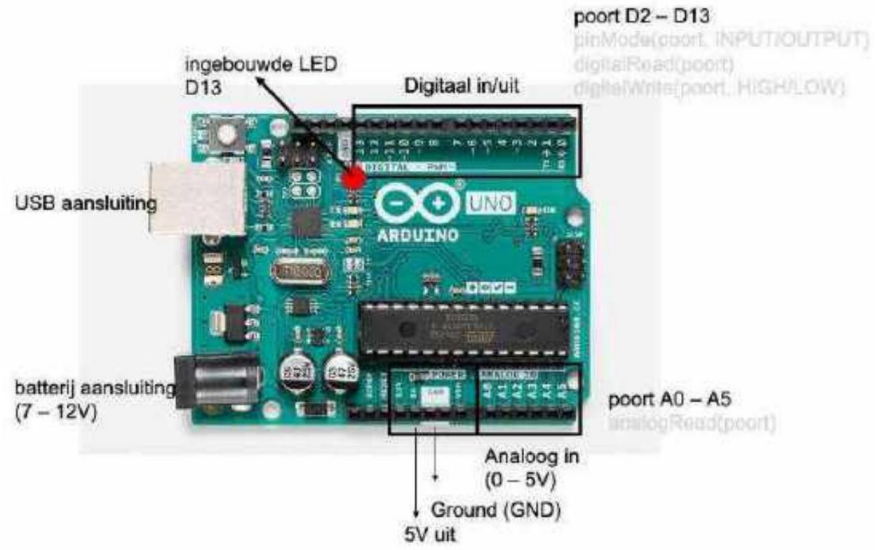
Daarom hieronder eerst een paar stappen die je in ieder project zet. De volledige stap-voor-stap opdracht is in een bijlage beschreven die verkrijgbaar is in de moduledatabase.

8 Stap 1: Kiezen van de hardware

Voor Arduino zijn veel verschillende onderdelen beschikbaar. Als je ze wilt laten doen wat je wilt, moet je de juiste onderdelen kiezen voor je project. Voor de digitale koortsthermometer wil je temperatuur meten, verwerken en laten zien. Je hebt in elk geval een temperatuursensor, schakelaar, Arduinobordje met voeding, schermpje (display) en LED nodig.

Wanneer je met andere spullen aan de slag wilt (zoals de Raspberry Pi, Micro:Bit, M-block of nog andere) moet je ook hier de juiste selectie maken voor je project. Vaak zijn sensoren en andere onderdelen door elkaar te gebruiken, mits de aansluitingen passen. Losse steekpennetjes zijn vaak vervangen door connectoren om onderdelen foutloos aan te sluiten.

9 De Arduino Uno



Figuur 4.9 Het bordje van de Arduino Uno

In figuur 4.9 zie je een afbeelding van een Arduino Uno. Er zijn meerdere soorten Arduino's, maar dit is de belangrijkste. Misschien heb je een kloon die er iets anders uitziet, maar dat maakt niet uit, de aansluitingen zijn hetzelfde. Je moet de Arduino zien als een klein computertje. Op het printplaatje zit een processor (net zoals in je telefoon) voor alle taken. Aan die Arduino sluit je sensoren, display, lampjes etc. aan. Dat doe je op de zwarte insteekpennetjes.

De aansluitingen:

1. USB aansluiting: Hiermee koppel je de Arduino aan de computer
2. Batterij aansluiting: De Arduino kan niet werken zonder spanning. Als de USB is aangesloten, dan krijgt de Arduino daarvan de spanning. Wil je zonder USB werken, dan kun je een batterij/accu/voedingskast aansluiten tussen 7 - 12V Vaak hebben de sensoren ook een spanning nodig. De 5V aansluiting levert die spanning (5V dus). De GND (= Ground) is de min-kant. Het maakt niet uit of de batterij of de voeding uit USB gebruikt wordt.
3. (rechtsboven) digitale in- en uitgangen: Dit zijn in- en uitgangen die een 0 of een 1 kunnen "lezen" (read) of "schrijven" (write). Dat kun je zelf bepalen. Je kunt hiermee een stand van een schakelaar

(aan/uit) lezen (read) of een lampje (LED) aan/uit zetten (write). Hoe dat werkt wordt hieronder verder uitgewerkt. In totaal zijn er 11 van dit soort aansluitingen, genummerd D2 t/m D13 (Digitale poorten).

4. (rechtsonder) analoge ingangen, genummerd $\mathrm{A0t/m}$ A5 (poorten): Dit zijn 6 bijzondere poorten die een spanning (tussen 0 en 5V) kunnen omzetten in een getal. Deze heten ADomzetters: Analoog (spanning) naar Digitaal (getal) omzetter. Deze moeten we gaan gebruiken om de sensor te gaan lezen en kun je dus ook alleen gebruiken om te "lezen".

5. De Uno heeft een ingebouwde LED, gekoppeld aan D13.

10 Stap 2: Opzoeken van de juiste drivers

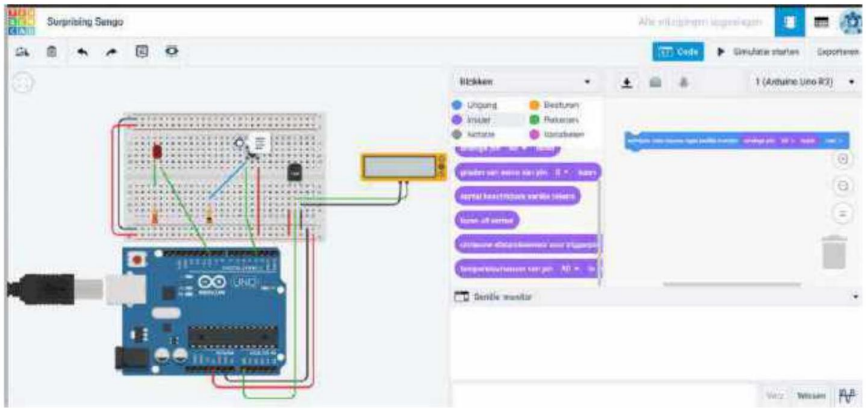
Als je een onderdeel, zoals de temperatuursensor, op een ingang van de Arduino aansluit moet je de Arduino informatie geven. Op welke pinnen is deze aangesloten? Komt er een digitaal of een analoog signaal uit? Wat betekent een waarde die het onderdeel doorgeeft of nodig heeft? Al deze informatie zit verpakt in stukjes code, de drivers. Op internet staan bibliotheken (libraries) met drivers voor al die sensoren en andere bouwstenen. Om die op te zoeken, heb je de naam van een onderdeel nodig, zoals de TMP36 (temperatuursensor). Wanneer je zelf een sensor of actuator bouwt, moet je zelf dit soort informatie bij elkaar zoeken en in een infobestandje zetten voor de Arduino.

11 Stap 3: Aan de slag, eventueel zónder te bouwen

Om de Arduino te vertellen wat die allemaal moet doen met die in- en uitgangen moet je deze programmeren. Je moet dus eigenlijk een app maken voor de Arduino. Dat doe je op de computer. Als het programma klaar is, stuur je het via de USB naar de Arduino en die gaat vervolgens doen wat je geprogrammeerd hebt. Het programmeren doe je in de Arduino IDE.

Het bouwen en programmeren van een apparaat kan een stuk eenvoudiger als je dit vooraf kunt samenstellen en testen met je computer. Voor Arduino kan dat heel mooi in een online omgeving: TinkerCad (ga naar tinkercad.com en maak een account aan).

Je kunt hier met plaatjes je apparaat samenstellen en de code (sketch) schrijven (in blokkentaal, of in de programmeertaal C). Je kunt het programma opslaan, downloaden en in de echte Arduino laden (*.ino bestand). Je kunt ook de Arduino-IDE (Integrated Development Environment) gebruiken om je sketch te maken of een bestaande sketch aan te passen. Als je het fysieke apparaat bouwt en de sketch laadt (via de Arduino IDE), doet deze wat je al uitgeprobeerd hebt. Tenminste: dat is wat je kunt verwachten. Uiteraard moet je testen of dat echt zo is. Daarna kun je de echte metingen gaan doen.



Figuur 4.10 In TinkerCad kun je virtueel een device bouwen (links de hardware en aansluitingen) en het programma in blokken weergeven (midden). De coderegels kunnen ook via een venster getoond worden. Met een knop bovenaan het scherm kan de simulatie gestart worden.

12 Stap 4: Kalibreren, testen en verbeteren

Als je een apparaat hebt gemaakt dat metingen doet, is kalibreren een belangrijk onderdeel. Een waarde uit de sensor (tussen 0 en 1024) moet omgerekend worden naar een correcte waarde op het schermpje. In TinkerCad kun je ook het kalibreren van je sensor uitvoeren, zodat je de juiste code kunt schrijven. Uiteraard controleer je die metingen met de fysieke Arduino nog wel.

Wanneer je metingen opslaat of verstuurt, is het ook nodig om te weten wat er gemeten is en op welk tijdstip. Daarvoor moet je ook een timer op de Arduino aansluiten. De gegevens kunnen op een SD-kaartje gezet worden of via een USB-kabel of draadloos via Bluetooth, WiFi of LoraWan naar een ander apparaat verstuurd. Opslag van data komt in deze opdracht nog niet aan bod, maar wel in die van Q-strip in H7.

Figuur 4.11 Een stukje sketch (code) voor Arduino (in de taal C). Achter de // tekens staat uitleg.