



elektor MAG
design > share > sell

665
SINCE 1961

MEI/JUNI 2021
ELEKTORMAGAZINE.NL

DHZ soldeerstation

nieuw ontwerp voor de Weller RT
en andere soldeerbouten



p. 30



In deze editie

- > Elektor @ 60:
gedachten bij zestig jaar elektronica
- > Java op de Raspberry Pi
- > WiFi voor de LoRa Switch
- > MicroPython voor microcontrollers
- > Seek Shot Pro warmtebeeldcamera
- > Objectgeoriënteerd programmeren -
een korte inleiding
- > Parallax Propeller 2:
de ontwikkelomgeving
- > Retrotronica: Micro-Professor
- > Basiscursus: condensatoren
- > Recht-op-reparatie beweging
laat van zich horen

en veel meer!



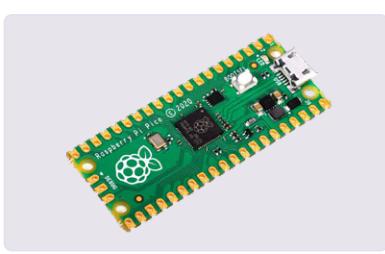
Oplaadbare LiPo-voeding
GreatScott! en Elektor presenteren
de LiPo-supercharger kit

p. 6



Houd de tijd bij met een IoT-knop
met ESP32-gebaseerde M5Stack
en Toggl-webservice

p. 48



Raspberry Pi Pico-board en RP2040
brug tussen SBC's en
microcontrollers

p. 57





elektor MAG

sixty > years > young

Dit jaar is Elektor zestig jaar jong.
De komende twaalf maanden eren
we ons erfgoed maar vooral ook
de community die het allemaal

mogelijk heeft gemaakt! We zijn
trots dat we al zo lang een bijdrage
mogen leveren aan de elektronica-
revolutie! Sinds **1961!**

61^e jaargang nr. 665,
mei/juni 2021
ISSN 2590-0765

Elektor is een uitgave van
Elektor International Media B.V.
Postbus 11, 6114 ZG Susteren, Nederland
Tel.: +31 (0)46- 4389444

Nieuwe abonnementen & bestellingen
service@elektor.nl - Tel. 046-4389444

Elektor International Media B.V. legt gegevens vast voor de uitvoering van de (abonnementen) overeenkomst. De door u verstrekte gegevens kunnen gebruikt worden om u te informeren over relevante diensten en producten. Stelt u daar geen prijs op, dan kunt u dit schriftelijk doorgeven aan:

Elektor International Media B.V.,
Afdeling Customer Service
Postbus 11, 6114 ZG Susteren.

Of per email: service@elektor.nl

In overeenstemming met de Wet bescherming persoonsgegevens zijn de verwerkingen van persoonsgegevens aangemeld bij de toezichthouder, Autoriteit Persoonsgegevens te Den Haag.

Druk: Pijper Media, Groningen
Distributie: Betapress, Gilze

Advertenties Benelux

Raoul Morreau
Tel. +31 (0)6 4403 9907
E-mail: raoul.morreau@elektor.com

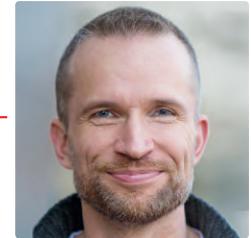
Advertentietarieven, nationaal en internationaal, op aanvraag. Alle advertentiecontracten worden afgesloten conform de Regelen voor het Advertentiewezen gedeponeerd bij de rechtbanken in Nederland. Een exemplaar van de Regelen voor het Advertentiewezen is op aanvraag kostenloos verkrijgbaar.

Auteursrecht

Niets uit deze uitgave mag verveelvoudigd en/of openbaar gemaakt worden door middel van druk, fotokopie, microfilm of op welke wijze dan ook, zonder voorafgaande schriftelijke toestemming van de uitgever. De auteursrechtelijke bescherming van Elektor strekt zich mede uit tot de illustraties met inbegrip van de printed circuits, evenals de ontwerpen daarvoor. In verband met artikel 30 van de Rijksoctrooiwet mogen die in Elektor opgenomen schakelingen slechts voor particuliere of wetenschappelijke doeleinden vervaardigd worden en niet in of voor een bedrijf. Het toepassen van de schakelingen geschieft buiten de verantwoordelijkheid van de uitgever. De uitgever is niet verplicht ongevraagd ingezonden bijdragen, die hij niet voor publicatie aanvaardt, terug te zenden. Indien de uitgever een ingezonden bijdrage voor publicatie aanvaardt, is hij gerechtigd deze op zijn kosten te (doen) bewerken. De uitgever is tevens gerechtigd een bijdrage te (doen) vertalen en voor haar andere uitgaven en activiteiten te gebruiken tegen de daarvoor bij de uitgever gebruikelijke vergoeding.

Jens Nickel

Hoofdredacteur Elektor Magazine



Blijf jong met ons!

Het is zover: Elektor bestaat 60 jaar. In april 1961 publiceerde Bob van der Horst het eerste nummer van zijn nieuwe tijdschrift *Electronica Wereld*, dat later werd omgedoopt tot *Elektuur* en vervolgens tot *Elektor*. De uitgever bespeurde een gat in de markt. De meeste elektroniekken van die tijd ging verscholen in academische literatuur, talloze datasheets, moeilijk te verteren artikelen en saaie diagrammen. Van der Horst voorzag dat veel mensen zoals hij met veel enthousiasme elektronische schakelingen zouden bouwen en deze creatief voor hun eigen doeleinden zouden aanpassen of zelfs helemaal opnieuw zouden ontwikkelen. Hierdoor heeft men de nodige kennis nodig, en de beste manier om die kennis over te brengen, is door de lezers stap voor stap het gevoel te geven dat ze iets gepresteerd hebben. Al doende leren, zogezegd.

Zestig jaar later zijn we dit principe nog steeds trouw. Multi-core programmeren, ontwikkelen met MicroPython en uw eigen hardware verbinden met webplatforms – dat zijn slechts drie van de onderwerpen waarmee we onze lezers stap voor stap vertrouwd willen maken, met die bewezen mix van theorie en praktijk. Als hoofdredacteur profiteert ik van het feit dat onze redacteuren en auteurs stuk voor stuk besmet zijn met het DHZ-virus. Daarom krijg ik bijna nooit een ‘droge’ of ‘te theoretische’ bijdrage onder ogen. Meestal moet ik onze creativiteit zelfs een beetje afremmen om te voorkomen dat een project te omvangrijk wordt (u wilt de nieuwe *Elektor* immers op tijd ontvangen). Via alle bijdragen houden we onze vinger aan de pols – dat houdt ons team en ons magazine fris. Ik weet niet wat er over enkele tientallen jaren allemaal in *Elektor* zal staan, maar onze opvolgers zullen ook dan nog steeds projecten bedenken die de opwindende technologieën van die tijd belichten!

ELEKTOR @ 60: ER ZIT NOG MEER AAN TE KOMEN!

Elektor viert 60 jaar uitgeven en elektronica innoveren met spannende speciale projecten. We werken aan een jubileumboek, een film, een live-evenement (World Ethical Electronics Forum) en het “mobiele Elektor-lab”. Wedden dat u nieuwsgierig bent? De komende weken komt er meer informatie. Ideeën voor andere bijzondere projecten? Vertel me waar u aan denkt: denise.bodrone@elektor.com!

Denise Bodrone, Coördinatie Elektor 60-feestcommissie



Ons team



Internationaal hoofdredacteur: [Jens Nickel](mailto:jens.nickel@elektor.nl) (redactie@elektor.nl)

Redactie: [Eric Bogers](mailto:eric.bogers@elektor.nl) (redacteur_NL-editie@elektor.nl), [Jan Buiting](mailto:jan.buiting@elektor.nl),

[Rolf Gerstendorf](mailto:rolf.gerstendorf@elektor.nl), [Dr. Thomas Scherer](mailto:dr.thomas.scherer@elektor.nl), [Clemens Valens](mailto:clemens.valens@elektor.nl)

[Mathias Claussen](mailto:mathias.claussen@elektor.nl), [Ton Giesberts](mailto:ton.giesberts@elektor.nl), [Luc Lemmens](mailto:lu Lemmens@elektor.nl),

[Clemens Valens](mailto:clemens.valens@elektor.nl)

[Denise Bodrone](mailto:denise.bodrone@elektor.nl)

[Giel Dols](mailto:giel.dols@elektor.nl), [Harmen Heida](mailto:harmen.heida@elektor.nl)

Directeur: [Don Akkermans](mailto:don.akkermans@elektor.nl)



Elektor is lid van FIPP, een organisatie die “in bijna 100 jaar is uitgegroeid tot een wereldwijde vereniging voor media-professionals”.



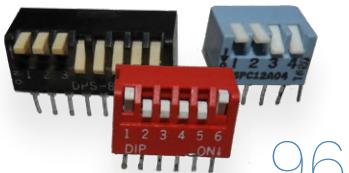
Elektor is lid van de VDZ (Verband Deutscher Zeitschriftenverleger), die “de gemeenschappelijke belangen behartigt van 500 Duitse consumenten- en B2B-uitgevers”.



Rubrieken

- 45 Developer's Zone**
geen angst vor de cellulaire module!
- 88 Oost West Lab Best**
waar Kurt Diedrich synthesizers bouwt
- 96 Vreemde onderdelen**
DIP-schakelaars
- 98 Interactief**
correcties & updates || brieven van lezers
- 100 Elektor Ethisiek**
wiens product is het eigenlijk?
- 102 Uit het leven gegrepen**
het grote blunderboek
- 104 Retrotronica**
Micro-Professor
- 114 Hexadoku**

Vreemde onderdelen



96

Achtergrond & info

- 14 Elektor @ 60**
gedachten bij zestig jaar elektronica
- 23 Review: Siglent tafelmultimeter SDM3045X**
- 37 Kennismaking met de Parallax Propeller 2**
deel 2: de ontwikkelomgeving en de code
- 57 Raspberry Pi Pico**
kennismaking met het Pico-board en de RP2040
- 64 MicroPython voor microcontrollers**
technieken voor kleine displays
- 76 Warmtezoeker**
de Seek Shot Pro warmtebeeldcamera
- 80 Objectgeoriënteerd programmeren**
een korte inleiding met C++
- 90 Java op de Raspberry Pi**
deel 1: GPIO's
- 110 Alle begin...**
basiscursus voor beginners

Projecten

- 6 Zelfbouw LiPo-supercharger kit**
van prototype tot productie
- 26 DC-stroomklem**
Hall-sensor + ferrietkern + Arduino

Wi-Fi voor de LoRa switch



30 Soldeerstation 2021

makkelijk om te bouwen!

40 WiFi voor de LoRa Switch

geïntegreerd in Home Assistant met ESPHome

48 Houd de tijd bij met ESP32 en Toggl

werken met de M5Stack

70 12-200 V DC/DC-converter

voor buizenversterkers

Binnenkort

Elektor juli/augustus 2021

Het volgende nummer is zoals altijd tot aan de nok gevuld met zelfbouwprojecten, achtergrondinformatie en tips en trucs voor elektronici.

- Magnetische levitatie
- LoRa-module met Raspberry Pi Pico
- Elektronische belasting voor AC en DC
- Positieve en negatieve micro-USB breadboard-voeding
- DHZ-camerasysteem voor Raspberry Pi
- Elektronisch kompas
- MicroPython op de ESP32 – stap voor stap
- Displays in Raspberry Pi-projecten

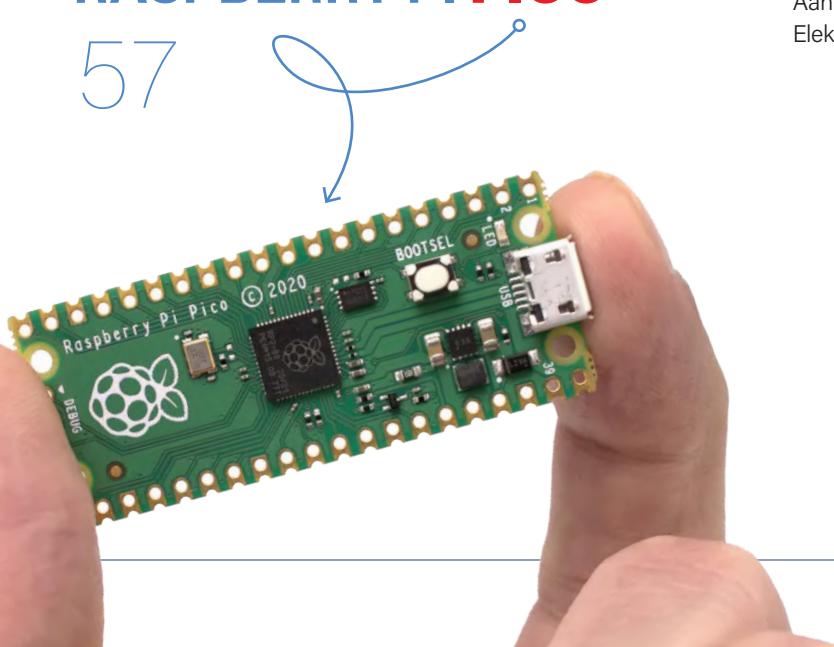
...en nog veel meer!

Aankondigingen onder voorbehoud.

Elektor juli/augustus 2021 verschijnt omstreeks 1 juli 2021.

Kennismaking met de
RASPBERRY PI PICO

57



elektor MAG
design > share > sell

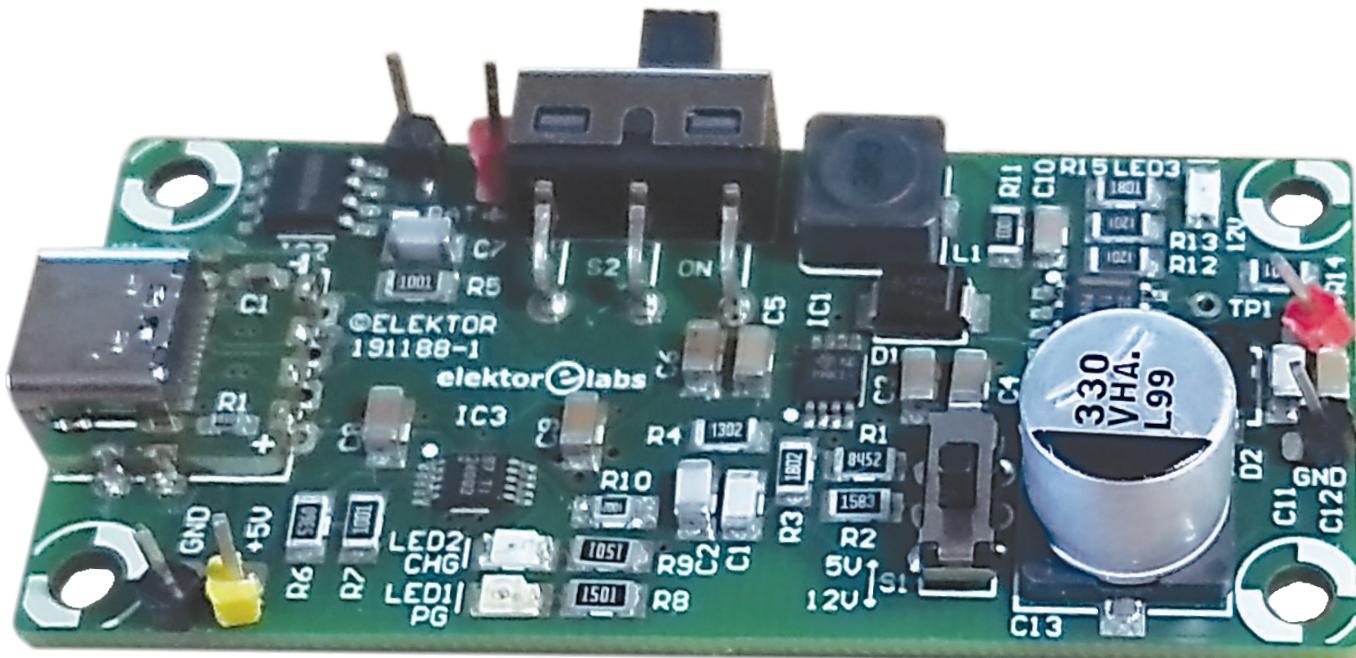


Zelfbouw LiPo-supercharger kit

van prototype tot productie

Mathias Claußen (Elektor)

GreatScott! – een op doe-het-zelf-elektronica gericht YouTube-kanaal met meer dan 1,4 miljoen abonnees – en Elektor presenteren een oplaadbare LiPo-voeding voor zelfbouw. De handige kit biedt gebruikers een instap in de wereld van SMD-solderen, met zorgvuldig gekozen 1206-componenten en een gedeeltelijk voorgemonteerde print. Het ontwikkelen van de kit stelde ons voor enkele interessante uitdagingen, en daarom willen we de lessen die we hebben geleerd tijdens het ontwikkelen van het product met u delen.



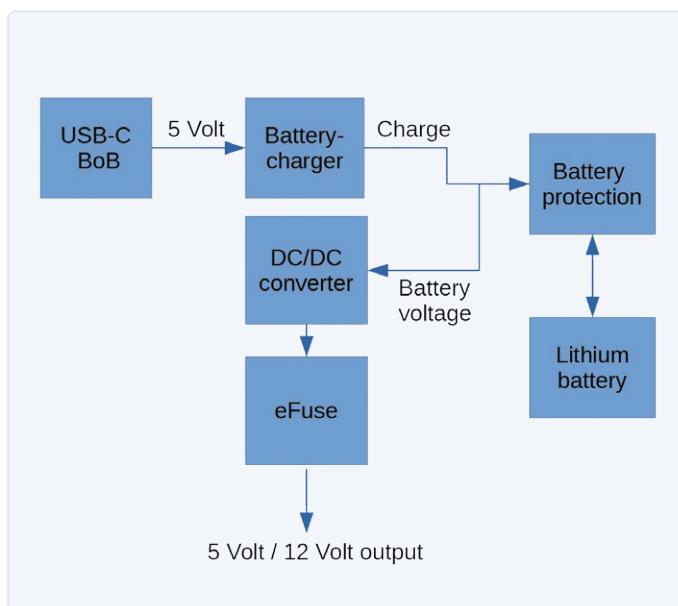
SPECIFICATIES

- › Ingang: 5 V ±10%
- › Uitgang: 5 V/1,5 A of 12 V/0,75 A
- › Enkelcel lithium-accu

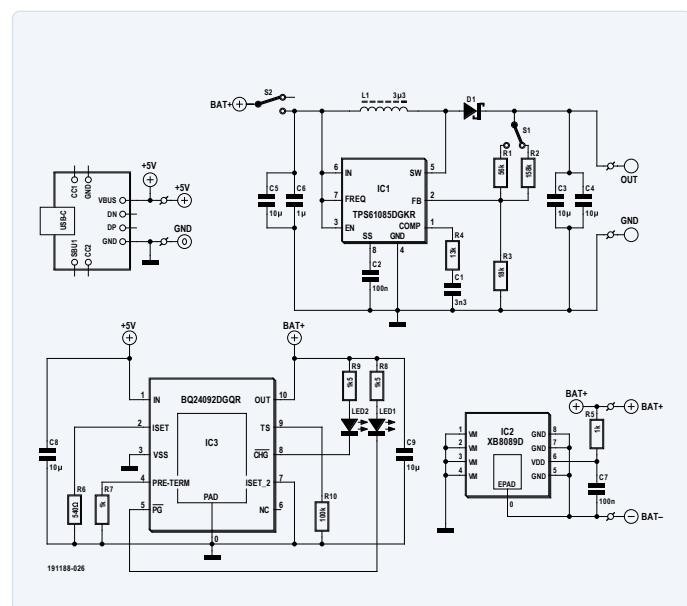
GreatScott! en Elektor ontmoetten elkaar tijdens productronica 2019 om een nieuw project te bespreken: een zelfbouw-voeding met een oplaadbare lithium-accu. De meesten van ons gebruiken de vertrouwde labvoeding voor onze projecten. Maar als u werkt aan

ontwerpen die uiteindelijk de werkbank verlaten, zoals een autonoom rijdende modelauto of een buitensor, dan zijn lange voedingskabels niet de beste oplossing. Dergelijke projecten resulteren meestal in een set batterijen die met tape bij elkaar worden gehouden en met hotmelt op een goedkope DC/DC-omzetter worden gelijmd als geïmprovideerde mobiele voeding. Om een robuuster systeem met extra beveiliging te ontwikkelen, bundelden GreatScott! en Elektor hun krachten om een oplaadbare accu-voeding te ontwikkelen die

met goed verkrijgbare lithium-accu's werkt. Het team besloot meteen al dat het een zelfbouw-project moest worden met SMD-componenten om het solderen daarvan te leren en om te demonstreren dat SMD-solderen geen 'zwarte magie' is die voorbehouden blijft aan goed opgeleide tovenaars. Tijdens de ontmoeting bleek dat GreatScott! al enkele schetsmatige schema's en berekeningen had voorbereid. Het verschil met andere voedingen is de uitgang die door de gebruiker tussen 12 V en 5 V kan worden omgeschakeld.



Figuur 1. Blokschema.



Figuur 2. Schema van de eerste versie.

Blokschema

Om een lithium-acculader annex voeding te bouwen, moeten de functieblokken op de juiste manier worden aangesloten. **Figuur 1** toont de blokken die bij het ontwerp zijn betrokken. Het eerste blok is de USB-connector, zodat een USB-lichtnetadapter kan worden gebruikt om de schakeling te voeden. Hier gebruiken we een print die uitsluitend passieve componenten bevat en waar alle benodigde signalen langs de randen toegankelijk zijn. Deze wordt later bovenop de hoofdprint geplaatst en kan worden hergebruikt voor andere projecten waar USB-C van pas komt. Dit blok wordt gevuld door de laadschakeling voor de Li-accu die de ingangsspanning gebruikt om een aangesloten accu te laden.

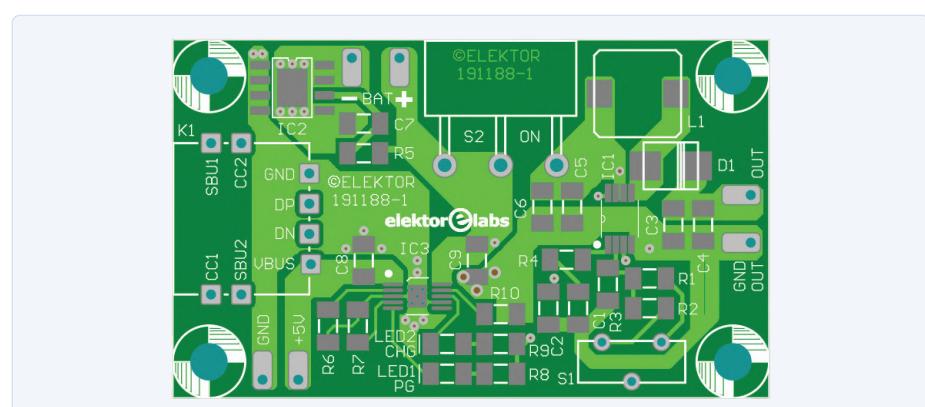
Om schade aan de aangesloten accu te voorkomen, hebben we de nodige beveiliging toegevoegd (in figuur 1 als apart blok getekend) om overstroom, overbelasting, diepe ontlading en verkeerde polariteit te voorkomen. In het midden ziet u de DC/DC-omzetter die aan zijn uitgang 5 V of 12 V levert. Het laatste blok na de DC/DC-omzetter is een eFuse die de uitgangsstroom beperkt om schade aan de DC/DC-omzetter te voorkomen. Wanneer deze blokken op de juiste manier worden aangesloten is het resultaat een draagbare oplaadbare voeding.

Standaardonderdelen

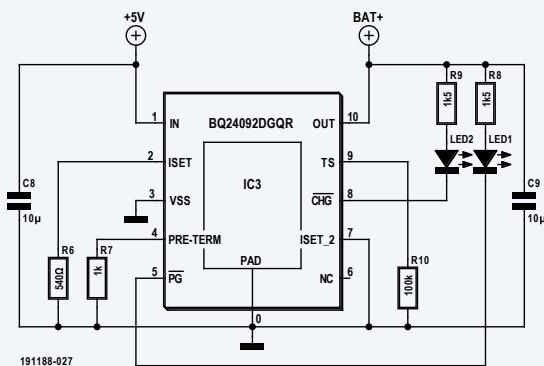
De onderdelenlijst is geoptimaliseerd voor onderdelen uit China – dat leek in december 2019 redelijk eenvoudig, aangezien goede

van en naar het Chinese vasteland in grote hoeveelheden voor een lage prijs konden worden getransporteerd. De meeste onderdelen kunnen ook bij Europese leveranciers worden gekocht. De belangrijkste IC's voor dit project zijn van een bekend merk, Texas Instruments. Slechts één IC was alleen bij Chinese distributeurs verkrijgbaar, maar dat lijkt geen probleem. Voor het laden wordt een BQ24092DGQR van Texas Instruments gebruikt die geschikt is voor oplaadbare lithium-ion- en lithium-polymeeraccu's. Om een uitgangsspanning van 12 V of 5 V te krijgen, wordt een TPS61085DGKR, ook van Texas Instruments, gebruikt die als DC/DC-boost-converter werkt. Andere onderdelen zoals condensatoren, weerstanden en spoelen zijn allemaal verkrijgbaar bij Europese distributeurs. Over het algemeen zijn in het

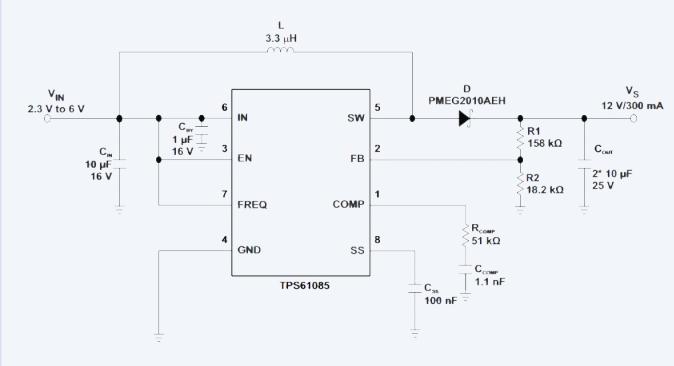
ontwerp standaardcomponenten gebruikt. Een medewerker van het Elektor-lab nam het schema, de onderdelenlijst en de eerste berekeningen onder zijn hoede. Met deze gegevens was het ontwerpen van een print snel gedaan en werd het eerste prototype besteld. Net voor de kerstdagen van 2019 was het gereed om te testen. Het schema is te zien in **figuur 2** terwijl de bijbehorende print in **figuur 3** wordt getoond. De eerste test door GreatScott! was veelbelovend. De print zou geproduceerd kunnen worden – maar zo simpel was het niet. Rond die tijd begon COVID-19 de wereld op zijn kop te zetten en toeleveringsketens te beïnvloeden; standaardcomponenten verdwenen steeds meer uit de schappen. Bovendien bleken er twee grote problemen in het ontwerp te zitten.



Figuur 3. Print-layout voor de eerste versie.



Figuur 4. Schema van het laadgedeelte.



Figuur 5. Schema van de DC/DC-converter.

Een nul te veel en gebraden IC's

Een eerste test wees uit dat voeding vanuit van een oplaadbare accu prima werkte, maar het opladen daarvan niet. Aangezien een BQ24092DGQR is geïnstalleerd, kenden we dit verschijnsel van een ander prototype. **Figuur 4** toont het schema van de laadschakeling. Weerstand R10 is nodig om deze chips te kunnen gebruiken zonder sensorweerstand in een accupack. Afhankelijk van het type BQ2409x dat wordt gebruikt, kan de waarde 10 kΩ of 100 kΩ zijn, en hier is de waarde 10 keer groter dan wat de BQ24092DGQR verwacht. Dat heeft tot gevolg dat de accu niet wordt opgeladen omdat het laad-IC denkt dat de temperatuur buiten het toegestane bereik ligt. De oplossing is simpel: R10 wordt 10 kΩ in plaats van 100 kΩ.

Het tweede probleem was ernstiger. Tijdens het testen ging de DC/DC-omzetter kapot als er een hoge stroom werd getrokken. Aangezien in het schema van figuur 1 slechts een beveiligings-IC is opgenomen dat de stroom van de accu beperkt tot 10 A, betekent dit dat er ongeveer 37 W in de DC/DC-boosttrap kan worden verstoort. Bij 5 V uitgangsspanning komt dat neer op 7,4 A die de belasting in theorie onder ideale omstandigheden zou kunnen trekken. Omdat de omstandigheden nooit ideaal zijn, wil dat zeggen dat de DC/DC-omzetter zo overbelast kan worden dat hij intern gebraden wordt. Er moet een beveiliging worden geïnstalleerd voor de DC/DC-omzetter – maar laten we eerst kijken hoe een DC/DC-omzetter eigenlijk werkt.

De werking van een DC/DC-omzetter

Het doel van een DC/DC-omzetter is om een DC-uitgangsspanning te genereren die lager of hoger is dan de DC-ingangsspanning. Met AC-ingangsspanningen is dit vrij

eenvoudig omdat u een transformator kunt gebruiken waarvan de uitgangsspanning afhankelijk is van de wikkelperhouding. DC-spanningsomzetters daarentegen moeten vóór de conversie eerst een soort wisselspanning genereren uit de DC-ingangsspanning. Verbazingwekkend genoeg bestonden dergelijke converters al vóór het halfgeleidertijdperk, meestal mechanisch en veel groter dan wat er in moderne elektronica te vinden is. Een korte beschrijving is te vinden op Wikipedia [1].

In moderne elektronica hebben DC-spanningsomvormers maar een paar externe componenten nodig, omdat de meeste functies in een klein IC zijn geïntegreerd. Als de uitgangsspanning lager is dan de ingangsspanning, hebben we met een zogenaamde buck-converter te maken. Als een lage ingangsspanning in een hogere uitgangsspanning moet worden omgezet, doen we dat met een boost-converter. De hier gebruikte TPS61085DGKR is een boost-converter die 5 V of 12 V genereert uit een ingangsspanning in een bereik van 3...4,2 V. **Figuur 5** maakt aanschouwelijk hoe dit werkt. Tussen ingang V_{IN} en uitgang V_S bevindt zich een serieschakeling van spoel L en diode D. Tussen de uitgang en massa is de condensator C_{out} aangesloten. De IC-pin SW is tussen de spoel en de diode aangesloten; in het IC zit daar een schakelaar naar massa. Als de schakelaar gesloten is, staat de ingangsspanning U_{IN} over de spoel. De stroom I_L via de SW-schakelaar naar massa neemt toe en creëert een magnetisch veld in de spoel. Diode D voorkomt dat condensator C_{out} zich kan ontladen terwijl de schakelaar gesloten is.

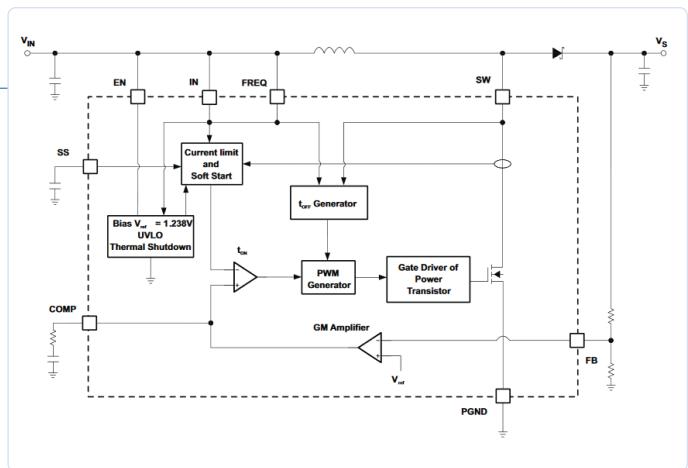
Zodra de schakelaar wordt geopend, probeert de spoel de energie die is opgeslagen in het magnetische veld weer kwijt te raken. De spanning op SW stijgt snel tot meer dan de

ingangsspanning (dus de polariteit van de spoel wordt omgekeerd) totdat deze spanning hoger is dan die over de uitgangscondensator; dan gaat de diode in geleiding. De stroom loopt in de richting van de belasting en laadt condensator C_{out} op tot de som van de spoelspanning en de ingangsspanning – tot de energie van het magnetische veld is uitgeput. Door de schakelaar te sluiten begint het hele proces opnieuw. Als dit proces zich permanent blijft herhalen, hebben we een uitgangsspanning die groter is dan de ingangsspanning.

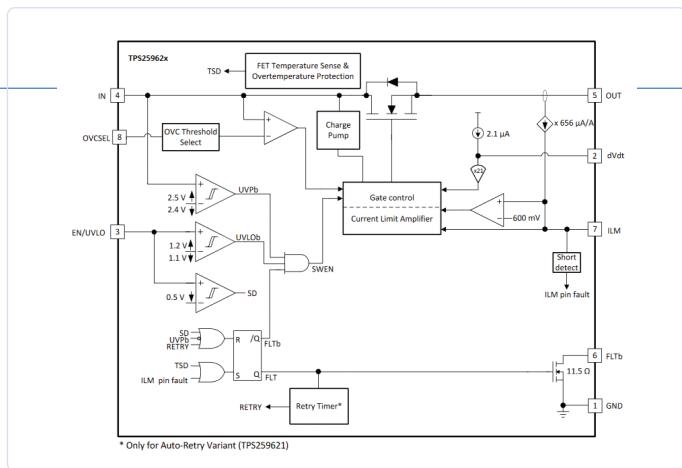
We werpen nu een blik op het inwendige blokschema van het converter-IC (**Figuur 6**) om te zien hoe de timing die nodig is voor de boostwerking en een stabiele uitgangsspanning moet worden geregeld. De geïntegreerde schakelaar – hier in de vorm van een FET – verbindt de ‘uitgang’ van de spoel met massa. Om de juiste timing voor de interne FET te genereren, wordt de uitgangsspanning vergeleken met een referentiespanning V_{ref} die bij dit IC 1,238 V bedraagt. Hierdoor kunnen we een spanningsdeler gebruiken die is aangesloten op de feedbackpin FB en waarvan de deelspanning 1,238 V bedraagt wanneer de uitgang 12 V of 5 V is. Als de spanning op de feedbackpin hoger is dan 1,238 V, zorgt de interne logica ervoor dat de gegenereerde uitgangsspanning wordt verlaagd totdat de feedbackpin weer precies 1,238 V voert. Er zijn intern beveiligingen ingebouwd om de stroom door de interne FET te beperken, een onderspanning-afschakeling te verzorgen en de DC/DC-converter helemaal uit te schakelen als het IC te warm wordt.

Overstroom en beveiliging

Als de uitgang van de converter zware belastingen of zelfs een kortsluiting aanstuurt, leidt dat tot overbelasting van de interne FET en de schakeling daaromheen, met als gevolg



Figuur 6. Intern blokschema van de DC/DC-converter.



Figuur 7. Intern blokschema van de eFuse.

het plotseling overlijden van het IC. Om dit te voorkomen, kunnen er op twee plaatsen beveiligingen worden toegevoegd, namelijk voor de converter en aan de uitgang. Als snelle oplossing met minder componenten zou het voor de hand liggen om de stroom en het vermogen te beperken die in de DC/DC-converter worden gestuurd. Als beschermend element zouden we een polymercomponent met positieve temperatuurcoëfficiënt (een herstelbare polyfuse) kunnen gebruiken die de stroom naar de DC/DC-converter beperkt. Dat lijkt wellicht een oplossing om de converter te beschermen, maar ten gevolge van het werkingsprincipe van de DC/DC-converter worden hierdoor ongewenste bijwerkingen veroorzaakt.

Als eerste poging hebben we zo'n polyfuse op de print gemonteerd. Dit beschermd de DC/DC-converter weliswaar op korte termijn tegen schade, maar veroorzaakte ongewenste oscillaties. Om dat te begrijpen hoeven we niet uitgebreid aan een prototype te meten; goed nadenken volstaat. Een herstelbare polyfuse begrenst de stroom door een verhoging van zijn interne weerstand bij

toenemende temperatuur. Omdat de zekering zelf een gedefinieerde weerstand heeft, zal de stroom erdoorheen de interne temperatuur verhogen. Hoe groter de stroom door de zekering, des te hoger diens temperatuur wordt en de resulterende weerstand van de zekering. Deze weerstand beperkt uiteindelijk de stroom door de zekering.

Als de uitgang van de DC/DC-converter fors belast wordt, zal de ingangsstroom ook een hoge waarde hebben. De polyfuse wordt daardoor warm en krijgt een hogere weerstand, waardoor de stroom erdoorheen wordt beperkt. De grotere weerstand van de polyfuse zal ook leiden tot een hogere spanningsval met als gevolg een lagere ingangsspanning voor de converter. De stroom in de converter is nu begrensd zodat die niet kapot kan gaan, maar moet nu de uitgangsspanning en het geleverde vermogen in stand houden met een nog lagere ingangsspanning, waardoor er voor de goede werking meer stroom nodig is. Deze grotere stroom zal de weerstand in de polyfuse nog verder verhogen met als gevolg een nog grotere spanningsval over de zekering.

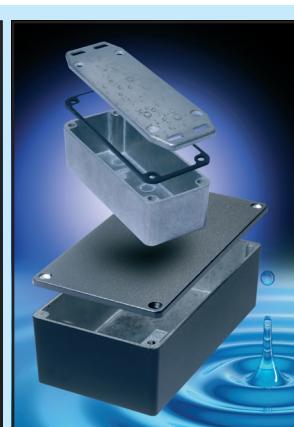
Dit leidt op zijn beurt tot een nog lagere ingangsspanning voor de converter. Op een gegeven moment zakt de ingangsspanning onder de drempel van de onderspanning-afschakeling en wordt de DC/DC-boostconverer uitgeschakeld. In uitgeschakelde toestand is het stroomverbruik lager. Hierdoor kan de polyfuse afkoelen en wordt diens weerstand lager. En dus tot een geringere spanningsval over de polyfuse, resulterend in een hogere ingangsspanning voor de converter. Deze zal weer opstarten en het hele verhaal begint van voren af aan, met oscillatie als gevolg. Dit wordt uiteraard aan de uitgang merkbaar, in de vorm van spikes en andere ‘viezigheid’ op de uitgangsspanning. Een polyfuse aan de ingang is dus geen ideale oplossing, maar een zekering aan de uitgang is dat wel. Bij een enkele uitgangsspanning en een enkele stroomlimiet supersimpel. Met twee uitgangsspanningen en twee verschillende stroomlimieten kan een enkele polyfuse echter geen afdoende bescherming bieden voor beide spanningen en beide stromen. Een veelzijdiger oplossing voor dit probleem, vooral bij veranderende



hammfg.com/small-case

5000+ modellen van kunststof,
spuitgegoten en geëxtrudeerde
behuizingen op voorraad

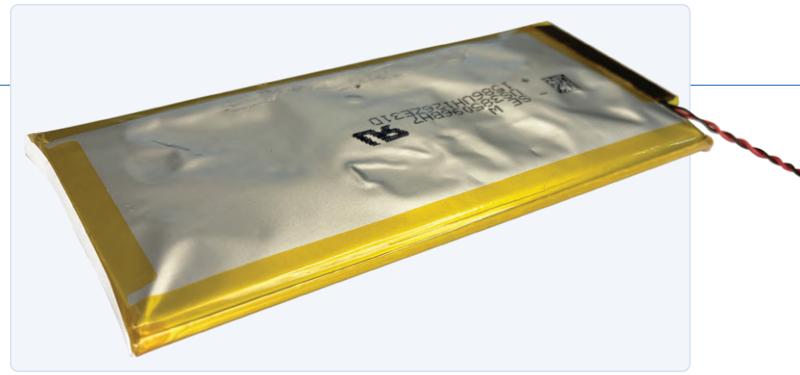
eusales@hammfg.com • + 44 1256 812812



Advertentie



Figuur 8. Beschadigde en opgeblazen platte batterij.



Figuur 9. Beschadigde batterij van een mobiele telefoon.

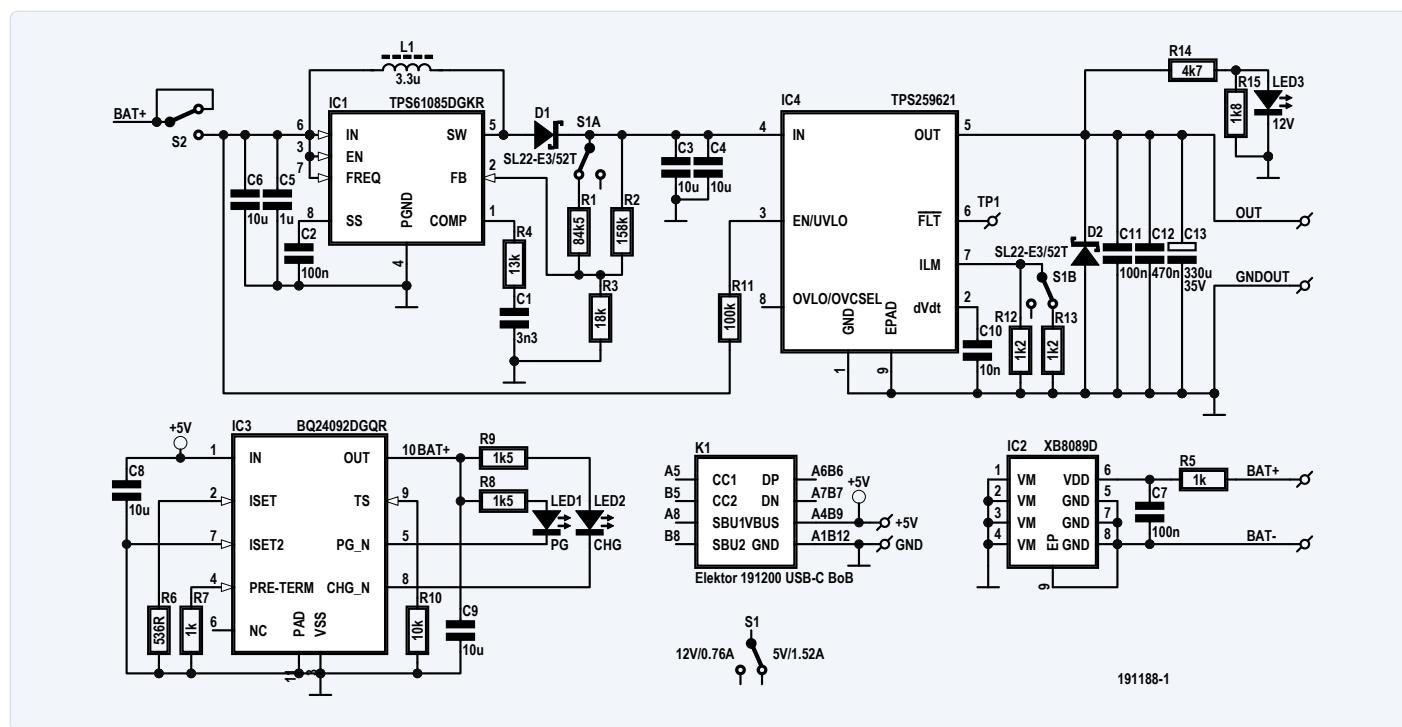
spanningen en stromen, is een eFuse, die al in een online Elektor-artikel [2] is genoemd. Na te hebben rondgesnuffeld voor eFuses voor een ingangsspanning van 5 V tot 12 V die 0,5 A tot 2 A verdragen, hebben we gekozen voor de Texas Instruments TPS259621DDAR. Zoals u in **figuur 7** ziet, bieden deze IC's meer dan alleen een simpele overstroombeveiliging: ze kunnen ook worden gebruikt voor over- en onderspanningsbeveiliging. In figuur 2 hebben gezien dat er een schakelaar is om de uitgangsspanning te kiezen. Omdat de stroomlimiet ook wordt ingesteld via weerstandswaarden en afhankelijk van de ingestelde spanning wordt gewijzigd om de DC/DC-converter te beschermen, kunnen we met één schakelaar die twee onafhankelijke kanalen veranderen en zowel de uitgangsspanning als de stroomlimiet in één

keer instellen. Zo'n dubbelpolige schakelaar vormt een goede oplossing voor de instelbare overstroombeveiliging.

Lithium-accu's: gemakkelijk te laden, maar ook brandgevaarlijk

Tegenwoordig worden in de meeste draagbare elektronische apparaten op lithium gebaseerde accu's gebruikt. Door de vooruitgang in de accutechnologie bieden die cellen een grote vermogensdichtheid per volume en per gewicht. Terwijl voor oudere oplossingen zoals NiCd en NiMh complexere schakelingen nodig zijn voor het laden en het einde van het laadproces te detecteren, zijn oplossingen op lithiumbasis eenvoudiger. Als de cel niet volledig leeg is en een restspanning heeft van ongeveer 2,5 V, wordt voor het opladen een CC/CV-benadering gebruikt – eerst een

constante stroom met een laad-eindspanning en beëindiging van het laden als een gedefinieerde spanning wordt bereikt. Voor lithium-ion en lithium-polymeer is dat typisch 4,15 V. De BQ24092DGQR doet precies dit tijdens het opladen van een aangesloten accu, door een begrenste stroom van maximaal 1 A totdat de celspanning 4,15 V bereikt. Ook is de chip voorzien van een oplaadtimer die het laden na een bepaalde tijd stopt, voor het geval dat de accu de waarde van 4,15 V niet bereikt. Een extra beveiliging, optioneel ondersteund door het oplaad-IC, is temperatuurbewaking. Hoogwaardige laders en accupacks zijn voorzien van temperatuurbewaking, meestal een temperatuurgevoelige weerstand, om oververhitting van de batterijen tijdens het opladen te voorkomen. Omdat in ons geval veel verschillende typen en merken accu's



Figuur 10. Het definitieve schema van de LiPo-supercharger.

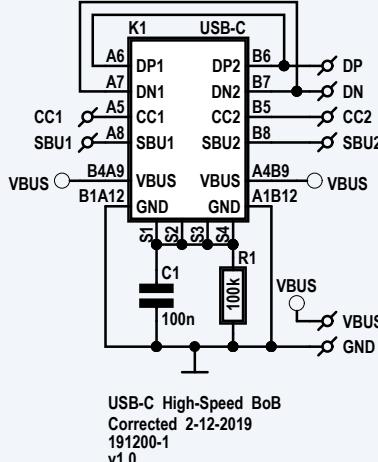
kunnen worden aangesloten, is het niet zinvol om deze optie hier te implementeren. Als om welke reden dan ook deze bewaking niet wordt gebruikt, moet in plaats daarvan een weerstand worden gemonteerd die ervoor zorgt dat de lader zonder temperaturubewaking werkt. U moet wel de juiste weerstand berekenen, anders wordt er niet geladen. Wanneer incompatibele lithium-accu's, zoals lithium-ion-fosfaat typen (LiFePo), worden aangesloten kunnen deze vlam vatten. Zorg ervoor dat u hoogwaardige cellen gebruikt, want een beschadigd exemplaar kan uitzetten of heet worden en mogelijk zelfs vlam vatten. **Figuur 8** toont enkele beschadigde lithium-accu's die hun oorspronkelijk platte vorm hebben verloren en bijna rond zijn geworden. **Figuur 9** toont een beschadigde cel van een mobiele telefoon, die met genoeg kracht is uitgezet om een vastgelijmd display van het chassis los te maken.

Definitieve specificaties

Nu moeten nog de parameters voor de eFuse worden vastgelegd. Waarden van 1,5 A bij 5 V en 0,75 A bij 12 V liggen in het veilige bereik voor de DC/DC-converter. Ook moet de laadstroom voor de aangesloten accu worden ingesteld. De BQ24092DGQR biedt maximaal twee laadstroominstellingen, afhankelijk van de configuratie van de pinnen ISET1 en ISET2. ISET2 ligt aan massa en dat betekent dat de chip de stroominstelling van ISET gebruikt. Met een weerstand van 536 Ω resulteert dit in een laadstroombegrenzing van 1 A voor de aangesloten accu. Zorg ervoor dat u een accu aansluit die deze laadstroom probleemloos kan verwerken!

Een beetje feedback en afronding van de print

Nadat de eFuse aan het ontwerp was toegevoegd, moest één ding worden besproken. Het originele schema had geen visuele indicatie of de uitgangsspanning was ingesteld op 5 V of 12 V. GreatScott! en Elektor concludeerden dat een extra LED als indicator de bruikbaarheid van het product aanzienlijk zou vergroten. We waren ook van mening dat de visuele feedback kan voorkomen dat er per ongeluk 12 V wordt aangesloten op een systeem dat slechts 5 V verdraagt. De schakeling om '12 V' te signaleren werkt met een LED en twee weerstanden. Hierdoor licht de LED alleen op als er 12 V op de uitgang staat. De tweede PCB-revisie werkte zoals verwacht. Om het ontwerp af te ronden moesten de laatste wijzigingen aan de tekst en het logo worden toegevoegd, referenties op de



Figuur 11. Schema van de USB-C BoB.



Figuur 12. De USB-C BoB.

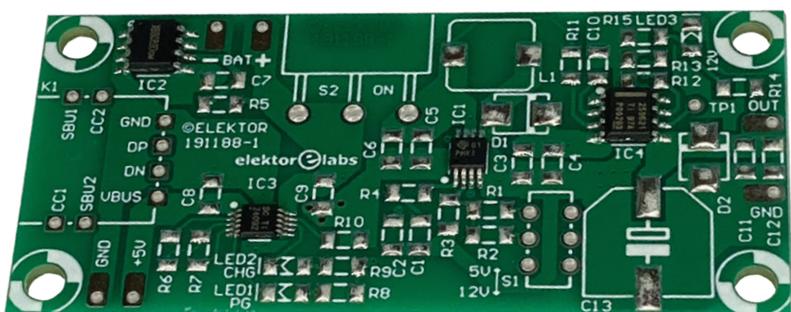
juiste plaats gezet en al het materiaal naar de productie gestuurd. **Figuur 10** toont het uiteindelijke schema van de zelfbouw LiPo-supercharger. **Figuur 11** toont het schema van de USB-C BoB.

Van prototype tot productie

Het in het lab ontwikkelen van producten en het bouwen van prototypes is iets anders dan de productie daarvan met geautomatiseerde machines en serviceproviders. We hebben het nog niet gehad over de kleine USB-C BoB (**figuur 12**) die op de hoofdprint wordt gemonteerd en een eigen projectnummer heeft. Omdat een USB-C-aansluiting voor voeding en data niet alleen voor dit product van pas komt, is daarvoor een kleine universeel inzetbare print ontworpen en geproduceerd. Alle belangrijke aansluitingen voor voeding en data-overdracht met USB2.0-snelheid zijn als halfronde soldeerilandjes aan drie zijkanten toegankelijk.

Dat betekende weliswaar dat ze handig in het gebruik waren, maar niet zo gemakkelijk te produceren. De losse PCB's waren zo klein dat ze moeilijk machinaal verwerkt konden worden. Om dit probleem op te lossen,

moest het productiebedrijf enkele technische maatregelen nemen om ze met hun machines te kunnen hanteren. Niettemin: ze zijn geproduceerd en zitten in de kit. Maar als u een kleine print ontwerpt met halfronde soldeerilandjes aan drie zijden, moet u eerst contact opnemen met de fabrikant en vragen wat de extra fabricagekosten zijn. Bovendien: voor een SMD-kit moeten de componenten in de kit afzonderlijk worden verpakt of georganiseerd. Omdat dit met de hand moet gebeuren, kunnen de kosten oplopen. Uiteindelijk kan het goedkoper zijn om een volledig geassembleerde print aan te bieden in plaats van een kit met onderdelen. Omdat het in dit specifieke geval ook om een SMD-soldeeroefening ging, zijn alleen die IC's voorgemonteerd die niet met de hand gesoldeerd kunnen worden. Wanneer u uw Gerber-bestanden en pick&place-gegevens aan een producent geeft, kunt u voor verrassingen komen te staan... In bijna de laatste fase van het proces hebben we een prototype ontvangen. Terwijl weerstanden, spoel, LED's en alle andere onderdelen correct in zakjes waren verpakt, zorgde de print voor een verrassing. Enkele componenten waren voorge-



Figuur 13. Print met foutief soldeermasker.



Figuur 14. De zelfbouw LiPo-supercharger netjes ingepakt.



Figuur 15. En dit zit er allemaal in de doos.

monteerd zoals de bedoeling was, maar er leek er een probleem te zijn met de soldeer-pasta. De betreffende laag van de print was namelijk zo geëxporteerd alsof *alle* componenten moesten worden geplaatst. En dus was soldeer-pasta aangebracht op alle pads op de print, waardoor het handmatig solderen van onderdelen moeilijk en gecompliceerd werd. In **figuur 13** ziet u die print, en krijgt u een indruk van het probleem. We moesten daarom een nieuwe set Gerber-bestanden maken met een aangepaste soldeer-pasta-laag, zodat tijdens de fabricage alleen op de pads van de voorgemonteerde componenten soldeer-pasta zou worden aangebracht.

USB-C PD EN DE WEERSTANDEM

Wanneer u een 'klassieke' USB-A-naar-USB-C kabel aansluit op een klassieke 5V/1A plug-in voeding, gaat het opladen prima. Dat is echter niet het geval bij USB-C PD (Power Delivery). Bij micro-USB worden gewoon VCC en GND gebruikt voor de voeding, maar bij USB-C PD zijn twee extra weerstanden nodig voor de instelling van de stroom. De zelfbouw LiPo-supercharger van GreatScott! gebruikt de bewezen micro-USB-aansluiting, wat betekent dat USB-C PD-adapters de schakeling in de meeste gevallen niet kunnen voeden.

Verpakking en handleiding

Tijd om het eindproduct te bekijken. In **figuur 14** ziet u de verpakte kit. Nadat alle onderdelen voor de kit zijn geproduceerd en verpakt, kunnen we niet volstaan met alles samen met een korte handleiding in een plastic zak te verpakken en te verzenden. Als u overstapt van het prototype naar productie voor de verkoop, moet u ook nadenken over



ONDERDELENLIJST ZELFBOUW LIPO-SUPERCHARGER

Weerstanden:

R1 = 84,5k, 1%, 0,25W, SMD 1206

R2 = 158k, 1%, 0,25W, SMD 1206

R3 = 18k, 1%, 0,25W, SMD 1206

R4 = 13k, 1%, 0,25W, SMD 1206

R5,R7 = 1k, 1%, 0,25W, SMD 1206

R6 = 536Ω, 1%, 0,25W, SMD 1206

R8,R9 = 1k5, 1%, 0,25W, SMD 1206

R10 = 10k, 1%, 0,25W, SMD 1206

R11 = 100k, 1%, 0,25W, SMD 1206

R12, 13 = 1k2, 1%, 0,25W, SMD 1206

R14 = 4k7, 1%, 0,25W, SMD 1206

R15 = 1k8, 1%, 0,25W, SMD 1206

C5 = 1μ, 10%, 50V, X7R, SMD 1206

C10 = 10n, 5%, 50V, X7R, SMD 1206

C12 = 470n, 10%, 50V, X7R, SMD 1206

C13 = 330μ, 20%, 35V, SMD 10,3x10,3

Spoelen:

L1 = 3μ3, 20%, 3,5A, 35mΩ, SMD

7,3x7,3x4,5 mm

Halfgeleiders:

D1,D2 = SL22-E3/52T, SMD SMB

LED1 = 11-21/GPC-AM2P1/2T, LED groen,

SMD 1206

LED2,LED3 = 15-21SDRC/S530-A2/TR8, LED
rood, SMD 1206

IC1 = TPS61085DGKR, SMD VSSOP-8

IC2 = XB8089D, SMD SOIC-8-EP

IC3 = BQ24092DGQR, SMD MSOP-10-EP

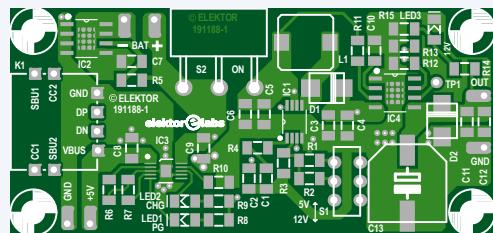
IC4 = TPS259621DDAR, SMD SO-PowerPad-8

Condensatoren:

C1 = 3n3, 5%, 50V, NP0, SMD 1206

C2,C7,C11 = 100n, 5%, 50V, C0G, SMD 1206

C3,C4,C6,C8,C9 = 10μ, 10%, 25V, X7R,
SMD 1206



Diversen:

K1...PC6 = pinheader, male, 1x1

S1 = schakelaar dubbelpolig 2-standen, THT,
9,1x3,5 mm (K2-2235D-F1)

S2 = schakelaar enkelpolig 2-standen, 250VAC,
3A (XKB, SS-12D06L5)

K1 = USB-C BoB, Elektor 191200-1

print 191188-1 v2.1



ONDERDELENLIJSTLIJST USB-C BOB

Weerstand:

R1 = 100k, 1%, 100mW, SMD 0603

Condensator:

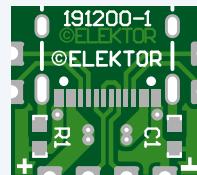
C1 = 100n, 10%, 100V, X7R, SMD 0603

Diversen:

K1 = USB-Type C, printconnector,

max. stroom 3A, SMD/THT

print 191200-1 v1.0



afgedrukt op 200% ware grootte

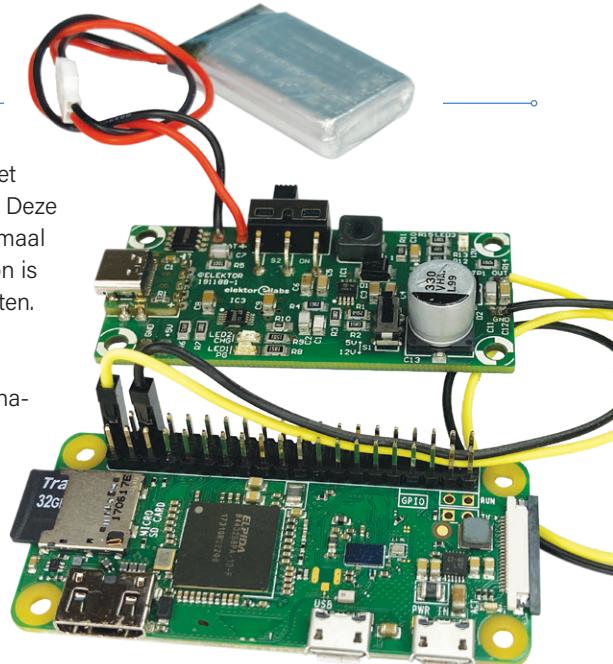
de verpakking en hoe u de koper van een handleiding kunt voorzien. Afhankelijk van de landen waar u producten verkoopt, kunnen er bepaalde regels voor het afdrukken van handleidingen in de betreffende talen gelden. Bij de zelfbouw LiPo-supercharger krijgt de koper zowel een geprinte QR-code met een link naar de instructies, als alle benodigde componenten (voorgesorteerd in geetiketterde zakjes). Door alleen de QR-code mee te sturen, kunnen we papier besparen en de planeet een beetje groener houden. In **figuur 15** ziet u wat er in de kit zit. U kunt een bouwvideo van GreatScott! bekijken op YouTube [3]. En als u twijfels hebt over het solderen van SMD-componenten, bekijk dan zijn tutorial [4]. Een oplaadbare accu is niet inbegrepen. Maar u kunt een geschikt exemplaar kopen in een online-shop, of u kunt er een gebruiken die u bij de hand hebt. Zorg ervoor dat de accu een laadstroom van 1 A verdraagt, en dat het een enkelcel lithium-ion- of lithium-polymeer-accu is.

Neem de tijd voor de montage, en wees een beetje geduldig. SMD-componenten, zelfs van formaat 1206, kunnen zich vrijwel onzichtbaar maken als ze van de werkbank vallen of zich achter andere onderdelen verschuilen. Als de bouw en een eerste test zijn afgerond, is de LiPo-supercharger klaar voor gebruik. Met schakelaar S2 op de print kunt u de accu van de DC/DC-omvormer aan- of afkoppelen. Zo kunt u de uitgang uitschakelen en bespaart u ook nog eens stroom, want zelfs een onbelaste DC/DC-omzetter kan enkele mA stroom trekken. Nadat u een accu hebt aangesloten, moet u die eerst helemaal laten opladen. LED1 van de LiPo-supercharger geeft aan dat de spanning die via de USB-C-connector binnen-

komt, in een geldig bereik ligt. Tijdens het opladen zie u dat LED2 gaat branden. Deze LED licht niet op op als de accu helemaal is geladen. Uw draagbare energiebron is dan klaar voor gebruik voor uw projecten.

De reis is het doel

Een draagbare stroombron met omschakelbare uitgangsspanning is buitengewoon handig. **Figuur 16** toont een Raspberry Pi Zero die wordt gevoed door de zelfbouw LiPo-supercharger. Als u een krap budget hebt en geen zin in solderen, kunt u natuurlijk verder klungelen met hotmelt, plakband, zwevende bedrading en goedkope Chinese componenten. Met deze kit kunt u echter alles zelf in elkaar zetten, wat door de SMD-soldeerervaring die u daarbij opdoet dubbel aantrekkelijk is. Een kant-en-klare oplossing die u uit een doos haalt en meteen kunt gebruiken, is veel minder leuk dan zelfbouw. Deze kit lijkt in dat opzicht op een Lego-project – u wilt het zelf maken en niet alleen maar hoeven uit te pakken. De reis is het doel en als u klaar bent, hebt u een handige draagbare stroombron die kan



Figuur 16. Een Raspberry Pi Zero die wordt gevoed door de LiPo-supercharger.

worden gebruikt voor de volgende projecten. U kunt alle schema's vinden op de Elektor Labs-pagina bij dit project [5]. En als u hulp nodig hebt, kunt u daar ook een bericht posten. 

191188-B-04

Een bijdrage van

Idee: **GreatScott!**

Ontwerp: **Ton Giesberts**

Auteur: **Mathias Claußen**

Redactie: **Jens Nickel, C.J. Abate**

Vertaling: **Eric Bogers**

Illustraties: **Patrick Wielders**

Layout: **Giel Dols**



GERELATEERDE PRODUCTEN

➤ **GreatScott! - DIY LiPo Supercharger Kit**
www.elektor.nl/diy-lipo-supercharger-kit-by-greatscott

➤ **Weller WE 1010 digitaal soldeerstation (Education Kit)**
www.elektor.nl/weller-we-1010-digital-soldering-station-education-kit

➤ **Fumetractor met LED-verlichting**
www.elektor.nl/fumetractor-with-led-light

Vragen of opmerkingen?

Heeft u technische vragen of opmerkingen? Stuur een e-mail naar de auteur via mathias.claussen@elektor.com of neem contact op met de redactie van Elektor via redactie@elektor.com.

WEBLINKS

- [1] Mechanische DC/DC-omzetter: [https://en.wikipedia.org/wiki/Vibrator_\(electronic\)](https://en.wikipedia.org/wiki/Vibrator_(electronic))
- [2] S. Cording, "The Modern Fuse," *Elektor Magazine.com*, 12.08.2020: www.elektormagazine.com/articles/modern-fuse
- [3] GreatScott!, "DIY LiPo Supercharger! (Charge, Protect, 5V/12V Boost V2)," 06.12.2020: <https://youtu.be/6LxRnf6sQNQ>
- [4] GreatScott!, "How to Solder Properly: Through-hole (THT) & Surface-Mount (SMD)" 03.09.2017: www.youtube.com/watch?v=VxMV6wGS3NY
- [5] Elektor Labs-pagina bij dit project: www.elektormagazine.com/labs/diy-lipo-supercharger-kit-by-greatscott

The cost and complexity of home computers is a serious deterrent to the newcomer to computer operating and programming. We know of many readers who would like to 'build their own' but who lack the necessary technical knowledge. The Junior Computer has been designed (for just this reason) as an attempt to 'open the door' to those readers who need a push in the right direction.

It should be emphasized that, although simple to construct, the Junior Computer is not a 'toy' but a fully workable computer system with the capability for future expansion. It has been designed for use by amateurs or experts, and software to be published will include a PASCAL compiler — the computer language of the future.

The purpose of this article is to give a general description of the operation and construction of the Junior Computer. It has been decided to publish a more detailed description in book form.

The arrival of 'The Junior Computer Books 1 and 2 on the market will be announced shortly. This, however, is a preview intended to reader an idea of what the computer entails.

... which microcontroller to use to control the robot and which language and/or programming you want to use for this purpose. If you buy the robot, you can choose from an add-on PCB for the PIC15F887 or a PCB for a AVR ATmega32, but with a piece of prototyping board and a minimum of soldering you can also control the robot with virtually any microcontroller, as long as it supports digital I/O, a couple of analog inputs, and RS232. Experience with the PIC232 or analogue inputs is by no means necessary. The comes with macros that radically simplify task of configuring the necessary settings. You can see from the overview in Figure 1 the block diagram in Figure 2, most of the components are controlled over the I2C bus. I2C protocol is very simple, and most microcontrollers have a hardware I2C interface module on board.

choice
programming language
sample programs in both Flowcode and c
Proton



Dit is de cover van de eerste Electronica Wereld. Later werd de naam gewijzigd in Elektuur – bedoeld als 'e-lectuur'. In 1970 verscheen de eerste Duitse editie onder de naam Elektor.

Elektor

gedachten bij zestig jaar elektronica

when I presented

my prototype of
the inverted
pendulum, a
colleague
asked me:

"And so you
just woke
up one mor-
ning and said
today I'm
going to make
an upside-
down pen-
cil?"

By Jean-Sébastien

a moving trolley, a quadcopter
a rod with a gliss balanced on
But none of these matches

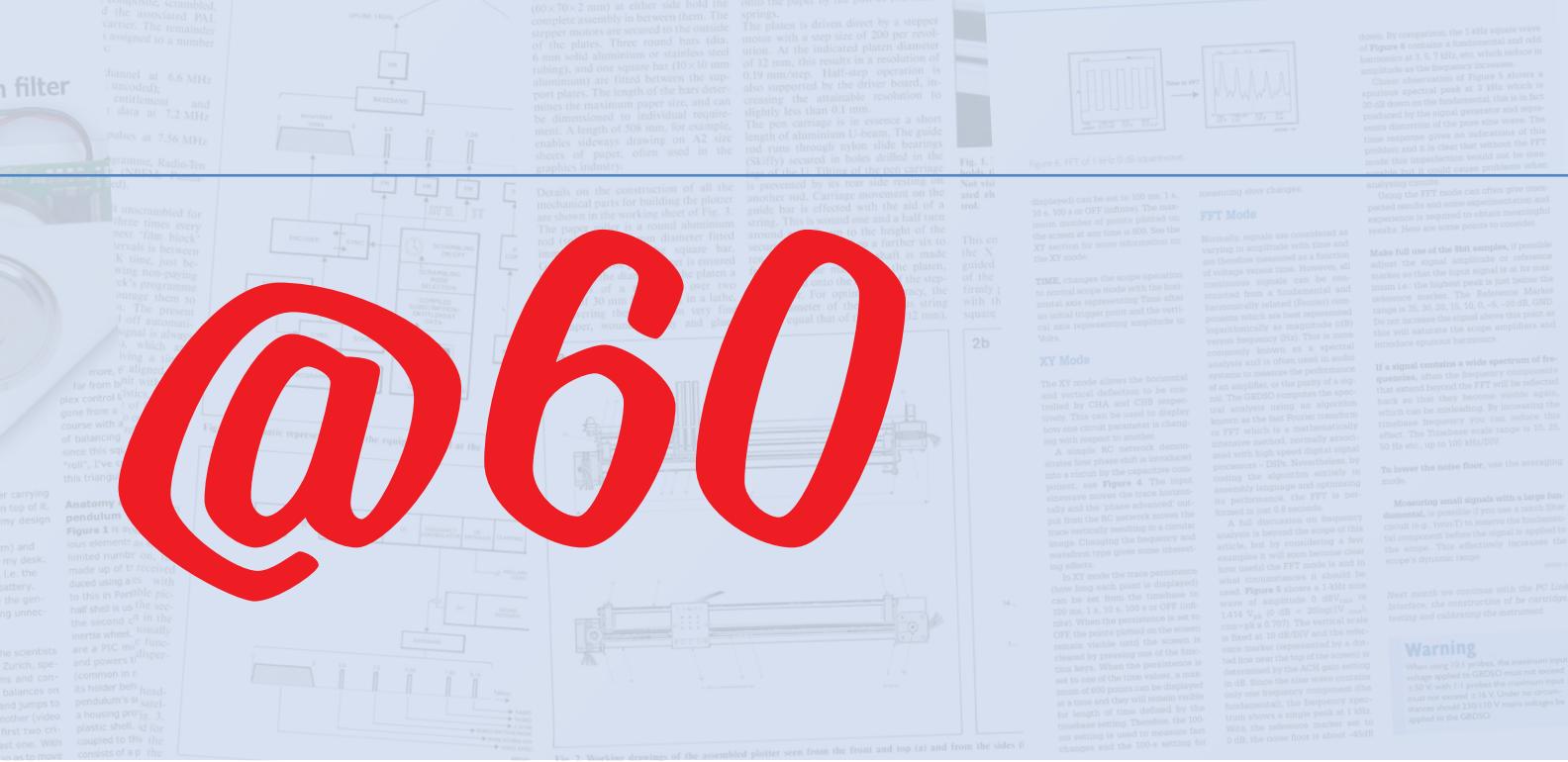
criteria, namely:

- Compact (around 4" / 10 cm)
- Light, so I can display it on
- Completely self-contained,
- Pendulum carries its own brain
- Simple, so as to demystify
- Principle without adding
- Complexity.

I found a recent project by Y.
at the Federal Polytechnic, Zurich, Switzerland:
in dynamic self-contain-
control: Cubi. It's a cube that
one of its edges or corners
move from one edge to the other
([2]). Cubi does meet my cri-
teria, but maybe not the

Jens Nickel (Elektor)

Zestig jaar Elektor magazine!
Dat is behoorlijk lang voor een tijdschrift over elektronica, een onderwerp dat zo snel evolueert dat het moeilijk is om alle ontwikkelingen bij te houden. Sinds 1961 hebben we niet alleen gerapporteerd over de nieuwste technologieën, apparaten, boards en componenten (met die typische mix van theorie en praktijk), we hebben onze ideeën en projecten gedeeld met zowel professionele elektronici als makers. Ter gelegenheid van ons jubileum hebben we onlangs een aantal van onze auteurs en redacteuren gevraagd om uit die zestig jaar Elektor die projecten en artikelen te selecteren die zij als het meest interessant beschouwen!



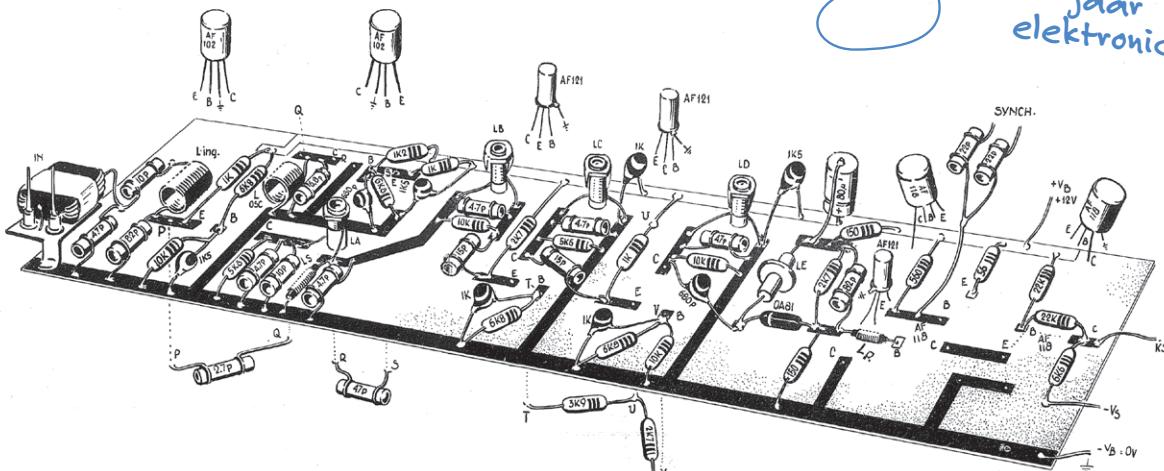
Ons bekende tijdschrift werd in april 1961 in Nederland gelanceerd. Niemand kan daar beter over vertellen dan Harry Baggen, die tientallen jaren hoofdredacteur was van de Nederlandse editie en ook jarenlang als internationaal hoofdredacteur actief was.

"Toen Bob W. van der Horst zijn eigen elektronica magazine startte onder de naam 'Electronica Wereld', had hij zich als doel gesteld een beter tijdschrift te maken dan wat er al bestond. Hij wilde af van die mystieke sluier rond het onderwerp en vooral de praktische kant belichten. In het begin bestond het magazine voornamelijk uit productrecensies en technische achtergrondartikelen, maar na een paar jaar kwam de praktijk steeds meer op de voorgrond. En zo is het tot op de dag van vandaag gebleven. Geen eindeloze theoretische verhandelingen – integendeel: praktische schakelingen met eigentijdse componenten, dat was en is het doel."

"Bob wilde ook altijd voorop lopen als het om nieuwe componenten ging. In de eerste jaren waren het nog vooral buizenschakelingen die in het tijdschrift verschenen, maar al snel werd hun plaats ingenomen

door halfgeleiders. Behalve transistoren werden ook veel FET's gebruikt. Elektor stond al snel bekend als eigenwijs, tegendraads, innovatief. Precies wat Van der Horst voor ogen stond!"

In dat eerste nummer stonden meer dan 50 schakelingen, zoals een vervormingsmeter en een ééntransistor-zender (dit nummer kan gratis worden gedownload van www.elektrormagazine.nl/210025-03). En er zouden nog veel, veel meer projecten volgen! Op de volgende pagina's treft u een aardige selectie van projecten plus commentaar van experts en auteurs. Elektor-leden kunnen de betreffende artikelen downloaden via de aangegeven weblinks. Om het 60-jarig jubileum van Elektor te vieren, hebben we tot 30 juni 2021 zeven van deze door de redactie uitgekozen artikelen gratis beschikbaar gesteld. Veel plezier!



Een passie voor een praktische benadering van elektronica zit in de genen van Elektor. Vanaf het allereerste begin wilden we dat onze lezers al bouwend zouden leren. Daarom waren onze projecten altijd geliefd binnen de Elektor-community.

down. By comparison, the 1-kHz square wave of Figure 8 contains a fundamental and odd harmonics at 3, 5, 7, 9, etc., which reduce in amplitude as frequency increases.

Closer observation of Figure 8 shows a spurious spectral peak at 3 kHz which is 20 dB down on the fundamental, that is, it is produced by the noise floor of the spectrum. The noise floor gives us indications of this problem and it is clear that without the FFT mode this imperfection would not be noticed.

Analysing current.

Once the FFT mode can often give unexpected results and some experimentation and experience is required to obtain meaningful results. Here are some points to consider:

Make full use of the FFT samples, if possible. Note the signal amplitude or reference marker so that the input signal is at its maximum i.e. the highest peak is just below the reference marker. The Reference Marker range is 30, 30, 10, 10, 10, -5, -5 dB, GND. Do not let the signal exceed this point as this will saturate the scope amplifier and introduce spurious harmonics.

If a signal contains a wide spectrum of frequencies, it is possible that some components may be reflected back from the FFT window so that they become visible again, which can be misleading. By increasing the timebase frequency you can eliminate this effect. The Timebase range is 10, 25, 50, 100 ms, up to 100 kHz/20V.

To lower the noise floor, use the averaging mode.

Measuring small signals with a large fundamental, it is possible if you use a notch filter circuit (e.g. zeta-T) to remove the fundamental component before the signal is applied to the scope. This effectively increases the scope's dynamic range.

Next month we continue with the PC Link Interface, the construction of the hardware, testing and calibrating the instrument.

Warning

When using 10:1 probes, the maximum input voltage applied to GBD50 must not exceed 50 V. With 1:1 probes the maximum input must not exceed 100 V. The maximum input voltage applied to GBD50 100 V must be applied to the GND.



Harry Baggen (voormalig hoofdredacteur)

Edwin-versterker (1975)

Elektuur stond vanaf zijn beginjaren bekend om zijn bijzondere audio-projecten voor zelfbouw en dat is tot de dag van vandaag zo gebleven. Een 'baanbrekend' ontwerp was in 1970 de Edwin-versterker, een naar huidige begrippen heel simpele audioversterker die voorzien was van een ruststroomloze uitgangstrap. Het werd een klassieker en het is een van de meest nagebouwde audioversterkers van de vorige eeuw.

www.elektormagazine.com/magazine/elektor-197509/57506

Elektorskoop (1975)

In de zeventiger jaren waren oscilloscopen voor hobbyisten bijna onbetaalbaar. Dat was voor de ontwerpers van Elektuur in die tijd een goede reden om een oscilloscoop voor zelfbouw te ontwikkelen en publiceren. Destijds een moeilijke klus, want vooral de oscilloscoopbuis, de bijbehorende aansturing en hoogspanning moesten zorgvuldig op elkaar worden afgestemd terwijl veel van die onderdelen (vooral de buis) heel moeilijk verkrijgbaar waren. Toch lukte het om een nabouwzekere schakeling te presenteren.

www.elektormagazine.com/magazine/elektor-197612/57794

Microprocessorgestuurde frekwentiometer (1985)

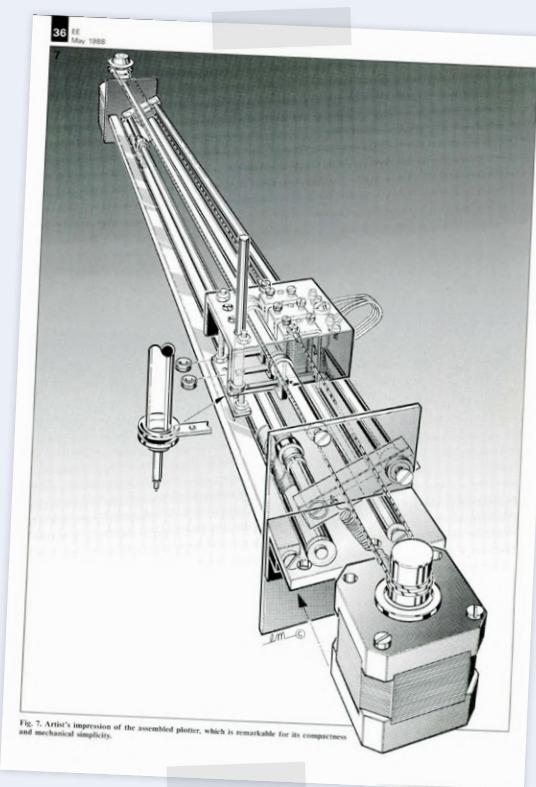
Begin jaren '80 bracht Elektuur een serie meetapparaten uit waarvan de μ P-gestuurde frekwentiometer (ja, toen nog met kw!) het hoogtepunt was. Dit juweeltje kon zich qua mogelijkheden en eigenschappen gemakkelijk meten met professionele apparatuur. Het apparaat had een frequentiebereik tot 1,2 GHz, autoranging, folietoetsen voor de bediening en een alfanumeriek display.

www.elektormagazine.nl/magazine/elektor-198501/43763

Mondriaan-plotter (1987)

In een tijd dat plotters alleen nog maar in laboratoria en ingenieursbureaus te vinden waren, verscheen in Elektuur het ontwerp van een bijzonder eenvoudige zelfbouw-plotter met de naam Mondriaan (MONtagevriendelijke DRkleurenplotter Als Aantrekkelijk Nabouwproject). De onderdelen voor de plotter kostten een schijntje ten opzichte van een echte plotter en als tekenpennen werden viltstiften gebruikt. Er verscheen later zelfs een HP-GL-driver voor de Mondriaan.

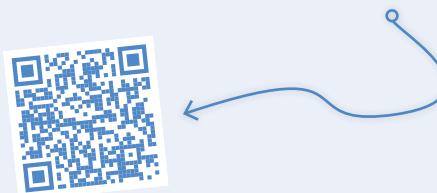
www.elektormagazine.nl/magazine/elektor-198710/44349



Surround-sound-dekoder (1995)

In een tijd dat surround-sound-apparaten juist op de markt verschenen, kwam Elektuur al met een zelfbouwschakeling. De schakeling werkte zonder speciale Dolby-IC's omdat die voor zelfbouw niet beschikbaar waren, maar toch was het Elektuur-lab erin geslaagd om met conventionele middelen een decoder te ontwerpen die het bestaande stereo-geluid uitbreidde met een midden- en een surround-kanaal. De printen waren niet aan te slepen.

www.elektormagazine.nl/magazine/elektor-199501/38379





Jan Buiting (boekredacteur, voormalig hoofdredacteur Engelse editie)

ATN-Filmnet decoder (1989)

Nooit gedurende mijn 36-jarig bij Elektor heb ik meer enthousiasme, lof en algemene 'weerklank' van de DHZ-elektronica-community ervaren en genoten dan met de publicatie van de 'ATN-Filmnet Decoder' in 1989. Dit briljant ontworpen project heeft Elektor onmiddellijk neergezet als een toonaangevende bron van 'doordachte, grappige en verfijnde' publicaties over het hacken van pay-TV-kanalen met pan-Europese dekking, allemaal dankzij enkele Astra- en Intelsat-satellieten. Alleen al in Scandinavië waren naar schatting 20.000 Elektor Filmnet-decoders in gebruik. "Alleen voor privé en experimenteel gebruik, Jan" – uiteraard! En: "Onze winters zijn lang en donker, Jan" – natuurlijk!

Zowel het ontwerp van het project als de artikelpublicatie waren hoogst ongebruikelijk voor Elektor. Ten eerste werd het project niet in de Nederlandse Elektuur gepubliceerd uit vrees voor juridische consequenties. Ten tweede, eveneens uit angst voor advocaten aan de telefoon of voor de deur, stond de print niet op de toenmalige 'EPS'-pagina's (nu de Elektor-shop). En ten derde publiceerde de ontwerper onder het pseudoniem "P.N.P. Wintergreen", naar een personage in Joseph Heller's boek Catch-22.

Wat door de uitgevers van Elektor als een 'vreemde eend in de bijt' werd beschouwd, heeft in feite duizenden hobbyisten in heel Europa ertoe aangezet om de decoder te bouwen. Het bericht verspreidde zich al snel dat Filmnet grandios was gehackt door Elektor. De Engelse editie van Elektor was voor het eerst sinds 1974 compleet uitverkocht, zodat een tweede oplage moet worden gedrukt. De Nederlanders werden 'naar behoren bediend' door stapels illegale kopieën van het originele Engelstalige artikel dat door elektronica-winkels werd verspreid. De dames van de telefooncentrale van Elektor lazen wekenlang een voorbereide tekst voor als reactie op een stormvloed aan telefoontjes. 'Door iemand gemaakte' printen werden illegaal verkocht in winkels en op grote radioamateurbijeenkomsten in het Verenigd Koninkrijk, Duitsland en Nederland.

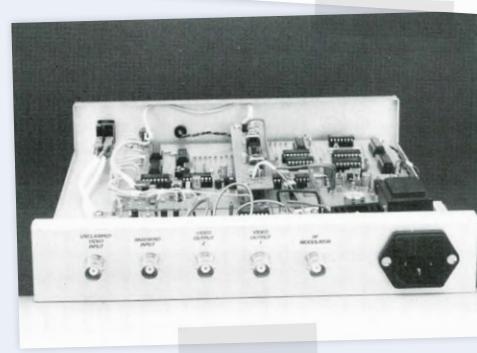
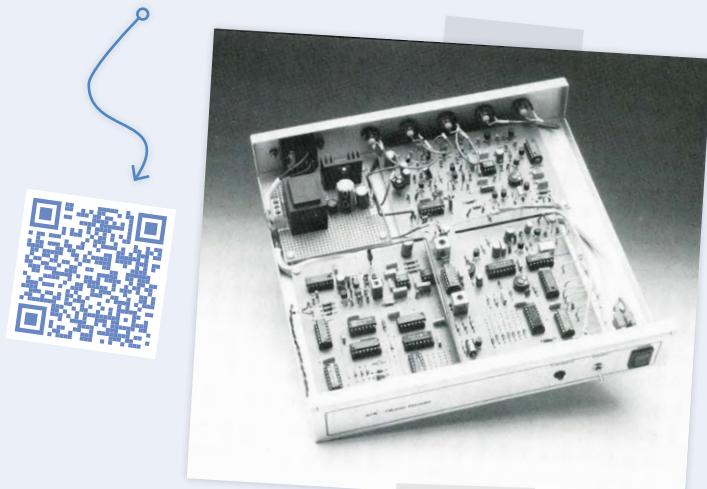
Het elektronische ontwerp van de decoder is een briljant voorbeeld van de jaren-80-stijl, met niets dan goedkope

componenten, waaronder CMOS IC's, de TBA120S en veel PNP-transistoren. Geen microcontrollers of andere nieuwertigheden – de gloriedagen van de door Elektor uitgevonden 'TUN&TUP' herleefden.

Filmnet veranderde hun scramble-formaat verschillende keren in de maanden na de Elektor-publicatie, maar werd keer op keer verslagen door slimme updates van de Elektor-decoder. Ik had de eer om deze tweaks in een paar afleveringen onder de naam *From the Satellite TV Desk* te publiceren. Na drie updates en corresponderende teksten voor de dames van onze telefooncentrale kwam P.N.P. Wintergreen met zijn ingenieuze 'intelligente aanpassing' – een toevoeging op basis van een sample&hold-schakeling met één enkele condensator. Vanaf dat moment merkten de bezitters van een actuele Elektor Filmnet-decoder niet eens meer dat de scrambling-modus was veranderd. Het was glorieus hacken van een erg duur scrambling-systeem dat aan de eigenaren van Filmnet was verkocht als 'uiterst problematisch om te omzeilen'. In latere jaren moesten de 'SAVE'-scrambling van BBC World TV Service en verschillende DMAC/CMAC/Irdeto satelliet-TV-zenders ook capituleren voor hacking door Elektor met even formidabele resultaten, enorme bijval van lezers en een Q-factor waarvan je vandaag de dag alleen maar kunt dromen.

Ik ken de ontwerper persoonlijk, maar hij blijft anoniem. Maar hij is een volbloed elektronicus. "Problematisch? – Dat woord komt in mijn elekronica-woordenboek niet voor."

www.elektormagazine.com/magazine/elektor-198903/47520





Luc Lemmens (Elektor Lab)

8052-BASIC-compuboard (1987)

8032-mini-board (1991)

MCS51-assembler kursus (1992)

Een van de eerste grotere projecten die ik begeleidde in het Elektor Lab was het 8032-mini-board, gepubliceerd in april 1991, als opvolger van het populaire 8052-BASIC-compuboard uit 1987. Het was niet zo groot wat de hardware betreft, maar dit artikel werd gevolgd door een achtdelige assembler-cursus met behulp van dit board, die door veel lezers werd gevuld. Voor degenen die geen assembler wilden leren, kon de 8052AH-BASIC microcontroller van Intel ook op dit board worden gebruikt. Met een klein programma kon de interne BASIC-interpreter (het volledige programmageheugen) 'illegal' worden gekopieerd naar een externe EPROM.

www.elektormagazine.nl/magazine/elektor-198711/45549

www.elektormagazine.nl/magazine/elektor-199104/37501

www.elektormagazine.nl/magazine/elektor-199209/37875



GBDSO Gameboy digitale oscilloscoop (2000)

De Nintendo Gameboy Digital Sampling Oscilloscope (GBDSO) die in het oktober-nummer van 2000 werd gepresenteerd, was opnieuw een doorslaand succes en een van de eerste volledig geassembleerde modules die in de Elektor-shop verkrijgbaar was. In eerste instantie zouden we maar één batch laten produceren, maar uiteindelijk bleek de vraag veel groter dan we hadden verwacht. Ik ben de tel kwijtgeraakt hoevaak we een 'allerlaatste batch' hebben besteld. Blijkbaar waren er veel lezers die nog ergens een Gameboy hadden liggen en die een tweede leven wilden geven als draagbaar meetinstrument.

www.elektormagazine.nl/magazine/elektor-200010/13600

Reukgenerator met CD4711 (1986)

De beroemde jaarlijkse Halfgeleidergids van Elektor was de dubbele uitgave voor de maanden juli en augustus, gevuld met meer dan honderd kleine, vaak opmerkelijke (sub-)schakelingen, tips en trucs. Een standaard feature was de 'grapschakeling', waarvan de reukgenerator met CD4711 uit 1986 een van de bekendste is. Jaren later kregen we nog steeds vragen van lezers of we contactgegevens hadden van de fabrikant Odorant Elektronik GmbH in Keulen.

www.elektormagazine.nl/magazine/elektor-198599/44043



Jens Nickel (hoofdredacteur)

Aan de slag met Embedded Linux (2012)

Ik was in 2012 erg trots als kersverse Duitse hoofdredacteur: in vergelijking met het voorgaande jaar was de betaalde oplage met 20% toegenomen. Maar dat was niet mijn verdienste, maar te danken aan een titelproject dat zijn tijd ver vooruit was.

Auteur Benedikt Sauter en zijn team hadden een compact board ontwikkeld waarop Linux draaide en dat voor die tijd ongelooflijk goedkoop was – slechts € 50!

www.elektormagazine.nl/magazine/elektor-201205/16548

Een eigen bus (2011)

In mijn eerste jaren als Elektor-redacteur was ik de enige die niet met de soldeerbout bezig was. Aan mijn 'contactvrees' voor praktische elektronica moest dringend iets worden gedaan. De toenmalige internationale hoofdredacteur liet me daarom een column schrijven over de mensen in ons lab. Hij realiseerde zich echter niet wat in mij losmaakte! Omdat er niet veel nieuws in het lab was, concentreerde ik me al snel op een bussysteem dat voor allerlei besturingen gebruikt kon worden. De feedback was overweldigend! Ik kreeg honderden mails en al snel kwam een groep van een tiental innovators bij elkaar om de specificaties van de ElektorBus te definiëren. Maar natuurlijk was er niet alleen lof... (de ElektorBus werd ook bestempeld als 'amateuristisch' en 'een ontwerp uit de jaren '60')

Later werden diverse projecten op die bus gebaseerd, en hij werd ook gebruikt in een afstudeerscriptie. Ik kreeg zelfs een telefoonje van een installatiebedrijf dat modules wilde ontwikkelen. Maar uitgerekend de mensen uit ons lab konden weinig genegenheid voor het project opbrengen. En dus, tot verdriet van veel lezers, stierf dit grootse project een stille dood.

www.elektormagazine.nl/magazine/elektor-201101/16229

Multifunctioneel Xmega-board (2013)

Mijn passie voor het ontwikkelen was kennelijk gewekt. Twee jaar later kon ik samen met een paar professionele engineers mijn eigen microcontrollerboard ontwerpen. We hebben er een Xmega en heel veel periferie op gezet, maar uiteindelijk werd het hele ding verkocht voor ongeveer € 100 (het display niet meegerekend). De engineers schonken hun ontwikkeltijd, maar ze hadden boards en prototypes gemaakt ter waarde van ongeveer € 1000. En deze kosten konden we niet terugverdienen. Een flop! En ik had veel energie in de software gestoken, zelfs een framework gemaakt met de Embedded Firmware Library dat niet alleen van de controllerhardware maar ook van de bedrading van het board abstracteerde. Een van onze lezers en ikzelf hadden veel programmeerpleszier tijdens het proces. Ik kreeg feedback van lezers die mijn aanpak 'ingenieus' vonden. Weer anderen beschouwden het geheel als overbodig. Maar wat uiteindelijk telt, is dat ik veel geleerd heb en misschien zelfs de community een paar ideeën heb meegegeven – dat is waar het bij Elektor om draait!

www.elektormagazine.nl/magazine/elektor-201310/23450
www.elektormagazine.nl/magazine/elektor-201305/20542

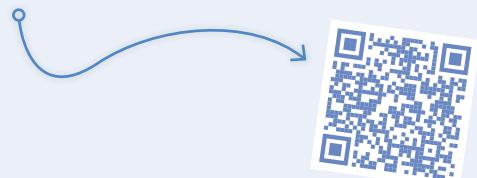


Clemens Valens (Elektor Lab)

EPROM Simulator (1989)

Toen ik begon met wat nu embedded systeemontwerp wordt genoemd, ontdekte ik al snel dat het herprogrammeren van EPROM's nogal tijdrovend was. Dus toen Elektor in 1989 de EPROM-simulator publiceerde, heb ik er meteen een gebouwd. Het uploaden van firmware naar een target-board werd net zo eenvoudig als het kopiëren van een bestand naar de parallele printerpoort van de PC. Veel sneller dan Arduino vandaag en zonder bootloaders; mijn collega's en ik hebben het jarenlang gebruikt. Totdat de parallele poort van de PC verdween en microcontrollers met ingebouwd flash-geheugen aan de macht kwamen...

www.elektormagazine.com/magazine/elektor-198912/47689



i-Pendule (2016)

De i-Pendule is misschien niet het nuttigste apparaat dat ooit is gepubliceerd, maar het was een uitstekend project! Geplaatst op een horizontaal oppervlak en na inschakelen springt hij plotseling op en blijft hij onbeweeglijk op zijn punt staan. Het was mijn taak om het Elektor-artikel te produceren en het prototype te bouwen.



Het ontwerp van Jean-Sébastien Gonsette was erg goed gedaan, maar het met de hand solderen van het IC voor de motorbesturing was een ramp. Op de een of andere manier is het me toch gelukt zodat ik een werkend apparaat had. Toen ik het aan een vriend liet zien, rolde hij over de vloer van het lachen. Het was een van de meest bevredigende reacties die ik ooit heb gekregen op iets dat ik zelf had gebouwd.

www.elektormagazine.nl/magazine/elektor-201605/28994



Halfgeleidergids

Ik kocht Elektor alleen als er iets in stond dat me echt interesseerde, met uitzondering van het jaarlijkse dubbelnummer, de Halfgeleidergids die ik elk jaar ongezien kocht. Naast de meer dan 100 schakelingen vond ik de advertenties ook leuk, vooral de catalogusachtige advertenties met honderden componenten op meerdere pagina's. Ik

60 jaar elektronica

bleef die ongezien kopen tot 2007, toen alles voor het eerst (en voor het laatst) om één thema draaide: robotica. 95% van de schakelingen betrof robotica en ik vond er niets aan. Het jaar daarop hoeftde ik de Halfgeleidergids niet meer te kopen: ik werkte inmiddels bij Elektor.

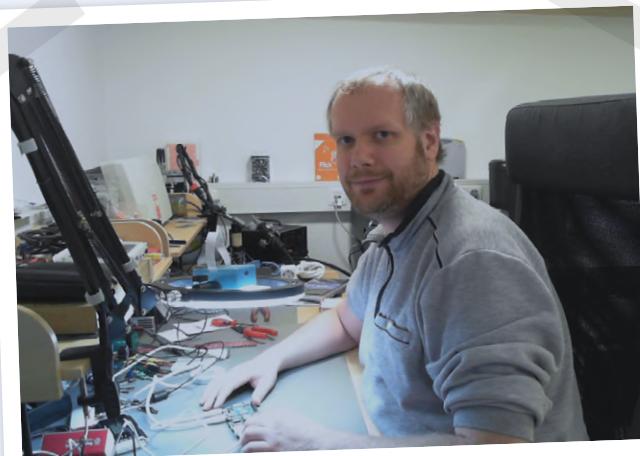
www.elektormagazine.nl/magazine/elektor-198499

www.elektormagazine.nl/magazine/elektor-200707

Formant (1977)

Te jong toen het werd gepubliceerd, niet genoeg geld toen ik geïnteresseerd raakte, kreeg ik eindelijk een Formant-muzieksynthesizer in handen toen twee vrienden samen een bijna werkend exemplaar kochten. Het belandde snel bij mij in mijn slaapkamer waar ik de bedrading vernieuwde, de omhullende-generatoren vervanging door een ontwerp uit een concurrerend elektronicamagazine en mijn gloednieuwe zelfgebouwde digitale frequentiemeter gebruikte om het geheel te kalibreren. Samen met mijn vrienden hebben we vele, vele uren op de tot een nieuw leven gewekte machine gespeeld.

www.elektormagazine.com/magazine/elektor-197704/57836



Mathias Clausen (Elektor Lab)

Logic analyzer voor Atari ST (1989)

Mijn eerste computer was een Atari ST, dus was ik best wel onder de indruk om in het archief te ontdekken dat in die tijd logic analyzers rond die machines werden gebouwd. Met slechts een paar logische IC's en goed geschreven software werd uw ST een achtkanaals logic analyzer met een samplefrequentie van 2 MHz. Dit was mogelijk door slim gebruik te maken van de externe HDD-interface van de ST. Het idee, gepubliceerd in 1989, werd bijna twee decennia later herontdekt met de Saleae Logic 8. In plaats van een 'oneigelijk' gebruikte HDD-interface en enkele 74HC-chips gebruikte de Logic 8 een microcontroller en USB. Natuurlijk met een betere sampling rate en meer geavanceerde software, maar het principe bleef hetzelfde.

www.elektormagazine.nl/magazine/elektor-198910/45945

Junior Computer (1980)

Om een beetje in deze retro-stijl te blijven: er zijn verspreid over de hele wereld nog steeds ontwikkelaars die graag code schrijven of systemen bouwen rond de MOS6502 en de afgeleiden daarvan. Hij was de CPU van veel bekende systemen, zoals de Nintendo NES, Atari 800, Apple II en (in een aangepaste versie) in de C64. In 1980 publiceerde Elektor de Junior Computer, een op de 6502 gebaseerd apparaat dat een instap in de wereld van microprocessoren bood. Met een kloksnelheid van 1 MHz, 1 KB RAM en 1 KB ROM was hij trager en minder krachtig dan een moderne ATmega-gebaseerde Arduino – maar destijds kon je dit zelf bouwen en repareren. Er zijn nog dertig jaar oude 6502-gebaseerde systemen in omloop, en met de Commander X16 van David Murray zit er zelfs een nieuw ontwerp op basis van deze CPU aan te komen. En als u het leuk vindt om uw eigen 6502-gebaseerde computers te bouwen: de CPU's worden ook vandaag de dag nog steeds geproduceerd.

www.elektormagazine.nl/magazine/elektor-198003/42691



E-Lock, de eerste Elektor-chip (2014)

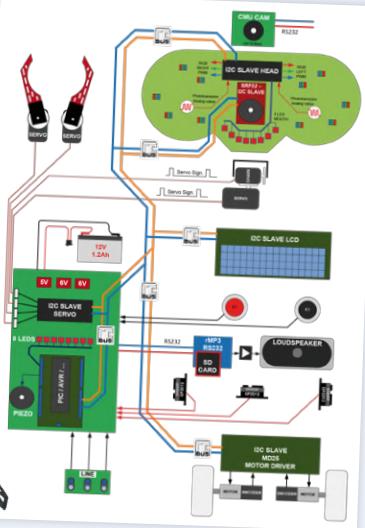
Over microchips gesproken, ook Elektor heeft een eigen chip geproduceerd. In 2014 verscheen de E-Lock, met daarin geïntegreerd bedraad Ethernet, ADC, I²C, DAC en enige GPIO's met veilige communicatie en codering. Uitgerust met hardware-encryptie en beveiligingselementen, was de chip bedoeld voor gebruik in het komende IoT.

Dit is een van de merkwaardiger Elektor-projecten, aangezien een deel van het innerlijk van deze chip een raadsel bleef. We zullen misschien nooit weten welke kern is gebruikt en hoe de interne registers werkten. Nieuwere chips zoals de ESP8266 of enkele WIZnet hebben zijn plaats ingenomen. Bedraad Ethernet heeft nog steeds bestaansrecht, maar gebruikers vragen nu steeds meer om draadloze IoT-oplossingen.

www.elektormagazine.nl/magazine/elektor-201404/26259

Elektor Proton Robot (2011)

Deze heeft op kantoor helaas veel te lang stof verzameld: de Elektor Proton. Een zelfrijdende robot, ontwikkeld als leerplatform voor scholieren en studenten. Met een modular ontwerp op basis van een I²C-bus was de Proton ontworpen om snel en eenvoudig toegang te geven tot de wereld van de robotica, met een hele reeks functies die het gemakkelijk maken voor zowel beginners als professionals om de Proton voor eigen doeleinden te gebruiken.



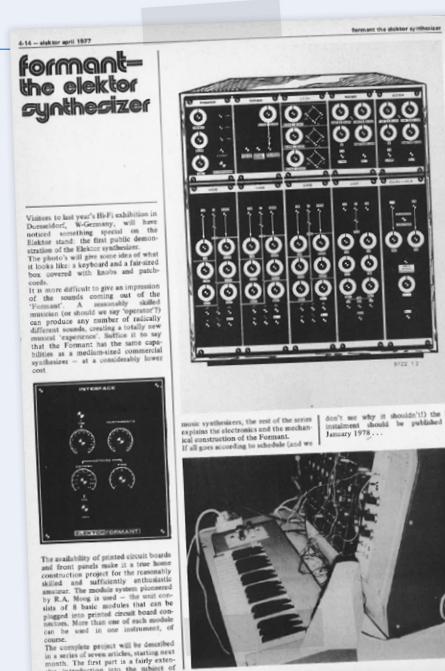
Aangezien er na zijn eerste optreden in 2011 inmiddels tien jaar zijn verstreken, zou de Proton die hier nog staat een update kunnen gebruiken met moderne nieuwe sensoren, vision en een krachtiger MCU als hart (misschien zelfs met WiFi). Dankzij de gebruikte I²C-bus zou dit eigenlijk geen probleem mogen zijn.

www.elektormagazine.nl/magazine/elektor-201105/16290



C. J. Abate (Content Director)

Velen van u zijn waarschijnlijk bekend met het 'YouTube-fenomeen'. U weet hoe het gaat: u begint op een avond na het eten met de GitHub-video van Clemens Valens (<http://bit.ly/elektor-github>) op het ElektorTV-kanaal – en komt twee uur later tot het besef dat u verdiept bent in een video over het bouwen van een blokhut in een uithoek van Alaska zonder stroom. Maar kent u ook het 'Elektor Magazine-fenomeen'? Ik heb niet van elk nummer van Elektor een papieren exemplaar in mijn thuisbibliotheek, maar ik heb wel toegang tot het digitale archief. In de afgelopen jaren heb ik daar talloze uren doorgebracht, kikkend en bladerend door de PDF's. Op een avond, na het bekijken van een video op YouTube die met de Elektor Formant te maken had, ging ik in het Elektor-archief zoeken naar de eerste artikelen over dat geliefde muzieksynthesizer-project. Anderhalf uur later merkte ik dat ik een artikel over een radardetecteur uit 1991 aan het lezen was. Met tientallen jaren aan inhoud onder handbereik is het moeilijk om een favoriet te kiezen. Hier zijn dus enkele artikelen die ik van harte kan aanbevelen, omdat ze een prima inleidingen bieden tot enkele onderwerpen die ik werkelijk interessant vind.



Formant (1977)

Dit project is van ver voor mijn tijd. Ik had er echter al een heleboel over gehoord, zodat ik onlangs besloot om wat onderzoek te doen. Natuurlijk kunt u op YouTube zelfs mensen vinden die muziek maken met de Formant!

www.elektormagazine.com/magazine/elektor-197705/57847

Het beest getemd: ontwerp je eigen chip (2012)

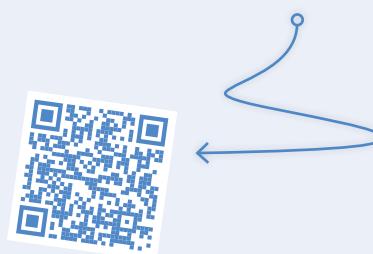
Enkele jaren geleden reisden Elektor-redacteur Jan Buiting en ik naar Californië om Xilinx te bezoeken. Terwijl ik enkele engineers ontmoette en de fabriek bezichtigde, besefte ik al snel hoe weinig ik ovr de FPGA-technologie wist. Het artikel "Het beest getemd" is een goed uitgangspunt.

www.elektormagazine.nl/magazine/elektor-201212/16688

Elektronenbuizen (1984)

Wie is er niet gefascineerd door buizen – u weet wel, die 'breekbare glazen dingen met al die ingewikkelde mechaniek erin'? Dit artikel is Elektor op z'n best. Of je nu een oude-stijl elektronica-liefhebber bent of een liefhebber van eigentijdse ontwerpen die hoogwaardige elektronica combineren met cool ogende retro-buizen, u zult dit een geweldig artikel vinden.

www.elektormagazine.nl/magazine/elektor-198411/43740





Michel Kuenemann (auteur)

Ik ontdekte Elektor halverwege de jaren '80 en ik was verbaasd over de kwaliteit van hun projecten. Ik heb de in 1985 gepubliceerde frequentieteller op basis van een 6502 gebouwd, en die doet het nog steeds. Tijdens mijn studie elektrotechniek was ik betrokken bij een vereniging van raketbouwers. We wilden de verandering van de atmosferische druk tijdens de klim van de raket meten. Elektor had een heel aardig project gepubliceerd met een druksensor en verschillende opamps. Het board is uiteindelijk met goed gevolg in de raket ingebouwd. Na mijn afstuderen heb ik een bedrijf opgericht en Elektor bleef door de jaren heen voor mij een belangrijke inspiratiebron. Dankzij Elektor heb ik de I²C-bus en vele andere technologieën ontdekt. Later kon ik bijdragen aan het tijdschrift met verschillende artikelen die betrekking hadden op de modelbouw, zoals een testbank voor model-verbrandingsmotoren (april 2009) en een op ZigBee gebaseerde afstandsbediening (september 2011). Ik ben nu 58 en Elektor blijft een grote invloed op mij uitoefenen. Een gelukkig 60-jarig jubileum voor Elektor en hartelijk dank aan het team en de auteurs voor al die goede artikelen!

Artikelen van Michel Kuenemann:

www.electormagazine.com/authors/171/michel-kuenemann



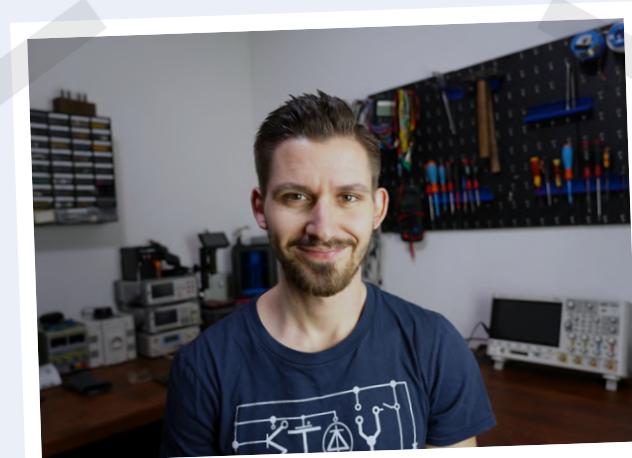
Andrew Pratt (auteur)

Ik las Elektor voor het eerst in de jaren '70, lang voor de komst van het internet. Technische tijdschriften zorgden voor informatie en nuttige advertenties over componenten en apparatuur. Nu biedt het internet bijna onbeperkte informatie, dus waar hebben we tijdschriften nog voor nodig? Websites zijn handig, maar fysieke media hebben veel voordelen. Elektor biedt beide. De website

rangschikt de informatie op één plek, zodat u snel vindt wat u zoekt. Drie van mijn boeken zijn door Elektor uitgegeven. Een boek of een artikel samen met praktijkoeferingen is een goede manier om te leren hoe dingen werken. U kunt altijd experimenteren, zelfs als u nooit een compleet project bouwt. Ook bestaat voor iedereen de mogelijkheid om een project te uploaden naar Elektor Labs Online, zodat anderen dat kunnen kopiëren of hun eigen werk verbeteren. Open source-software is een spectaculair succes, Elektor doet dit ook voor elektronica.

Boeken van Andrew Pratt:

[www.elektor.nl/catalogsearch/result/?q=Andrew Pratt](http://www.elektor.nl/catalogsearch/result/?q=Andrew%20Pratt)



GreatScott! (auteur, YouTuber)

Het heeft best een tijdje geduurd voordat ik Elektor ontdekte. Ik had al jarenlang YouTube-video's over elektronica geproduceerd, toen me een aanbeveling van een kijker opviel. Het ging over een bouwpakket van Elektor. De link bracht me naar de Elektor-website, waar ik enkele uren heb doorgebracht. Ik merkte dat de behandelde onderwerpen overeenkomsten vertoonden met mijn eigen elektronica-video's. Een paar maanden later werd ik benaderd door Elektor met de vraag of we samen enkele videoproducties konden doen – en de rest is geschiedenis. Ik ging Elektor lezen en in samenwerking met Elektor produceer ik video's over uiteenlopende elektronica-onderwerpen. Een win-winsituatie voor iedereen!

Gratis online artikel over de DIY LiPo Supercharger

Kit van GreatScott!: www.electormagazine.nl/articles/zelfbouw-lipo-superchargerbundel



Siglent SDM3045X tafelmultimeter



Harry Baggen (Nederland)

Elke elektronicus heeft wel een of meer multimeters. Maar hebt u er wel eens over gedacht om in plaats van een handmultimeter een tafelmodel aan te schaffen? Die biedt meestal veel meer features en een beter display. De Siglent SDM3045X is een 4½-digit tafelmultimeter met een basisnauwkeurigheid van 0,02% en veel aansluitmogelijkheden. Hier volgt onze indruk na een testperiode van enkele maanden.

Iedereen die wat met elektronica doet, heeft wel een of meer multimeters. Je hebt er al een voor 10 euro, maar voor meer degelijkheid, meer veiligheid, meer mogelijkheden en een grotere precisie loopt dat bedrag al snel op tot enkele honderden euro's. Vooral de nauwkeurigheid is iets dat behoorlijk in de prijs tot uiting komt.

Ik heb thuis altijd met een eenvoudige multimeter gewerkt. Vorig jaar is die na decennia trouwe dienst eindelijk vervangen door een beter exemplaar met een dubbel display en vooral een grotere nauwkeurigheid dan de oude. Dat beviel erg goed. Toen ik de kans kreeg om een tafelmodel multimeter van Siglent uit te proberen, leek het me goed om die eens te vergelijken met mijn handmultimeter. De Siglent is weliswaar duurder in de aanschaf, maar misschien is hij die meerprijs wel waard. Welke meter is handiger in gebruik en wat zijn de voor- en nadelen van elk type?

Het apparaat

Dit is weliswaar de 'kleinste' multimeter van Siglent uit een reeks van drie, maar afgezien van een kleiner aantal digits en een lagere basisnauwkeurigheid biedt deze SDM3045X alles wat de duurdere versies ook hebben. Hebt u dus voldoende aan deze precisie, dan krijgt u hiermee de beste prijs/kwaliteitsverhouding.

De SDM3045X is voorzien van een stevige metalen behuizing met op de hoeken kunststof beschermkapjes. De draagbeugel kan in verschillende posities worden geplaatst, zodat de voorzijde van de meter onder een hoek omhoog staat. De kast heeft dezelfde afmetingen als veel andere apparatuur van Siglent (zoals de generatoren), zodat de apparaten gemakkelijk op elkaar kunnen worden gestapeld.

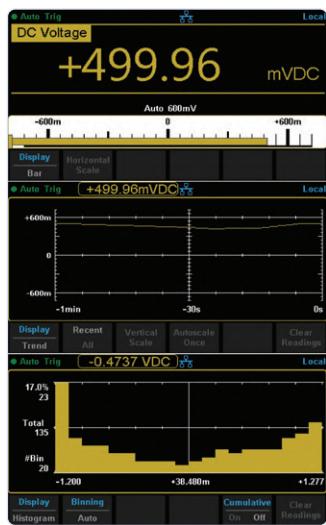
Aan de voorzijde zit een duidelijk 4,3" kleuren-display. Daaronder bevinden zich zes softmenu-toetsen waarvan de functies op het display verschijnen. Rechts zitten de gewone functietoetsen (de meeste hebben een dubbele functie) plus een set cursortoetsen voor het manoeuvreren door menu's en het instellen van waarden. Er zijn vijf aansluitbussen (**figuur 1**), twee voor de gewone metingen, twee sense-ingangen voor vierpunts weerstandsmetingen en een stroommeetingang. Aan de achterzijde vinden we een euro-net-aansluiting, een USB- en een LAN-bus voor de verbinding met een computer of netwerk, en twee BNC-bussen voor externe triggering. Verder is er ook een zekeringhouder voor het stroommeetgedeelte.

Meetmogelijkheden

De SDM3045X heeft natuurlijk alle functies die je ook bij een goede handmultimeter aantreft: spannings- en stroombereiken,



Figuur 1. Er zijn ingangsbussen aanwezig voor vierdraads weerstandsmetingen.



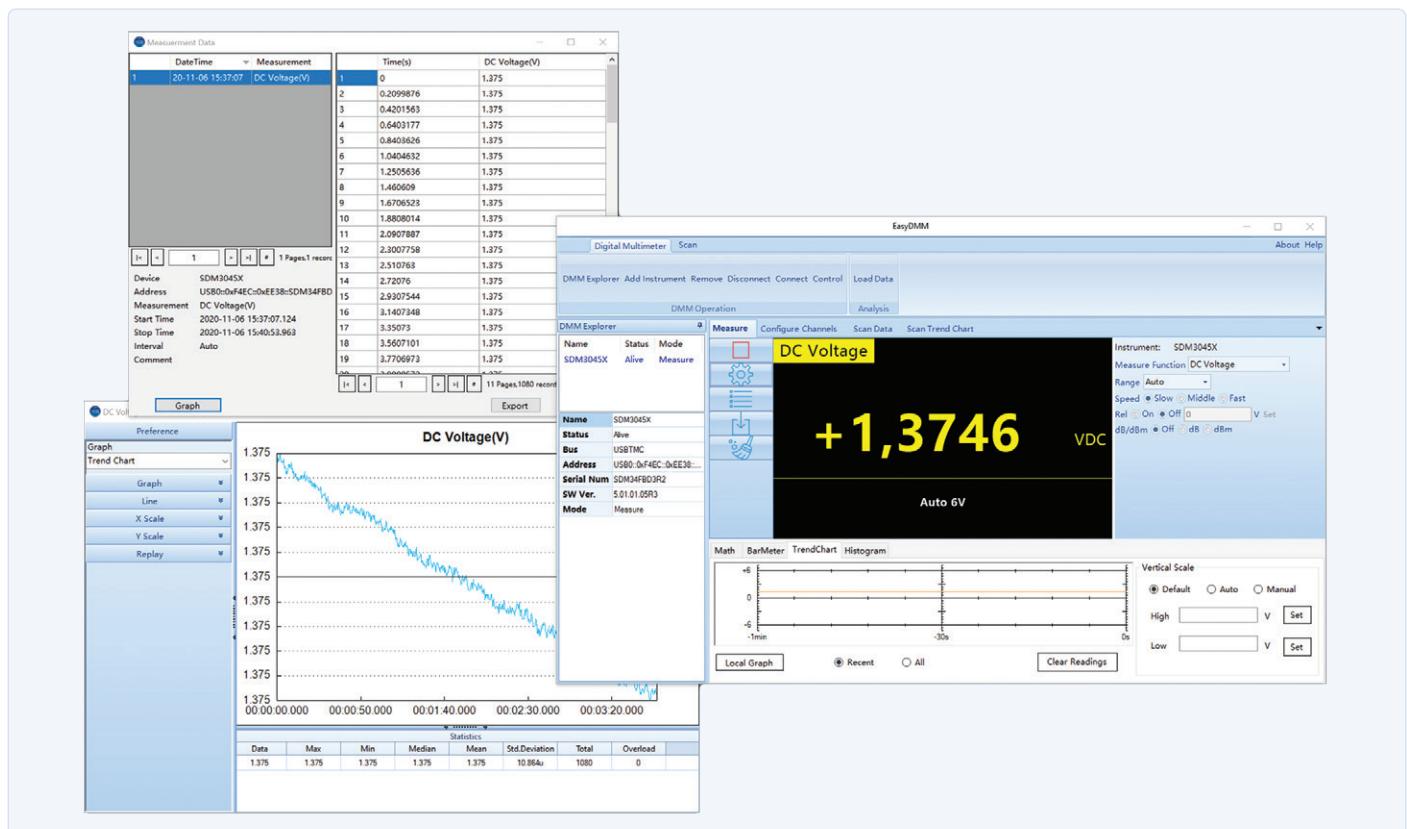
Figuur 2. Er is keus uit verschillende weergavemogelijkheden.



Figuur 3. De meter kan via USB of LAN met een PC worden verbonden.

weerstands-, capaciteits-, frequentie-, dB(m)-, temperatuur- en diode/doorgangsметing, autoranging en – zoals dat tegenwoordig bij de betere multimeters gebruikelijk is – de mogelijkheid om twee meetwaarden tegelijkertijd te tonen (bijvoorbeeld wisselspanning en frequentie). De meter heeft een display dat maximaal

66000 counts weergeeft bij een basisnauwkeurigheid van 0,02% (gelijkspanning). Hij kan maximaal 150 metingen/s verrichten. Het apparaat is niet ontworpen voor het meten aan hoogspanningsinstallaties (CAT I (10000 V)/CAT II (300 V)), maar daar gebruik je zo'n tafelmeter ook niet voor in een elektronicalab. Via LAN of



Figuur 4. De gratis software EasyDMM voor het bedienen van de meter en het inlezen van meetdata via de PC.

USB kan de meter via de PC worden bediend of kunnen meetgegevens worden opgeslagen.

Bij spanningsmetingen kun je kiezen uit verschillende weergavemogelijkheden, zoals een extra balkweergave onder de waarde, een trendgrafiek of een histogram (**figuur 2**). Er zijn ook verschillende mathematische functies. Bij veel meetfuncties zijn er nog allerlei instelmogelijkheden. Het gaat te ver om hier alles op te noemen wat je met deze meter kunt doen, maar zo kun je bijvoorbeeld in het 600-mV-bereik kiezen tussen een ingangsimpedantie van 10 MΩ en 10 GΩ, bij het testen van diodes kun je de testspanning instellen tussen 0 en 4 V, en bij de temperatuurmeting kun je kiezen uit verschillende configuratiebestanden, afhankelijk van het gebruikte type temperatuursensor. Het is mogelijk om de meter handmatig of extern te triggeren om een bepaald aantal metingen te laten uitvoeren. Verder kunnen een hoge en lage drempel worden gespecificeerd om te testen of een waarde daartussen ligt. In het interne flash-geheugen van 1 GB kan de meter tot 10.000 meetwaarden opslaan.

Via een USB-connector aan de voorzijde kunnen instellingen of meetwaarden op een USB-stick worden opgeslagen. Wie de meter met een PC wil verbinden, heeft de keus uit een USB- en LAN-verbinding (**figuur 3**). De bijbehorende software EasyDMM (**figuur 4**) is alleen geschikt voor Windows, maar dat zal voor het gros van de elektronici geen probleem zijn. Zowel het bedienen van de meter als het opslaan van meetreeksen is met deze software mogelijk. Net zoals de meeste andere Siglent-meetapparaten maakt de meter gebruik van standaard SCPI-commando's, handig voor degenen die hun eigen testprocedures willen samenstellen.

In de praktijk

Het grootste verschil tussen een tafelmultimeter en een handmultimeter is waarschijnlijk het feit dat de eerste netgevoed is. Dat lijkt misschien een nadeel, maar als je de meter altijd op dezelfde plek gebruikt merk je daar nauwelijks wat van. Je geeft de SDM3045X een plaats op je labtafel, sluit hem aan en dan blijft hij daar gewoonlijk ook staan bij de 'scoop', de voeding en andere meetapparatuur. Ik heb eens bijgehouden hoe vaak ik mijn handmultimeter meeneem naar een andere plek, maar dat gebeurt zelden. Meestal zit ik aan de labtafel te werken en daar gebruik ik ook de multimeter. Sinds de SDM3045X op mijn labtafel staat (**figuur 5**), maak ik nog maar weinig gebruik van mijn handmultimeter (ik gebruik beide als ik twee waarden tegelijk wil meten). De 3045X staat een stuk stabieler dan een hand-DMM met zijn uitklapsteuntje en ik vind de bediening met de druktoetsen een stuk prettiger dan met een draaischakelaar.

In het begin is het wel even wennen aan de bediening en alle functies, vooral omdat er zoveel instelmogelijkheden zijn. En je moet bij het inschakelen even geduld hebben totdat de 'computer' is opgestart, dat is duidelijk anders dan bij een handmultimeter. Maar dan heb je verder ook alleen maar voordelen: een groot en helder display en ongeveer alle features die je kunt bedenken voor een multimeter. De precisie is uitstekend en bij wisselspanningsmetingen is het grote frequentiebereik tot 100 kHz een pluspunt, de meeste hand-multimeters halen niet eens het audiobereik.

Wie nog meer nauwkeurigheid wil hebben, die kan uitwijken naar het duurdere broertje SDM3055 dat een digit extra biedt. Het grote voordeel van de SDM3045X is echter dat hij in tegenstelling tot de 3055 geen ventilator heeft en in gebruik dus muisstil is. De bijbehorende PC-software is overzichtelijk en ideaal voor het loggen van

meetdata. U merkt het al: ik heb mijn handmultimeter nauwelijks nog uit zijn etui gehaald, dit apparaat biedt zoveel meer!

Conclusie

Wie bereid is wat meer geld uit te geven voor een goede multimeter, die kan ik alleen maar adviseren om eens te kijken naar de SDM3045X en alle mogelijkheden die in de datasheet beschreven worden. Zowel op een professionele als hobby-werkplek is deze tafel-multimeter een uitstekende metgezel die heel veel meer kan dan een hand-multimeter en naar mijn ervaring ook beter te bedienen en af te lezen is. ◀

200720-02



Figuur 5. De SDM3045X voelt zich uitstekend thuis tussen andere labapparatuur.

Vragen of opmerkingen?

Hebt u vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur of naar de redactie van Elektor, via redactie@elektor.com.

Een bijdrage van

Tekst en illustraties:
Harry Baggen

Redactie: **Eric Bogers**
Layout: **Giel Dols**



GERELATEERDE PRODUCTEN

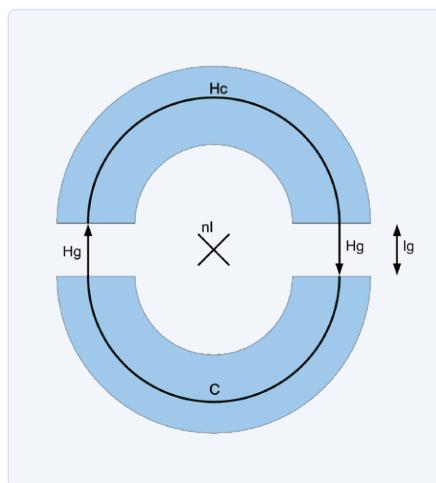
► **Siglent SDM3045X Multimeter**
www.elektor.nl/siglent-sdm3045x-multimeter

DC-stroomklem

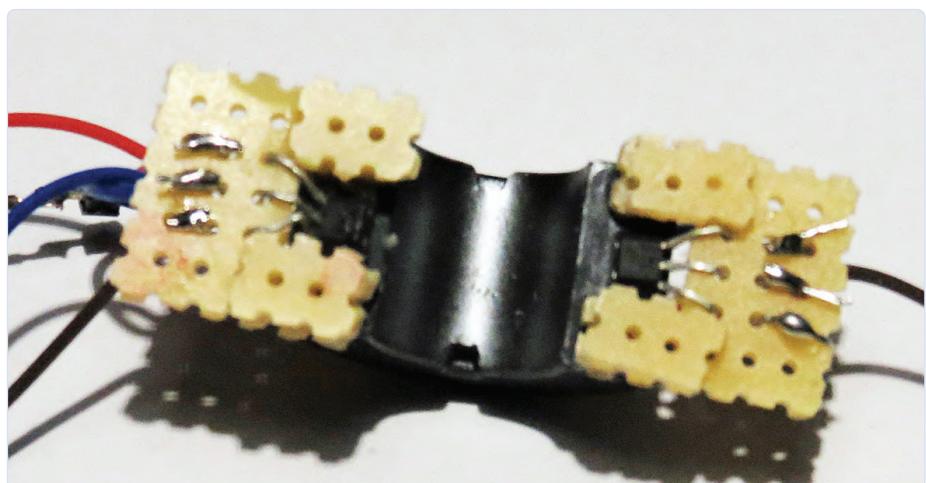
Hall-sensor + ferrietkern + Arduino

Prof. dr. Martin Oßmann (Duitsland)

Wanneer u de stroom door een draad moet meten, zet u normaliter een ampèremeter in serie met de kabel. Als u een zogenaamde stroomtang gebruikt, hoeft u het circuit niet te onderbreken. De eenvoudiger uitvoeringen van zo'n stroomtang kunnen alleen wisselstroom meten omdat ze sensoren gebruiken die alleen op AC reageren (die werken met andere woorden volgens hetzelfde principe als een transformator). Om gelijkstromen te meten, kunnen we Hall-sensoren gebruiken.



Figuur 1. De geometrie van het magnetische veld.



Figuur 2. Montage van de Hall-sensoren op de kern.

Voordat we bij een project als dit aan het ontwerp beginnen, is het een goed idee om de materie te onderzoeken en commerciële voorbeelden te bestuderen om te zien hoe die zijn ontworpen en waar verbetering mogelijk is. Een stroomtang bestaat in feite uit twee halve ferrietkernen. Die kern is zo opgebouwd dat de twee helften gescheiden zijn door een luchtspleet (**figuur 1**). De stroom die door de door de kern omsloten draad vloeit, heeft de waarde nl .

Wiskundige relaties

De waarde n staat voor het aantal windingen (waarmee de geleider die de te meten stroom voert, om de detectiekern wordt gewikkeld); het is een vermenigvuldigingsfactor die wordt toegepast op de meting van het magnetische veld dat de kern magnetiseert. De magneti-

sche fluxdichtheid (magnetische inductie) in de kern B_c (waarbij c voor *core* of kern staat) is hetzelfde als in de luchtspleet B_g (waarbij g voor *gap* of spleet staat) zodat $B_g = B_c$. Dit volgt uit de vergelijkingen van Maxwell. Eenvoudigheidshalve kunnen we aannemen dat de magnetische velden in de kern en de luchtspleet homogeen zijn. De materiaalvergelijking $B_g = \mu_0 \times H_g$ is van toepassing in het gebied van de luchtspleet, terwijl in de kern geldt $B_c = \mu_r \times \mu_0 \times H_c$. Ferrietmaterialen hebben een typische relatieve permeabiliteit μ_r van ongeveer 2.000. Hieruit volgt $H_g = \mu_r \times H_c$, dus in de kern is de magnetische veldsterkte een factor μ_r kleiner dan in de luchtspleet. In de praktijk mogen we daarom de magnetische veldsterkte in de kern vaak verwaarlozen. De wet van Ampère beschrijft de relatie tussen het geïntegreerde magnetische veld rond een geslotenlus en de stroom door die lus. De

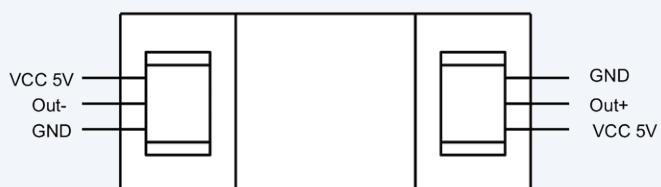
lijnintegraal van het magnetische H -veld rond een gesloten kromme is gelijk aan de totale stroom nl . In figuur 1 is deze kromme aangegeven als C . De luchtspleet heeft hier de lengte $l_g = 1,6$ mm; de lengte van een kernhelft wordt gegeven door $l_c = \pi \times d_c / 2$ waarbij de kerndiameter $d_c = 18$ mm bedraagt. Volgens de wet van Ampère geldt:

$$2 \times l_g \times H_g + 2 \times l_c \times H_c = nl$$

Als we hier invullen $H_c = H_g / \mu_r$ en dan H_g oplossen, krijgen we:

$$H_g = nl / (2 \times l_g + 2 \times l_c / \mu_r)$$

Hiermee kunnen we het magnetische veld dat door de sensoren wordt gedetecteerd, berekenen als functie van de stroom. In dit project worden twee stuks A 1324



Figuur 3. Aansluitingen van de Hall-sensoren.



Figuur 4. Als de draad meerdere malen om de kern wordt gewikkeld, neemt de gevoeligheid toe.

LUA-T Hall-sensoren met een gevoelighed $S = 5 \text{ mV} / \text{G}$ gebruikt. Dit resulteert in een uitgangsspanning van de sensoren die gelijk is aan:

$$U = 2 \times S \times \mu_0 \times nl / (2 \times I_g + 2 \times I_c / \mu_r)$$

Door waarden in de vergelijking in te vullen, krijgen we:

$$U = I \times 38,92 \text{ mV} / \text{A}$$

Als we kijken naar de stromen die doorgaans in 'doorsnee' elektronische applicaties voorkomen, hebben we te maken met niet al te extreme spanningen in het mV-bereik. Als we het veld in de ferrietkern negeren, kunnen we de uitdrukking vereenvoudigen tot:

$$U = S \times \mu_0 \times nl / I_g$$

hetgeen $U = I \times 39,26 \text{ mV} / \text{A}$ geeft, een vereenvoudigde schaalfactor die dicht genoeg bij de hierboven afgeleide waarde ligt.

De ferrietkern

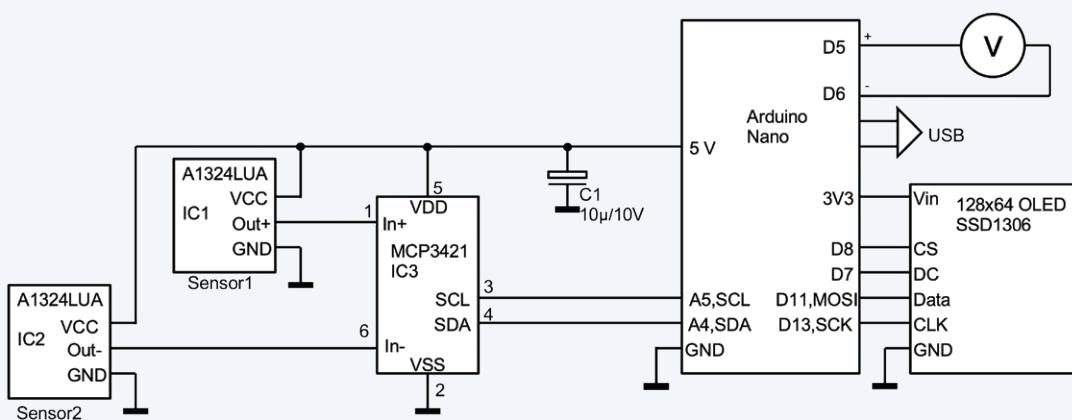
Het kernelement van de DC-stroomklem bestaat uit de twee helften van een standaard scharnierend ontstorings-ferrietkern die vaak om kabels zijn geklemd die uit een behuizing met elektronische apparatuur komen (**figuur 2**). De twee Hall-sensoren zijn aan de platte uiteinden van de ene helft van de kern gelijmd en vormen zo de luchtspleten. Een klein stukje van 1,6 mm dik pertinax of gaatjesprint (waarvan al het koper is verwijderd) draagt de sensor en bepaalt de dikte van de luchtspleet. In **figuur 3** ziet u hoe de sensoren worden geplaatst.

De twee kernhelften kunnen vervolgens met bijvoorbeeld plakband of een elastiekje tegen elkaar worden geklemd. De stroomvoerende geleider kan dan door het gat in het midden van de constructie worden geleid. **Figuur 4** toont een voorbeeld waarbij een eenaderige geleider meerdere keren rond de kern is gelust, met als effect dat de gevoelighed van de Hall-effectsensor lineair toeneemt.

De elektronica

Een MCP3421 wordt gebruikt als analogoog/digitaal-omzetter. Dit IC is een goedkope 18-bit converter met een interne spanningsreferentie van 2,048 V en een differentiële ingangstrap. De uitgangsspanning van de Hall-sensor bedraagt in rust de halve voedingsspanning. Wanneer ze zoals weergegeven in de luchtspleet tussen de kernhelften zijn geplaatst, produceert de magnetische flux ten gevolge van de stroom in de stroomvoerende kabel een positieve spanningsverandering aan de uitgang van de ene sensor en een corresponderende negatieve verandering bij de andere sensor. Op deze manier verbonden, worden de veranderingen van hun uitgangsspanning effectief opgeteld in de differentiële ingangstrap van de A/D-omzetter. De resolutie van de 18-bit converter bedraagt $2,048 \text{ V} / 2^{17} = 0,015 \text{ mV}$, genoeg voor deze toepassing.

Als CPU wordt een Arduino Nano-microcontroller gebruikt; deze heeft meer dan genoeg GPIO-mogelijkheden voor



Figuur 5. De complete schakeling van de stroomklem.

ons doel. Een goedkoop OLED-display van 128 x 64 pixels dient als scherm. De relatief eenvoudige schakeling is in opgebouwde vorm te zien in **figuur 5**. Mocht u liever een analog instrument gebruiken, dan kunt u aan de uitgang ook een moving-coil voltmeter aansluiten. Eén volt uitgangsspanning komt dan overeen met één ampère meetstroom.

De commando's

Om de schaalparameters in de software van de auteur te gebruiken en in te stellen, kunnen een paar commando's worden ingevoerd via de seriële interface van de Arduino IDE. De gemeten waarden worden ook weergegeven via die seriële interface.

Commando '0' = nulinstelling

Dit commando mag alleen worden gegeven als er geen meetstroom door de meetkern van de stroomklem loopt. De software voert een nulcorrectie uit en slaat de waarde van deze offset op in niet-vluchttig EEPROM. De Hall-sensoren hebben een offsetspanning (die temperatuurafhankelijk is) en die moet worden gecompenseerd om nauwkeurig te kunnen meten.

Commando '1' = schaalfactor voor 1 A

Deze kalibratiestap wordt gebruikt om de interne schaalfactor te bepalen bij een stroom $I_1 = 1 \text{ A}$ door de te meten kabel stroomt. De schaalfactor wordt berekend en vervolgens opgeslagen in EEPROM.

Commando '5' = schaalfactor voor 500 mA

Zoals het commando '1' maar dan voor een referentiestroom van 0,5 A.

Commando 'u'

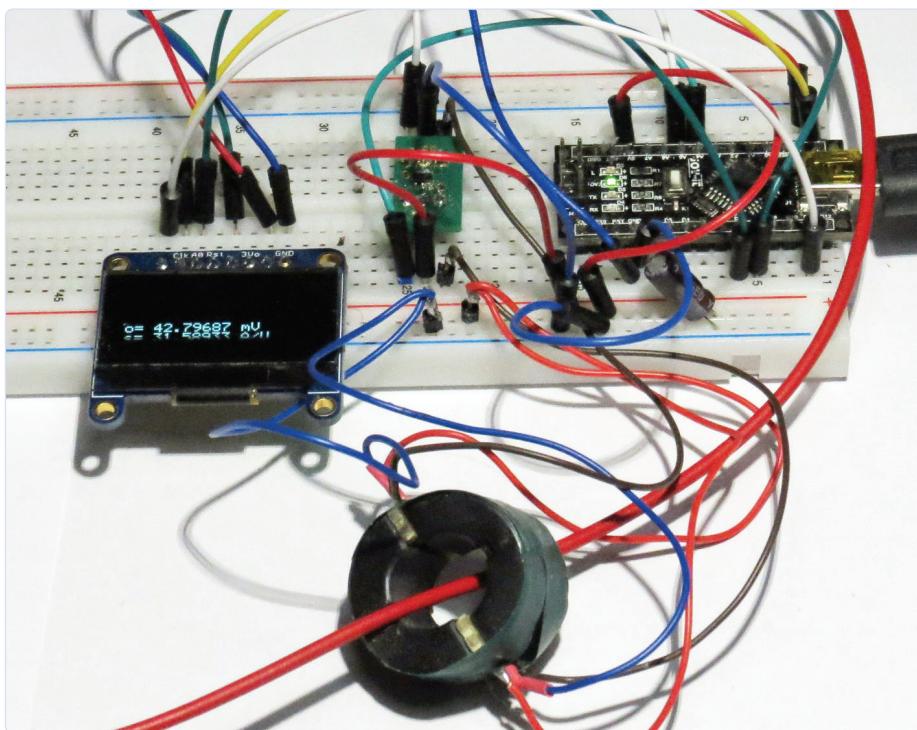
Als na deze 'u' een waarde wordt ingevoerd, wordt deze waarde weergegeven als een spanning. De stroomklem gaat vervolgens gedurende een zekere periode naar de 'DVM-kalibratiemodus'; dan kunt u de '+' en '-' gebruiken om de uitvoer op de exacte waarde in te stellen. ▶

Commando 'd'

De standaardwaarden van de parameters worden naar EEPROM geschreven om deze voor het eerste gebruik te initialiseren.

Bouw

Om het principe van deze schakeling te demonstreren, heb ik het prototype opgebouwd op breadboard en de diverse componenten in de schakeling met jumperdraden aangesloten (**figuur 6**). Op de voorgrond ziet u de ferrietkern; het kleine



Figuur 6. Het op breadboard gebouwde prototype.

display is links te zien en de Arduino Nano aan de rechterkant. De software is ontwikkeld met behulp van de Arduino IDE en vervolgens gedownload naar het Nano-board. De nieuwste versie van deze software is gratis beschikbaar op de Elektor-projectpagina bij dit artikel [1]. Het prototype van het stroomklem-project is getest; het gedroeg zich zoals verwacht na het opstarten. We hopen dat deze schakeling u zal inspireren tot eigen experimenten of om andere projecten te bouwen waar stromen magnetisch worden gemeten! ▶

200595-03

Vragen of opmerkingen?

Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur of naar de redactie van Elektor, via redactie@elektor.com.

Een bijdrage van

Ontwerp en tekst:

prof. dr. Martin Oßmann

Redactie: **dr. Thomas Scherer**

Vertaling: **Eric Bogers**

Layout: **Giel Dols**

WEB LINK

[1] Projectpagina bij dit artikel: www.elektormagazine.nl/200595-03



GERELATEERDE PRODUCTEN

► **0.96" 128x64 OLED-display voor Arduino**

www.elektor.nl/blue-0-96-oled-display-spi-6-pin

► **PeakTech Clamp meter 4350**

www.elektor.nl/peaktech-4350-clamp-meter

Op zoek naar een nieuwe baan met veel #spanning en een #stroom aan werk met #frequente uitdagingen? #Wissel dan als de #bliksem je cv uit met ons! #Ohm te solliciteren stuur je een mailtje naar jointheteam.benelux@trescal.com.

CALIBRATION ENGINEERS

Om onze groei te ondersteunen zijn we op zoek naar Calibration Engineers in België en Nederland! Als sollicitant heb je bij voorkeur een opleiding (Bachelor / HBO) elektronica afgerond en heb je ervaring met elektrische meetapparatuur. Geen formele opleiding afgerond, maar wel een bulk aan praktische ervaring? Ook dan ben jij de persoon die we zoeken!

Onze Calibration Engineers vormen de basis van onze activiteiten. Als (toekomstig) expert op vlak van kalibratie en metrologie krijg je regelmatig opleiding van ons zodat je voorloper blijft in het vakgebied. Je kalibreert en controleert elektrische meetinstrumenten zoals frequentie analyzers, oscilloscopen, digitale multimeters, spectrum analyzers, RF-generatoren, RF-powermeters, antennes, EMI ontvangers, ... Deze instrumenten worden door onze klanten gebruikt om hun bedrijfsprocessen (research, productie, ...) correct en nauwkeurig te laten verlopen. Deze kalibraties kunnen zowel in één van onze laboratoria, als ter plaatse bij de klant worden uitgevoerd.

OVER TRES CAL

Trescal is uitgegroeid tot een internationale leider in de kalibratiewereld. Wij bieden alle industrieën de nodige kalibratiediensten voor hun meetinstrumenten en dit in een ongeëvenaard aantal domeinen. Ons serviceaanbod omvat kalibratie in de domeinen elektrisch, druk, temperatuur, CO₂, relatieve vochtigheid, dimensioneel en mechanisch. Daarnaast voeren we ook 3D metingen en validaties uit. Onze activiteiten spitsen zich toe in sectoren als: Elektronica, Telecom, Luchtvaart en Defensie, Automotive en Transport, Energie, Chemie en Petrochemie, Medisch en Farmaceutisch. Met onze meer dan 150 laboratoria, verspreid over 26 landen kalibreren onze 3850 medewerkers dagelijks de meetinstrumenten van onze klanten met hoogwaardige standaarden. In de Benelux hebben we locaties in Zoetermeer (NL), Hengelo (NL), Antwerpen (BE), Louvain-la-Neuve (BE) en Wellin (BE). We zijn ISO 9001, VCA en ISO 17025 geaccrediteerd.

WAAROM TRESCAL?

We besteden veel aandacht aan de opleiding en ontwikkeling van onze medewerkers om ze te blijven uitdagen en te interesseren in hun werk. We doen dit ook om voorloper te blijven binnen de wereld van kalibratie en metrologie, en hierdoor onze klanten steeds de best mogelijke service te kunnen bieden.

Daarbij investeren we in de automatisatie van een aantal kalibratieprocessen om onze levertermijnen aan klanten te verkorten en de foutenmarge tot een minimum te beperken. Zo hebben we intern verschillende robots en software ontwikkeld voor de kalibratie van handheld multimeters, massa's (van 1mg tot 50 kg), temperatuursondes, ...

We hebben software ontwikkeld om de administratieve taken tot een minimum te beperken zodat onze Calibration Engineers zoveel mogelijk tijd kunnen besteden aan wat ze goed kunnen én graag doen: met techniek bezig zijn!



Hebben we je interesse gewekt? Bezoek dan zeker onze website www.trescal.com voor meer informatie en een actueel overzicht van onze vacatures en/of stuur je cv naar jointheteam.benelux@trescal.com.

Soldeerstation 2021

makkelijk om te bouwen!

Luc Lemmens en Mathias Claussen (Elektor)

Hoewel er volop kant-en-klare soldeerstations te koop zijn, zijn er nog steeds mensen die deze apparatuur het liefst zelf bouwen. Het zelfbouw-soldeerstation voor Weller RT-soldeerbouten dat we in Elektor januari/februari 2019 [1] presenteerden, bleek erg populair te zijn bij onze lezers. Naar aanleiding van de feedback die we op dit project kregen, hebben we besloten om een nieuw ontwerp voor de Weller RT te maken, dat ook Hakko FX-8801 en JBC T245 solderbouten ondersteunt.



PROJECT-INFO

Tags

gereedschap, lab, solderen, Weller, Hakko, JBC

Niveau

beginners – gevorderden – experts

Tijd

ongeveer 4 uur

Gereedschap

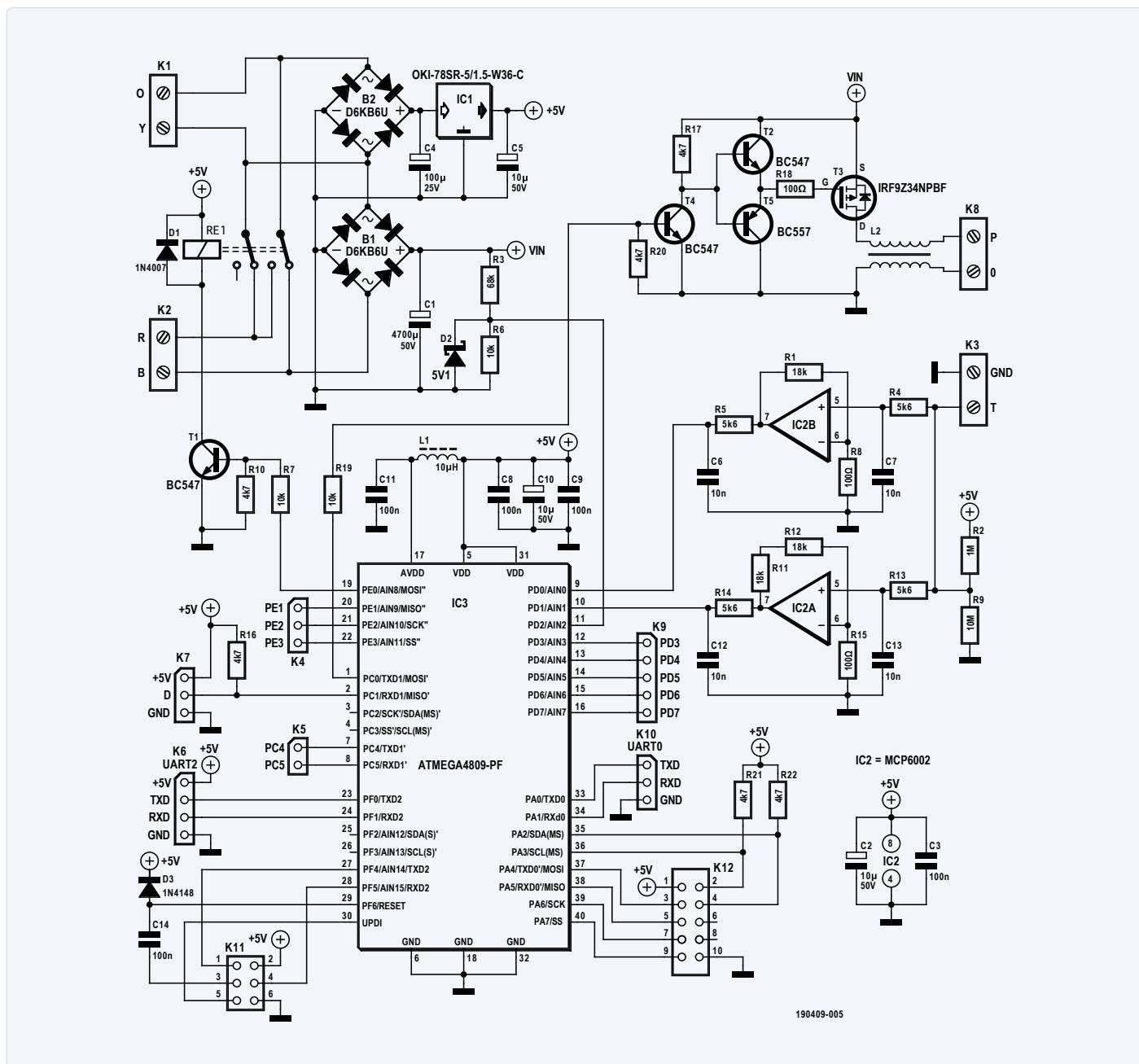
solder- en mechanisch gereedschap,
3D-printer (optioneel)

Kosten

ongeveer € 70

Het eerste en zeer belangrijke ontwerp-criterium was om de totale kosten zo laag mogelijk te houden. Niets speciaals, maar gewoon een goed werkend, robuust ontwerp. Het ziet er misschien wat ouderwets uit in het huidige SMD-tijdperk, maar als het gaat om het solderen van hardware geven de meeste hobbyisten nog steeds de voorkeur aan through-hole onderdelen. Hoewel het steeds moeilijker wordt om deze (vooral IC's) te vinden, wilden we een compleet through-hole ontwerp maken om dit nieuwe project nabouwvriendelijk te maken. Om de bouw en

bedrading van dit soldeerstation te vereenvoudigen, bestaat het uit twee printplaten (hoofdprint en displayprint) die met elkaar zijn verbonden met één flatcable. Er wordt een nieuwe AVR-microcontroller van Microchip Technology gebruikt, die het mogelijk maakt om de software voor de Arduino IDE te ontwerpen, zodat lezers die de software willen veranderen dit eenvoudige ontwikkelplatform kunnen gebruiken. We waren best trots op en tevreden met het compacte ontwerp van het vorige soldeerstation met zijn externe standaard 12V_{DC}-voeding.



Figuur 1. Schema van de hoofdprint.

Veel van onze lezers klaagden echter dat het veel te klein en niet zwaar genoeg was om tijdens het solderen op zijn plaats op de werkbank te blijven. En we moeten toegeven: het is misschien goed voor draagbare toepassingen, maar het is verre van ideaal voor een desktop-soldeerstation. Het integreren van de voeding in het nieuwe ontwerp biedt een gemakkelijkere manier om dit probleem op te lossen: een 60VA-ringkerntrafo voegt aardig wat gewicht toe in de behuizing. Een groter LED-display verbetert de leesbaarheid van de temperatuurstelling, in vergelijking met

het kleinere OLED-display van het eerdere station. Het toont minder informatie, maar de belangrijkste waarde (de temperatuur) is nu in één oogopslag te zien.

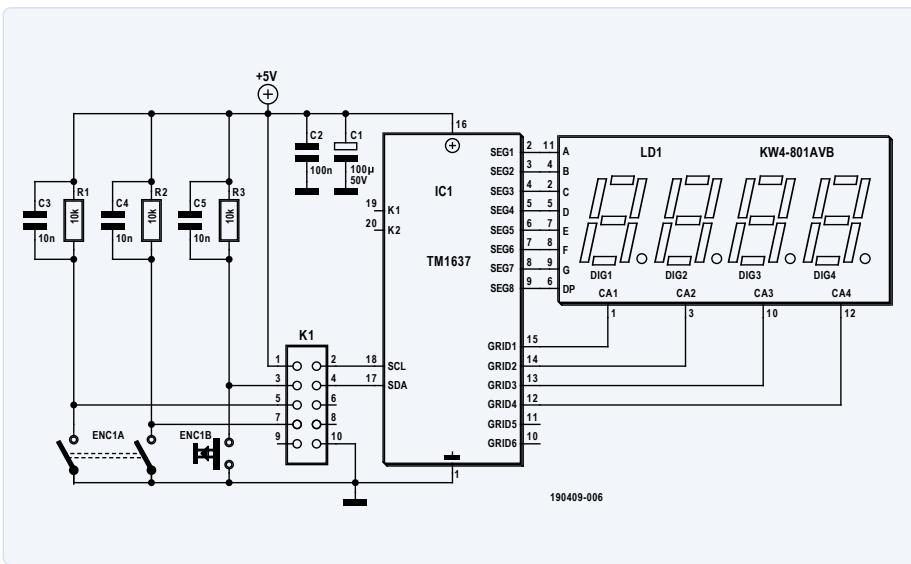
Als het niet nodig is...

...verander dan niets. Vanuit hardware-oogpunt werkte het vorige ontwerp prima, en in dat opzicht hebben we maar weinig veranderd (**figuur 1**). De meet- en regelcircuits voor de temperatuur lijken wat dat betreft sterk op elkaar. Tijdens de prototype-fase hadden we echter al onze twijfels over het nut en de

noodzaak van de stroommeting met shuntweerstand en INA138 sense-versterker. Het maakte de hardware en software om de temperatuur te regelen onnodig ingewikkeld, dus we lieten dit deel van de schakeling helemaal weg.

Voeding

Het nieuwe soldeerstation wordt gevoed door één grote ringkerntransformator (2 x 12 V, 60 VA); de twee secundaire wikkelingen gaan naar K1 en K2. Het voorgeschreven type in de onderdelenlijst heeft ook twee primaire



Figuur 2. Schema van de displayprint.

wikkelingen, waardoor de voeding voor zowel 115 V_{AC} (primaire wikkelingen parallel) als voor 230 V_{AC} (wikkelingen in serie) geschikt is. U kunt natuurlijk ook een transformator kopen met dezelfde 'secundaire' specificaties, maar waarvan de primaire bij uw eigen netspanning past. Een van de beide secundaire wikkelingen wordt gebruikt voor de +5V-voeding van de logische en regelschakelingen, via bruggelijkrichter B2 en DC/DC-converter IC1. Afhankelijk van de aangesloten soldeerbout worden de secundaire wikkelingen parallel of in serie geschakeld door RE1; als het relais is uitgeschakeld is de lagere spanning geselecteerd. De wisselspanning wordt gelijkgericht en afgevlakt (B1 en C1) en via spanningsdeler R3 en R6 geschaald naar het ingangs bereik van de interne A/D-converter van de microcontroller. Zenerdiode D2 dient als overspanningsbeveiliging.

Het vermogen dat aan het soldeerbout wordt geleverd – en dus de temperatuur van de punt – is PWM-gestuurd, met dezelfde schakeling (T2...T5) als we gebruikten in het eerdere DIY-soldeerstation [1]. Common mode-smoerspoel L2 onderdrukt RF-interferentie op de kabel van de soldeerbout.

Meting van de temperatuur

De opampschakelingen rond IC2A en IC2B versterken de spanning op de temperatuursensor-aansluiting respectievelijk 361 maal (C-type voor Hakko) en 181 maal (K-type voor Weller en JBC). We gaven de voorkeur aan twee afzonderlijke vaste versterkers in plaats van het omschakelen van weerstanden om de versterking aan te passen. Het type

soldeerbout wordt softwarematig ingesteld en de microcontroller selecteert de bijbehorende voedingsspanning en A/D-converteerringang om de temperatuur van de punt te meten. Een DS18B20 1-Wire temperatuursensor kan worden aangesloten op K7 om de omgevingstemperatuur te meten voor koude las-compensatie van het thermokoppel in de soldeerbout.

Programmeren en debuggen

Gelukkig begrijpen en weten fabrikanten van microcontrollers zoals Microchip/Atmel dat er nog steeds een markt is voor through-hole onderdelen, dus zelfs de recente ATmega4809 met Microchip's 'nieuwe' Unified Program and Debug Interface (UPDI) is beschikbaar in een 40-pins DIP-versie, ideaal voor prototyping en zelfbouwprojecten. K11 biedt een UPDI voor deze microcontroller die – zoals de naam al doet vermoeden – wordt gebruikt voor zowel programmeren als debuggen. Meer informatie over deze opvolger van de AVRISP (die ons al zoveel jaren van dienst is!) is te vinden op Elektor Labs [2]. Deze pagina laat ook zien hoe u een Arduino Uno-board kunt gebruiken als programmeerinterface voor dit project. Houd er rekening mee dat de redelijk betaalbare Microchip Snap UPDI-programmeer-/debuginterface (nog) niet wordt ondersteund in de Arduino IDE.

Boxheader K12 vormt de verbinding met het frontpaneel (display en draai-encoder), dat later wordt besproken. De pin die in de componentenopdruk met 'spare' is aangeduid, wordt niet gebruikt in de huidige versie van de hard- en software, maar kan in latere ontwik-

kelingen worden gebruikt voor een extra in- of output op de displayprint.

Vrijwel alle ongebruikte pinnen van de microcontroller worden naar SIL-connectoren op de print gevoerd. Twee UARTS's zijn te vinden op respectievelijk K6 en K10 en andere (analoge en digitale) I/O's op K4, K5 en K9, maar geen van deze heeft een speciale functie of wordt ondersteund in de huidige versie van de firmware.

Displayprint PCB 190409-2

Op zoek naar betaalbare displays, kwamen we veel kant-en-klare 4-cijferige I²C-gestuurde modules tegen, vaak met een TM1637 LED-displaydriver van Titan Microelectronics. Deze displays worden veel gebruikt door hobbyisten en kunnen ook in ons soldeerstation worden ingezet. Voor een soldeerstation zouden drie cijfers voldoende zijn geweest om de temperatuur weer te geven, maar de viercijferige versies bleken goedkoper te zijn. Omdat we ook een draai-encoder met drukknop op het frontpaneel hebben en we de bedrading tussen hoofd- en frontprint zo eenvoudig en netjes mogelijk wilden houden, hebben we besloten om onze eigen print te maken met al deze componenten erop en slechts één flatcable om de twee printen te verbinden. Om die reden hebben we een eigen ontwerp gemaakt met de TM1637 en een viercijferig 7-segment LED-display (**figuur 2**). De TM1637 is zowel in DIP- als in SOIC-uitvoering verkrijgbaar en hoewel we het soldeerstation uitsluitend met through-hole onderdelen wilden bouwen, maakten we een uitzondering voor dit IC door footprints voor beide packages op de print aan te brengen. De belangrijkste reden hiervoor was dat de verzending van de DIP-IC's uit Azië vertraging opliep door COVID-19, en dat we een paar complete modules in het lab hadden liggen. Het uitsolderen van een SOIC van een van deze modules was de snelste manier om ons prototype verder te ontwikkelen.

Software

De code zelf bestaat uit verschillende onderdelen die in elkaar grijpen. Deze worden geïmplementeerd als klassen en objecten die op basis daarvan zijn gemaakt. Wie met het Arduino-framework werkt, gebruikt objecten en klassen, vaak zonder het te weten. Heel algemeen gesteld staat een klasse voor een bouwbeschrijving waarmee in de code een passend object wordt gecreëerd, dat wil zeggen wat er ontstaat bij constructie volgens de beschrijving. Een inleiding tot de object-georiënteerd programmeren in C ++ valt ver buiten het bestek van dit artikel; daarom zullen

we de objecten en klassen hier op een zeer abstracte manier bespreken.

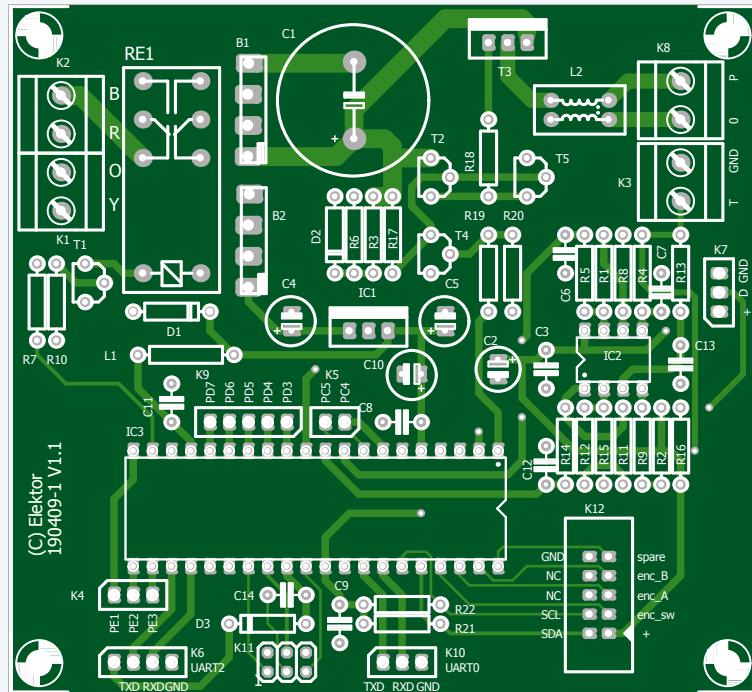
Zoals het display, in de software *frontend* genoemd. Het dient om de waarden (temperatuur, menu en foutcodes) te tonen die vanuit het soldeerstation naar het frontend worden overgedragen. Het heeft niet veel zin om aanpassingen in de code door te voeren voor elk verschillend type display (OLED, alfanumeriek LCD of 7-segment).

Hier komen de klassen en objecten om de hoek kijken: voor de kern van de software is het frontend een display dat altijd dezelfde functies biedt. Met de juiste klassen (bouwbeschrijvingen) kan dus de software voor een soldeerstation gebouwd worden voor de hier beschreven hardware. Als u (bijvoorbeeld) liever een 0,96-inch OLED gebruikt in plaats van het LED-display, hoeft alleen de code voor dit frontend te worden geïntegreerd.

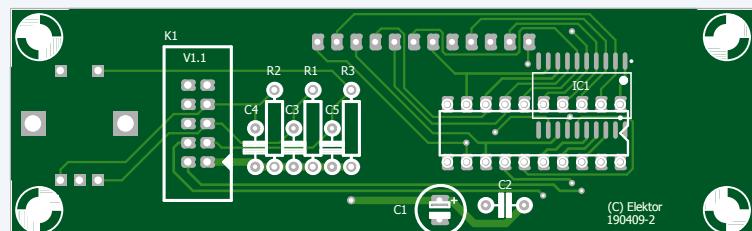
Het programma van het eerdere zelfbouw-soldeerstation (180349) is ontworpen om dit mogelijk te maken. De kern van het geheel blijft in principe hetzelfde, alleen de onderdelen eromheen worden aangepast aan de hardware.

Er zijn ook aanpassingen gemaakt voor het meten van de temperatuur. Zoals te zien in het schema zijn er twee afzonderlijke opamps gemonteerd, één voor type-K en één voor type-C thermokoppels, zodat soldeerbouten en tips van verschillende fabrikanten kunnen worden gebruikt. Bovendien is er een koude las-compensatie op basis van een DS18B20; een geschikte sensor wordt automatisch gedetecteerd en geëvalueerd. Afhankelijk van de soldeertip die is ingesteld in het soldeerstation, wordt automatisch analoge ingang A0 of A1 geselecteerd en de meetwaarde omgerekend naar de corresponderende temperatuur.

Vervolgens wordt de koude las-compensatie toegepast (voor zover geïnstalleerd). De temperatuur op dit punt is afkomstig van een 1-Wire sensor DS18B20; voor de uitlezen daarvan zijn enige trucs nodig. Na het starten van de software zoekt het systeem naar deze 1-Wire sensor; als er geen wordt gevonden, wordt dit aangegeven in de software en er wordt geen poging meer gedaan om opnieuw naar de sensor te zoeken. Als er wel een sensor wordt gevonden, vindt het uitlezen van de temperatuur in twee stappen plaats. De eerste stap wordt elke twee seconden uitgevoerd en start de registratie van de huidige temperatuur. De sensor heeft nu minimaal 750 ms conversietijd nodig voordat de waarde gereed is om te worden gelezen en verwerkt. Omdat de MCU niet gedurende 750 ms mag stoppen, wordt het begin van de conversie



Figuur 3a. Onderdelenopstelling van de hoofdprint.



Figuur 3b. Onderdelenopstelling van de displayprint.

gemarkeerd. Als dit meer dan een seconde geleden is, wordt de waarde opgehaald van de sensor en wordt aangegeven dat het uitlezen is voltooid. Na een seconde wordt de volgende leescyclus gestart. Op deze manier kan de kern andere taken uitvoeren tijdens de temperatuurconversie.

PWM met timer A

De timers in ATmega4809 verschillen een beetje van die in zijn voorgangers. In deze nieuwe microcontroller kan timer A zes PWM-uitgangen leveren, wat erg praktisch is voor het aansturen van servo's, LED's en andere componenten. Dit is ook de standaardconfiguratie van *MegaCoreX*, het hardwarepakket dat in de Arduino IDE voor de ATmega4809 moet worden geïnstalleerd. Timer A is een 16-bit timer met drie

PWM-uitgangen, maar is standaard geconfigureerd als een 8-bits timer met zes uitgangen. Voor servo's is dit voldoende, maar voor de vermogensregeling van het soldeerstation is het niet nauwkeurig genoeg bij 10 kHz. Daarom moet de timer A worden teruggezet naar de 16bit-modus voor het genereren van PWM-signalen in ons soldeerstation.

Draai-encoder met drukknop

De draai-encoder wordt verwerkt met een timer interrupt-routine. Elke 250 µs worden de pinnen van de draai-encoder ingelezen, en de draairichting wordt bepaald uit de laatste vier pin-toestanden. Omdat de draai-encoder mechanische contacten heeft, kan verdraaien onzuivere flanken op de ingangen van de ATmega opleveren en zal er altijd wat ruis aanwezig zijn als geen filtering wordt toege-



ONDERDELENLIJST HOOFDPRINT

Weerstanden:

R1,R11,R12 = 18 k
 R2 = 1 M
 R3 = 68 k
 R4,R5,R13,R14 = 5k6
 R6,R7,R19 = 10 k
 R8,R15,R18 = 100 Ω
 R9 = 10 M
 R10,R16,R17,R20,R21,R22 = 4k7

Spoelen:

L1 = smoorspoel 10 μH, 130 mA
 L2 = common mode smoorspoel 10 A,
 Laird CM2545x171B-10

Condensatoren:

C1 = 4700 μF, 50 V, steek 10 mm, 22x41 mm
 C2,C5,C10 = 10 μF, 50 V, steek 2 mm, 5x11 mm
 C3,C8,C9,C11,C14 = 100 nF, 50 V, X7R, steek
 5,08 mm
 C4 = 100 μF, 50 V, steek 3,5 mm, 8x11 mm
 C6,C7,C12,C13 = 10 nF, 50 V, steek 5 mm, X7R

Halfgeleiders:

D1 = 1N4007, 1000 V, 1A
 D2 = zenerdiode 5,1 V, 500 mW
 D3 = 1N4148, 100 V, 200 mA, 4 ns
 B1, B2 = bruggelijkrichter D6KB6U, 600V, 6A
 T1,T2,T4 = BC547C, 45 V, 100 mA, 500 mW,
 hfe = 400

T3 = IRF9Z34NPBF, P-MOSFET, 55 V, 17 A,
 100 mΩ
 T5 = BC557C, -45 V, -100 mA, 500 mW,
 hfe = 400
 IC1 = DC/DC-converter OKI-78SR-5/1.5-
 W36-C, 5V, 1,5 A
 IC2 = dual-opamp MCP6002-E/P
 IC3 = 8-bit MCU ATmega4809-PF

Diversen:

RE1 = vermogensrelais, 5 VDC, DPDT, 8 A,
 Schrack RT424005
 K1,K2,K3,K8 = printkroonsteen 5,08 mm,
 2-polig, 630 V
 K4,K7,K10 = pinheader, 1 rijig, 3-polig, verticaal
 K5 = pinheader, 1 rijig, 2-polig, verticaal
 K6 = pinheader, 1 rijig, 4-polig, verticaal
 K9 = pinheader, 1 rijig, 5-polig, verticaal
 K11 = pinheader, 2-rijig, 6-polig, verticaal
 K12 = 10-polige boxheader, steek 2,54 mm

Overig:

ringkerntransformator 60 VA, 2x115 V, 2x12 V,
 MCTA060/12
 primaire zekering 20 mm, 630 mA @ 240 VAC
 primaire zekering 20 mm, 1,25 A @ 115 VAC
 K&B 59JR101-1FR-LR IEC-connector 42R-serie
 (netentree, Conrad 736709)

10-polige IDC-stekker (2 stuks)
 10-adige flatcable, ca. 20 cm
 Print 190409-1 V1.1

DISPLAYPRINT

Weerstanden:

R1,R2,R3 = 10 k

Condensatoren:

C1 = 100 μF, 50 V, steek 3,5 mm, 8x11 mm
 C2 = 100 nF, 50 V, X7R, steek 5,08 mm
 C3,C4,C5 = 10 nF, 50 V, steek 5 mm, X7R

Halfgeleiders:

LD1 = 4-cijferig 7-segment LED-display
 KW4-801AVB (Luckylight)
 IC1 = I²C LED-driver TM1637

Overig:

ENC1 = draai-encoder met drukknop,
 Bourns PEC11R-4225F-N0024
 K1 = 10-polige boxheader, steek 2,54 mm
 Print 190409-2 V1.1

past. Drie RC-laagdoorlaatfilters en software-filtering worden gebruikt om de ruis te onderdrukken. De drukknop van de draai-encoder wordt in dezelfde functie verwerkt en dus ook elke 250 μs gecontroleerd.

Fouten...

Elke 50 ms wordt de werkelijke temperatuur van de soldeerbout vergeleken met de ingestelde waarde. Als het station een temperatuur hoger dan 650 °C detecteert, wordt dit geïnterpreteerd als een defecte sensor en wordt de verwarming uitgeschakeld om oververhitting te voorkomen. Als het station

na zes seconden opwarmen geen temperatuurstijging aan de punt detecteert, wordt dit ook als een fout beschouwd en wordt het opwarmen afgebroken. Bij een defect in de temperatuursensor toont het station E-01 op het display, voor een fout in de verwarming E-03. Door kort op de draaiknop van de encoder te drukken wordt de fout bevestigd en probeert het station de normale werking voort te zetten.

Bouw van de hardware

De Gerber-bestanden van de twee printen (**figuur 3**) kunnen worden gedownload van

[3]: u kunt deze bestanden gebruiken om de printen te bestellen. Printen met through-hole onderdelen zijn niet al te moeilijk. De displayprint heeft de meeste onderdelen aan de bovenzijde, met een dubbele footprint voor het displaydriver-IC1 (SOIC en DIL); natuurlijk mag maar één van de twee gebruikt worden. Het LED-display en de draai-encoder worden aan de andere kant van de print gesoldeerd.

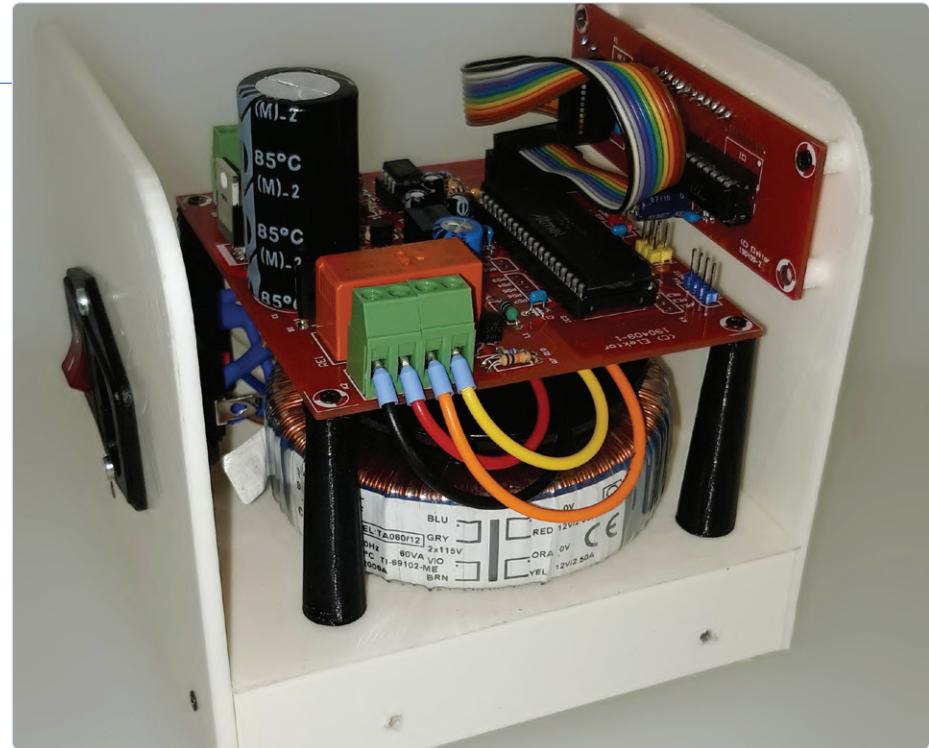
De vorige keer omvatte ons kleine soldeerstation geen behuizing. Dit zorgde er voor dat lezers moeite hadden om de elektronica in te bouwen. Dit keer is het project completer:

WEBLINKS

- [1] **Zelfbouw-soldeerstation met temperatuurregeling:** www.elektormagazine.nl/magazine/elektor-72/42292
- [2] **UPDI-programmer voor ATmega4809 en ATTiny816/817:** www.elektormagazine.nl/labs/arduino-for-updi-programming-for-atmega4809-and-tiny816817-with-jtag2updi
- [3] **Projectpagina bij dit artikel:** www.elektormagazine.nl/190409-02
- [4] **Behuizing STEP-bestanden:** <https://github.com/DK1CMB/Elektron-SolderironCase>
- [5] **Knop STEP-bestanden:** www.thingiverse.com/tracert/designs

het omvat niet alleen de printen en de bedrading, maar ook een 3D-printbare behuizing die is ontworpen door Caroline Claußen. Alle STEP-bestanden kunnen worden gedownload van [4] en geprint met de meest gangbare 3D-printers zoals de Anycubic I3 Mega-S (zie het kader *Gerelateerde producten*). Eén opmerking: bij sommige onderdelen kost het printen veel tijd; laat een werkende 3D-printer echter nooit onbeheerd achter omdat deze apparaten vlam kunnen vatten omdat ze met grote stromen werken.

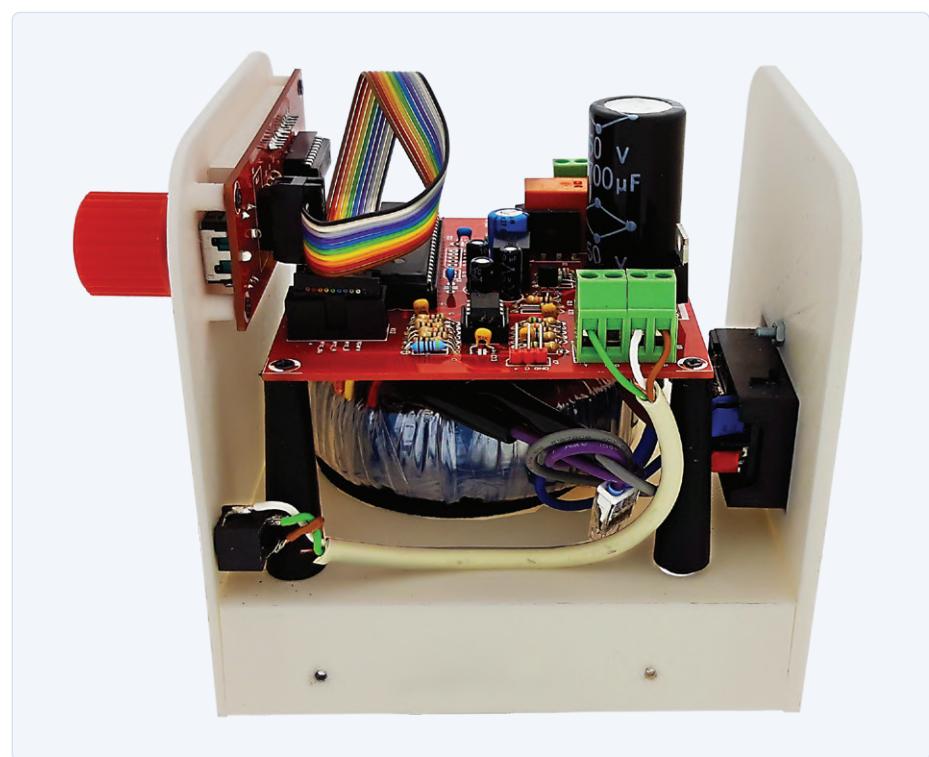
Als u zich ooit hebt afgevraagd waarom PCB CAD-tools zoals KiCad en Altium 3D-modellen voor de componenten kunnen weergeven: het maakt het ontwerpen van een behuizing een stuk eenvoudiger. Vanuit deze PCB-ontwerpprogramma's kunt u uw creatie exporteren als 3D STEP-bestand en dit gebruiken in elke tool die met dit bestandsformaat overweg kan. Vrijwel alle 3D-software kan dan worden gebruikt om een mooie behuizing rond het 3D-model van de print te construeren. Het leverde hier negen onderdelen op die geprint moeten worden. Het printen kan tot 24 uur duren, afhankelijk van uw printer. Als alle onderdelen klaar zijn, is de montage de laatste stap; er zijn echter enkele onderdelen nodig die niet geprint kunnen worden. Ten eerste is een IEC-netentree met geïntegreerde schakelaar en zekeringhouder nodig, die met schroeven en moeren in het achterpaneel bevestigd wordt (**figuur 4**). Er zijn enkele draden en faston-connectoren nodig om de schakelaar en de zekering met de primaire wikkelingen van de transformator te verbinden. Houd er bij het aansluiten van de transformator rekening mee dat de klemmen op K1 en K2 zijn gemarkerd met de eerste letters van de isolatiekleuren van de secundaire wikkeldraad (resp. zwart, rood, oranje en geel). Als u een ander merk of type transformator gebruikt, zorg er dan voor dat de 'start-of-winding'-markeringen van de draden, normaliter met stippen in de datasheet aangegeven, overeenkomen met de door ons gebruikte Multicomp-transformator. En vergeet niet om de zekering aan de primaire kant van de transformator te plaatsen. Om de voor- en achterkant en de kap aan elkaar te schroeven gebruikt u M2.5-schroeven. De print monteert u op vier afstandsbussen, die in de bodem moeten worden geplaatst. Daarna zet u de print daarop en steek u de schroeven erin; zo compenseert u de onnauwkeurigheid van uw 3D-printer. Met een beetje geduld krijg je een constructie die lijkt op die van **figuur 5**.



Figuur 4. Netentree met zekeringhouder en schakelaar.

Let op: dit soldeerstation is grondig getest met een Weller RT-soldeerbout. Helaas hadden we geen Hakko FX-8801 of JBC T245 boutjes bij de hand om te testen, maar het soldeerstation zou ook goed moeten werken met deze merken/modellen.

De aansluiting voor de soldeerbout (**figuur 6**) kan op verschillende manieren worden uitgevoerd. Er zit hiervoor zit geen pasklaar gat in de behuizing. Voor ons prototype met de Weller RT-soldeerbout hebben we een connector voor een 3,5mm-klinkstekker gebruikt – niet de beste oplossing maar het werkt wel. Gebruik



Figuur 5. Een blik in de 3D-geprinte behuizing.

geen flexibele audioverlengkabel, de koperdoorsnede daarvan is te klein om de stroom voor het verwarmingselement te transporteren. We hebben dit geprobeerd en in plaats van de soldeerbout op te warmen hadden we binnen een minuut een meer dan handwarmer kabel. We hebben dit opgelost door een kabel met een grotere koperdoorsnede te gebruiken. Als laatste: de knop. Omdat iedereen zijn eigen voorkeuren heeft wat betreft knoppen (afmeting en vorm), kun u bij Thingiverse zoeken naar de draaiknop die u het beste bevalt. We hebben hier als voorbeeld het ontwerp van Domain Mittu [5] genomen. Voor de knop hebben we TPU-filament gebruikt, dit is zachter en geeft een fijne grip. We hebben de rode kap geprint met PTEG-filament; dat is niet zo gemakkelijk te hanteren maar is sterker dan PLA. Maar u kunt de kap ook met PLA printen.

Gebruik

Voordat een soldeerbout voor de eerste keer wordt aangesloten, houdt u de drukknop van de draai-encoder ingedrukt tijdens het inschakelen. Dit opent het opstartmenu voor de selectie van het type soldeerbout. U kunt kiezen uit C0, C1 en C2, overeenkomend met achtereenvolgens de Hakko FX-8801, de JBC T245 en de Weller RT. Druk tien seconden op de draai-encoder om deze selectie op te slaan; het station zal dan opnieuw opstarten met de juiste spanning (12 V/24 V) en de correcte analoge ingang voor de temperatuurmeting. Deze selectie wordt natuurlijk opgeslagen en blijft bewaard na het uitschakelen. Tijdens normaal bedrijf wordt de draai-encoder gebruikt om de temperatuur van de soldeerbout aan te passen. Het display licht op met maximale helderheid wanneer deze instelling wordt gewijzigd. De nieuwe temperatuurstelling wordt opgeslagen vijf seconden nadat de knop is losgelaten. Het display dimt vervolgens naar normale helderheid en toont de werkelijke temperatuur van de punt. Tijdens het opwarmen van de bout knippert de linker decimale punt van het display. We zijn er zeker van dat met dit nieuwe soldeerstation de meeste minpuntjes van het eerdere ontwerp zijn opgelost, maar er is natuurlijk nog ruimte voor verbetering. Om de hardware gemakkelijk te kunnen uitbreiden, hebben we bijna alle ongebruikte microcontroller-pinnen naar uitbreidingsconnectoren op de hoofdprint geleid. Mocht u zelf aanpassingen of uitbreidingen maken die interessant kunnen zijn voor andere lezers, laat het ons dan weten! ↗

190409-02



Figuur 6. Aansluiting van de Weller RT.

Vragen of opmerkingen?

Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteurs of naar de redactie van Elektor, via redactie@elektor.com.

Een bijdrage van

Idee, ontwerp en tekst: **Mathias Claussen**,

Luc Lemmens

Afbeeldingen: **Patrick Wielders**

Redactie: **Luc Lemmens**

Vertaling: **Hans Adams**

Layout: **Giel Dols**



GERELATEERDE PRODUCTEN

➤ **Weller WE 1010 Digitaal soldeerstation (Education Kit)**
www.elektor.nl/weller-we-1010-digital-soldering-station-education-kit

➤ **Fumetractor met LED-verlichting**
www.elektor.nl/fumetractor-with-led-light

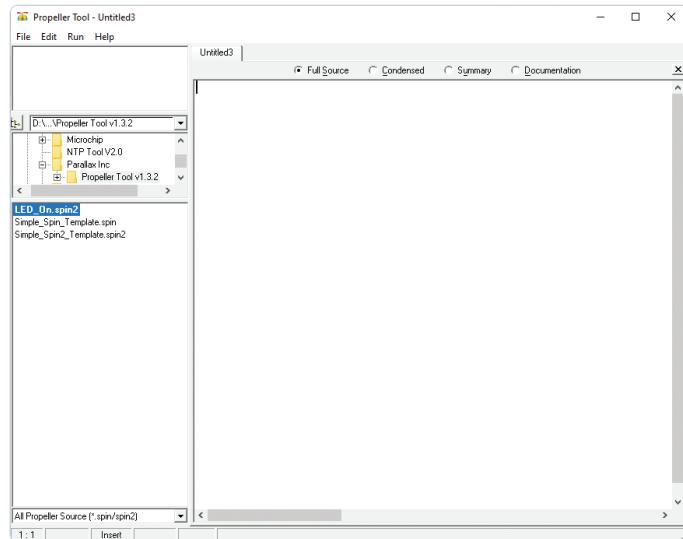
➤ **Anycubic i3 Mega-S 3D-printer (kit)**
www.elektor.nl/anycubic-i3-mega-s-3d-printer-kit

Kennismaking met de Parallax Propeller 2 (deel 2)

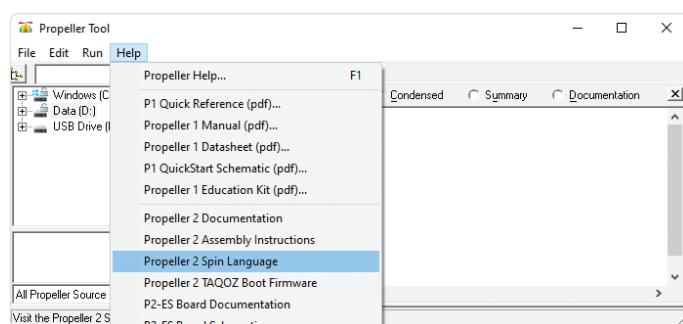
de ontwikkelomgeving en de code

Mathias Claußen (Elektor)

In het eerste deel van deze serie hebben we de Propeller 2 geïntroduceerd, de geavanceerde nieuwe multicore microcontroller van Parallax. Nu bekijken we hoe we met de taal Spin2 een LED kunnen aansturen.



Figuur 1. Gebruikersinterface van de Propeller Tool.



Figuur 2. De Spin2-documentatie is via het Help-menu toegankelijk.



Nadat we in deel 1 de mogelijkheden van de Parallax Propeller 2 hebben leren kennen, kijken we nu naar de ontwikkelomgeving en het schrijven van code. Lees hier hoe u de Spin2-taal kunt gebruiken om een LED aan te sturen.

Toegang tot de onboard-LED's

Laten we beginnen met de software-omgeving en het aansturen van een van de LED's op het board. We ontwikkelen onder Windows 10 met de door de leverancier geleverde tools. Het resultaat is een basis-softwaresetup om code te compileren en te uploaden naar de Propeller 2. Parallax heeft zijn Propeller Tool versie 2.3 Alpha ter beschikking gesteld, inclusief Propeller 2-ondersteuning voor Spin/Spin2. Als assembleertaal uw voorkeur heeft, kunt u PNut gebruiken als ontwikkelomgeving of van inline assembler gebruik maken. Als u net als ik graag in C/C++ codeert: de compiler en ontwikkelomgeving zijn momenteel helaas nog niet beschikbaar. Er wordt aan gewerkt om dit te verbeteren, zoals te zien op de videopresentatie [1] over C/C++ op de Propeller 2.

De basis-ontwikkelomgeving

U kunt de Propeller Tool 2.3 Alpha downloaden op [2]. Download eerst de Propeller Tool 1.3.2 en voeg als tweede stap in de programmap de Propeller Tool 2.3 Alpha-executable toe. Nadat de installatie is voltooid, kunt u de editor starten en beginnen met coderen. De gebruikersinterface van de tool is te zien in **figuur 1**.

Coderen in assembleertaal of in Spin2?

De vraag is: Spin2 of assembleertaal voor de eerste code? Om het eenvoudig te houden, gaan we voorlopig aan de slag met Spin2. Spin2 is een interpreter zoals BASIC, maar dan anders. De commando's en taalreferenties voor Spin2 zijn in de Propeller Tool te vinden onder de menu-optie Help (**figuur 2**).

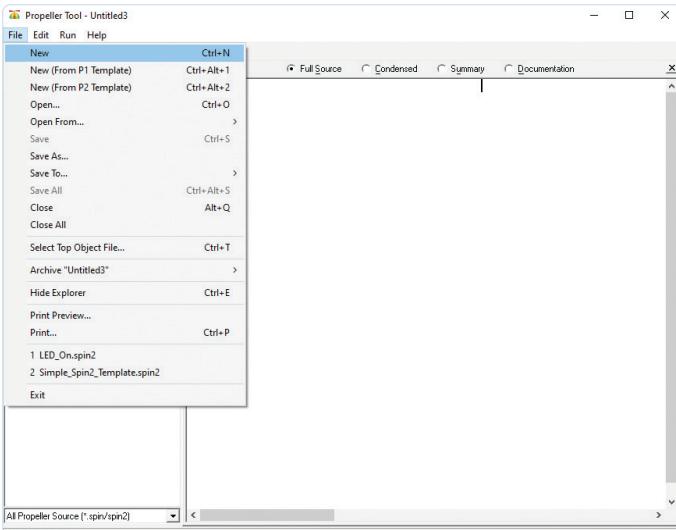


Figure 3. Een nieuw Spin2-project aanmaken.



Figure 4. Close-up van de LED's die zijn aangesloten op MCU-pinnen 56...63.

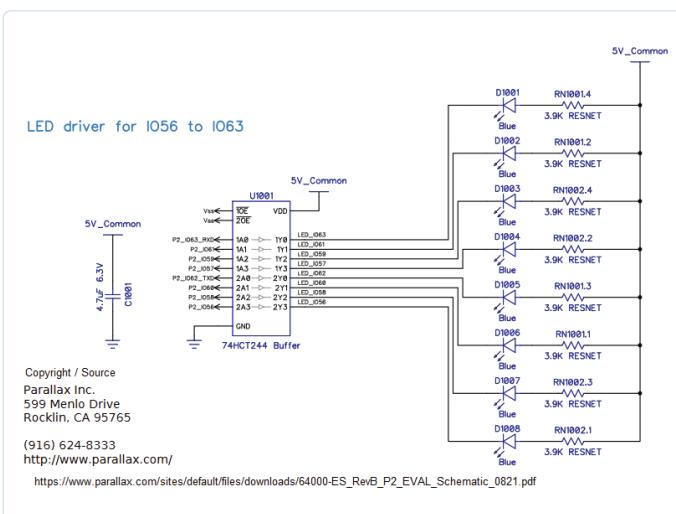


Figure 5. De LED's worden aangestuurd via een buffer van het type 74HCT244.

Om met Spin2 aan de slag te gaan, kunt u gebruik maken van een handleiding, maar bedenk wel dat deze nog niet af is. Via het menu krijgen we dus de voorlopige documentatie voor de geïmplementeerde Spin2-taal op de Propeller 2 MCU.

Aangezien dit uiteindelijk geïnterpreteerde code is, zijn ongeveer zes instructie-cycli, dus 12 klok-cycli, nodig om één commando uit te voeren. Dit is – in termen van klokcycli – niet de snelste manier, maar met Spin2 beginnen is gemakkelijker dan met assembleertaal. Omdat Spin2 meer een high-level benadering heeft, blijven van commando's de meeste assembleertaal-instructies verborgen en zijn ze makkelijker te gebruiken. U kunt een nieuw Spin2-project maken zoals in **figuur 3** getoond.

I/O-pinnen

Voor we aan het coderen slaan, kijken we kort hoe de I/O-pinnen werken. We gebruiken ze hier als gewone I/O-pinnen; we zullen later een uitgebreid overzicht geven over al hun mogelijkheden. In het algemeen moeten we de pin van onze keuze instellen als een uitgang om iets met een laag of hoog niveau aan te sturen. Op het evaluatie-board zijn de LED's (gemarkeerd met P56...P63) duidelijk zichtbaar (**figuur 4**). Dit is een groep LED's die verbonden is met pinnen 56...63 op de Propeller 2; we gaan hier de eerste van die LED's (nummer 56) aansturen. Uit het schema (**figuur 5**) blijkt dat ze niet rechtstreeks op de chip zijn aangesloten, maar via een achtvoudige buffer van het type 74HCT244. De anodes van de LED's hangen aan V_{CC} ; de I/O-pin kan met behulp van de buffers de kathodes aan massa leggen om de betreffende LED' te doen oplichten.

Geïnverteerde LED

Voor de code die we gaan schrijven, betekent dit dat een hoog niveau op de MCU-pin de LED zal uitschakelen en een laag niveau hem zal laten oplichten. Met deze omgekeerde logica in gedachten, moeten we in onze code pin 56 hoog maken om de LED uit te schakelen. Als we niets doen, is de pin standaard als ingang geconfigureerd en zal de achtvoudige buffer denken dat de pin logisch hoog is, zodat de aangesloten LED wordt uitgeschakeld. Om in onze Spin2-code de LED te laten oplichten, moeten we dus het volgende doen:

- de MCU initialiseren, plus tenminste één core (cog);
- pin 56 als uitgang configureren;
- pin 56 laag maken;
- niets doen.

Stap één, de initialisatie, wordt in dit geval voor ons gedaan; daar hoeven we ons nu geen zorgen over te maken. De Spin2-code zelf beslaat slechts een paar regels. De code begint met de functie `pub main()` gevuld door de methode `pinwrite()` (**figuur 6**). Die `pinwrite()` is ingebouwd in de Spin2-taal en maakt het mogelijk om een of meer pinnen met een gegeven waarde in te stellen. Hier gebruiken we pin 56, waar onze LED op is aangesloten, en geven voor het uitgangsniveau een '0' door om de pin laag te maken en de LED in te schakelen. De laatste regel `repeat` forceert de cog in een eindeloze lus aangezien de code onder de `repeat` zal worden uitgevoerd. In dit geval staat daar verder geen code, waardoor de cog gedwongen wordt niets te doen en te blijven doorgaan. Zonder `repeat` zou de cog het einde van de code bereiken en ophouden de I/O-pinnen aan te sturen.

De code naar de Propeller 2 overbrengen

Als de code klaar is, kunt u deze uploaden naar het Propeller 2-evalu-

The screenshot shows the Propeller Tool interface with the title bar "Propeller Tool - P2: LED_On". The menu bar includes File, Edit, Run, Help. The left sidebar shows a file tree with "LED_On" selected. The main window displays the code:

```

P2: LED_On* | 
pub main()
pinwrite(56, 0)
repeat

```

The status bar at the bottom shows "All Propeller Source (*.spin/spin2)" and "1: 9 Modified Insert Compiled PUB main - 8 bytes".

Figure 6. De code om LED56 te laten oplichten.

atieboard en daar doen uitvoeren. Hiervoor sluit u het board aan op een van de USB-poorten van het systeem en kiest u in het menu *Run->Compile Current->Load RAM* om de code direct in het RAM van de MCU te uploaden en te laten uitvoeren. Aangezien de LED op pin 56 nu oplicht, is deze eerste soort "Hello world" nu klaar. Nu kunt u zich afvragen of we de LED ook kunnen laten knipperen – ja natuurlijk

kan dat. In Spin2 hebben we een **WAITMS**-instructie die bijvoorbeeld met **WAITMS(500)** de uitvoering van code 500 ms zal vertragen. Aangezien we hebben gezien dat de code na **repeat** steeds opnieuw wordt uitgevoerd, is de volgende stap het laten knipperen van de LED. Gebruik de informatie die u tot nu toe hebt verzameld om de code te ontwikkelen en aan te passen zodat de LED gaat knipperen. Hiervoor zullen we later een voorbeeldoplossing geven.

Nu we een I/O pin kunnen aansturen, is de volgende stap om uit te zoeken hoe we seriële data kunnen versturen. De meesten van ons zijn gewend aan deze manier van debuggen, vooral als eerste of tweede stap op een apparaat met een AVR. Aangezien dit de Smart-Pin functies zal betreffen, zullen we ons in de volgende aflevering daarmee vertrouwd maken. ▶

200479-B-04

Een bijdrage van

Tekst en illustraties:

Mathias Claußen

Vertaling: **Jelle Aarnoudse**

Redactie: **Jens Nickel,**

C.J. Abate

Layout: **Giel Dols**

Vragen of opmerkingen?

Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur via mathias.claussen@elektor.com of naar de redactie van Elektor via redactie@elektor.com.

WEBLINKS

- [1] Parallax, "Propeller 2 Live Forum Early Adopter Series – C Programming with Eric Smith," 6 juli 2020: <https://bit.ly/propeller2-c>
- [2] Download van de Propeller Tool 2.3 Alpha: <https://propeller.parallax.com/p2.html#software>

Advertentie

Zelf EMC testen uitvoeren?

www.rentalab.nl

€150,-
per 2 uur
excl. BTW

Geautomatiseerde EMC faciliteiten:

- Geleide immuniteit
- Geleide emissie
- Gestraalde immuniteit
- Gestraalde emissie
- Pulse immuniteit (ESD/EFT/Surge)



rent a lab

Rent a Lab in 3 snelle stappen:

1. Ga naar www.rentalab.nl
2. Plan en reserveer een tijd slot (per blok 2 uur)
3. Bevestig & betaal direct online

Wi-Fi voor de LoRa Switch

geïntegreerd in Home Assistant met ESPHome

Clemens Valens (Elektor)

De Elektor LoRa Switch [1] heeft een fraaie relaisprint die gemakkelijk in WiFi-netwerken kan worden gebruikt als de LoRa-module wordt vervangen door WiFi. Zo krijgen we de Elektor WiFi Switch.



In het maart/april-nummer 2020 van *Elektor* hebben we een afstandsbedienende schakelaar gepubliceerd [1]. Deze bood terugkoppeling van de stand van de schakelaar en communicatie via LoRa. Omdat hij was ondergebracht in een waterdichte IP66-behuizing, was hij geschikt voor gebruik buitenhuis. Het was

daarnaast een modular project met het relais en de voeding op de ene print en het LoRa-communicatiegedeelte op de andere. Dat interesseerde me, omdat ik op zoek was naar een buitenschakelaar die gemakkelijk in mijn domoticasysteem geïntegreerd kon worden. Dit systeem is gebaseerd op Home

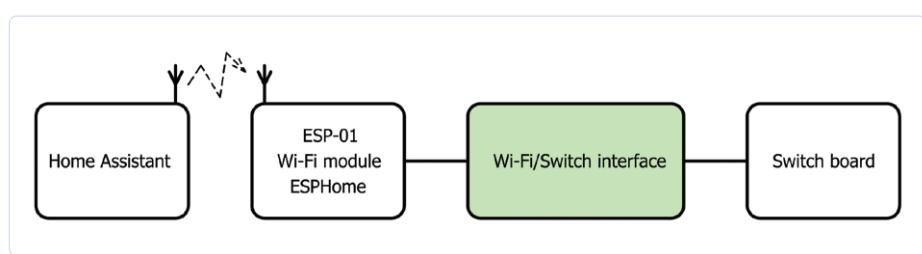
Assistant en WiFi, maar niet op LoRa. Ik weet zeker dat het mogelijk is om LoRa eraan toe te voegen, maar dat wilde ik niet. In plaats daarvan dacht ik dat ik, door simpelweg de LoRa-module te vervangen door een WiFi-module, daar ESPHome op zou kunnen draaien, dat prima met Home Assistant werkt (**figuur 1**) [2].

Definitie van de interface

Het eerste wat moet gebeuren als we iets willen aanpassen, is uitzoeken hoe het moet worden aangesloten. Het schema van de LoRa-schakelprint is niet erg ingewikkeld (**figuur 2**). Bovenaan hebben we de voeding op basis van een AC/DC-omvormer MOD1 die 5 V levert (inclusief wat spul ter bescherming en filtering).

In het midden zien we een bistabiel relais, RE1, dat de belasting in- en uitschakelt. Een bistabiel relais lijkt veel op een mechanische schakelaar doordat het zijn toestand behoudt, zelfs nadat de stroom is uitgeschakeld – net als een mechanische schakelaar dus. Er zijn twee stuursignalen om van toestand te veranderen: *set* en *reset*. Deze worden geleverd door de beide MOSFET's T1 en T2; hiervoor zijn actief-hoge stuursignalen vereist. Merk op dat het relais is beveiligd met 5A-zekering F2 terwijl het relais maximaal 16 A kan schakelen. De reden hiervoor is dat de printsporen zoveel stroom niet verdragen; de zekering beschermt ze zodoende tegen doorsmelten. De schakelprint kan belastingen tot ongeveer 1 kW aan.

Links onder zien we een optocoupler IC1 die parallel staat met de belasting. Als de belasting is ingeschakeld, is de optocoupler ook



Figuur 1. Blokschema. In dit artikel houden we ons vooral bezig met het lichtgroene blok.

ingeschakeld. De LED stuurt de transistor in geleiding en de uitgang van de optocoupler wordt laag. Als de belasting is uitgeschakeld, is de uitgang hoog. C4 filtert samen met R4 het AC-signaal (50 Hz of 60 Hz) weg, zodat een fraai stabiel niveau overblijft.

De connector rechtsonder maakt alle benodigde signalen toegankelijk. S1 is bedoeld voor aansluiting van een drukknop zodat de schakelaar ook lokaal bediend kan worden.

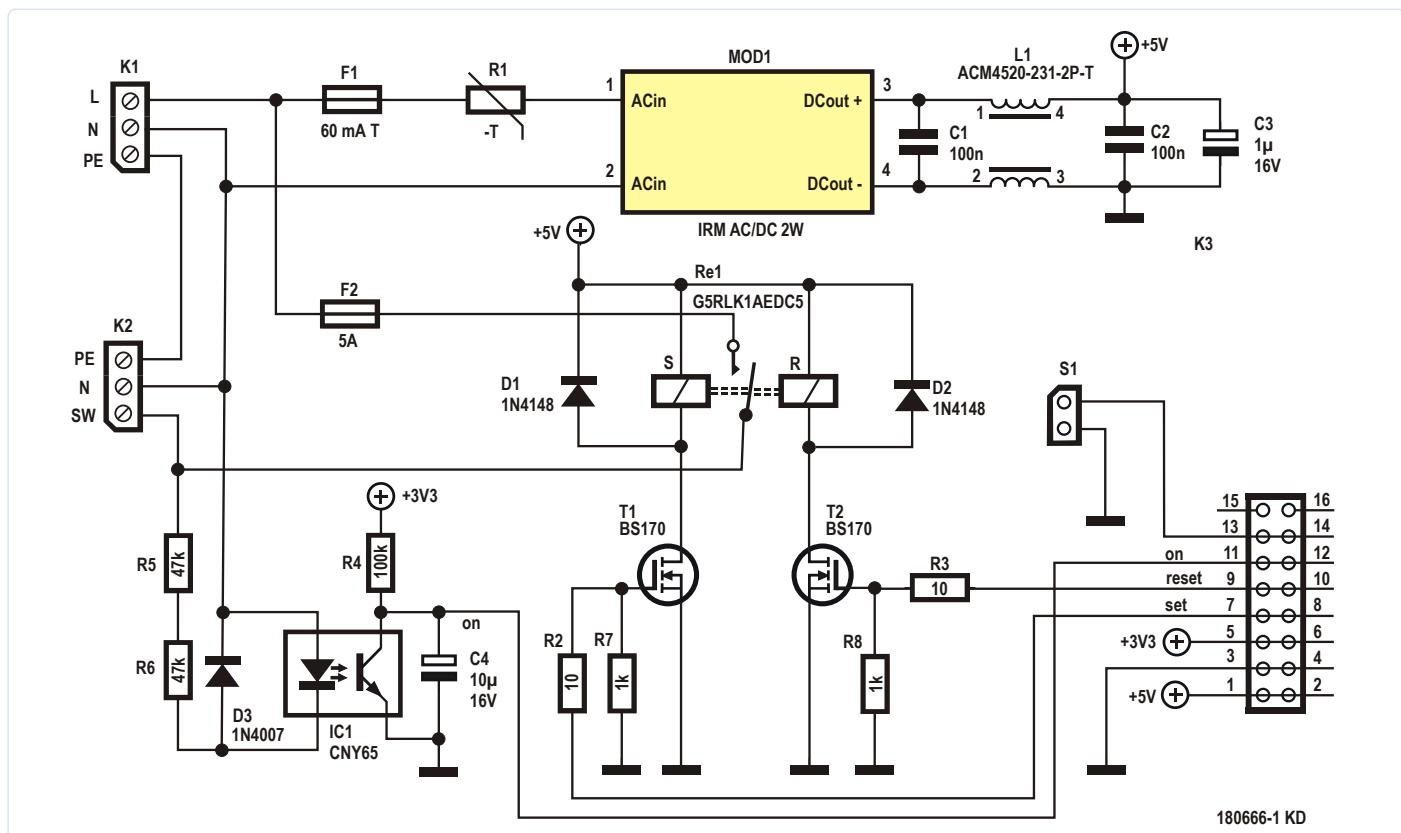
Een belangrijk detail is dat de optocoupler is aangesloten op een 3,3V-voeding, maar dat er geen 3,3V-voeding op het print voorhanden is. Deze spanning moet worden geleverd door de besturingsprint.

De ESP-01 WiFi-module

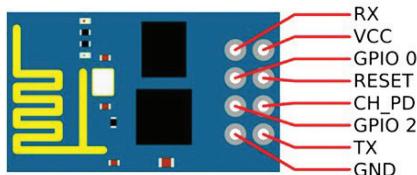
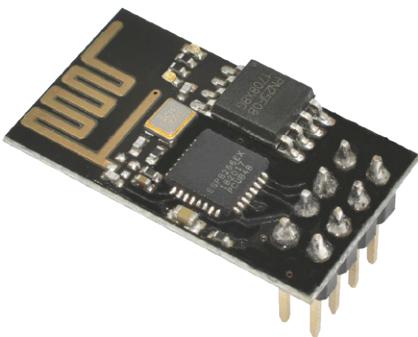
We kunnen nu de specificaties oopschrijven voor de WiFi-besturingsprint die de relaisprint moet aansturen:

- er zijn twee digitale ingangen nodig, één om de uitgang van de optocoupler te lezen en één voor S1;
- er zijn twee digitale uitgangen nodig om T1 en T2 aan te sturen;
- hij moet 3,3 V leveren voor de optocoupler.

Wie zegt "vier digitale I/O's en WiFi en 3,3 V", denkt meteen aan de ESP-01-module, want deze heeft precies vier GPIO-poorten en WiFi



Figuur 2. De relaisprint van de Elektor Lora Node (maart/april 2020, [1]) heeft een bistabiel relais als hart.



Figuur 3. De kleine en goedkope ESP-01-module is uitgerust met een ESP8266EX-microcontroller met ingebouwde WiFi-ondersteuning. Hij heeft vier I/O-poorten (GPIO 0, GPIO 2, RX & TX) en heeft een 3,3V-voeding (VCC) nodig.

en hij draait op 3,3 V. Deze populaire module op basis van de ESP8266EX is goedkoop en goed verkrijgbaar (**figuur 3**).

Er is echter één maar. De vier GPIO-poorten van de ESP-01 moeten omzichtig worden behandeld, omdat ze ook bepalen hoe de module na het inschakelen zal werken. Voor normaal gebruik moeten GPIO0, GPIO1 en GPIO2 hoog zijn tijdens het inschakelen. Dit staat ook in de opmerkingen onder de tabel 'Pin Definitions' in de datasheet van de ESP8266EX. Veel ESP8266-gebruikers zijn op de hoogte van de vereisten voor GPIO0 en GPIO2, maar niet iedereen kent die voor GPIO1, beter bekend als TXD – de seriële uitgang.

De GPIO-poorten en de stuursignalen

Op de relaisprint zijn twee van de vier stuursignalen altijd laag bij het opstarten; dat zijn 'Set' en 'Reset' vanwege de weerstanden R7 en R8. Als de belasting is ingeschakeld terwijl de relaisprint opstart, is het 'On'-signaal ook laag. Bedenk dat dit heel wel mogelijk is omdat het relais bistabiel is en ook zonder bekraftiging in de laatste stand blijft staan. Ingang S1 zweeft (is open). De toestand bij het opstarten wordt bepaald door de WiFi-besturingsprint, tenzij iemand toevallig op de drukknop drukt wanneer het systeem opstart...

Dus al met al hebben we te maken met vier signalen die allemaal *laag* kunnen zijn bij het opstarten. Deze gaan naar vier poorten, waarvan er drie *hoog* moeten zijn bij het opstarten. Het gebruik van een ESP-01 module is in deze situatie dan ook een uitdaging.

De oplossing: voeg transistors toe

Figuur 4 toont de oplossing die ik bedacht heb:

- RXD (of GPIO3) is de enige pin die vrijelijk kan worden aangesloten. Ik heb hem daarom aangesloten op de uitgang van de optocoupler omdat diens niveau bij het opstarten onvoorspelbaar is. RXD wordt een ingang.
- Als GPIO0 laag is bij het opstarten, start de module op in de flash-programmeermodus. Dit kan handig zijn voor het actualiseren van de firmware, dus heb ik deze op S1 aangesloten. GPIO0 is dus ook een ingang.
- GPIO2 en TXD (of GPIO1) zijn nu over en moeten daarom de uitgangen worden. Om ze te ontkoppelen van de lage impedanties ten gevolge van R7 en R8



ONDERDELENLIJST

Weerstanden (alle 5%, 50V, 0,1, 0805):

R1,R2,R3,R4,R5 = 10k

R6 = 470Ω

Condensatoren:

C1,C3 = 100n, 0805

C1,C4 = 10μ, 16V, steek 2 mm

Halfgeleiders:

IC1 = LD1117AS33

LED1 = LED groen

T1,T2 = BSS84

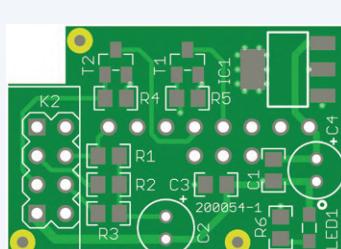
Diversen:

K1 = 8-polige pinheader, steek 2,54 mm

K2 = 2x4-polige busstrip, steek 2,54 mm

K3 = 3-polige pinheader, steek 2,54 mm

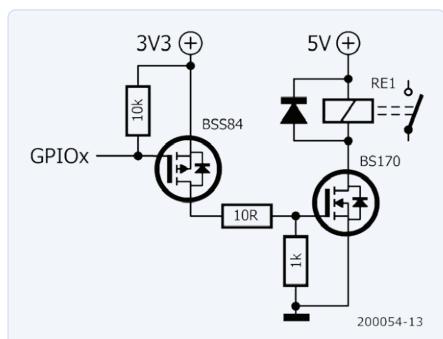
print 200054-1



op de relaisprint, heb ik P-MOSFET's gemonteerd, met een pull-up weerstand aan hun gate. Dit zorgt ervoor dat bij het opstarten GPIO2 en TXD een hoog niveau zullen zien. Wanneer de ESP-01 eenmaal is opgestart, kunnen ze worden geconfigureerd als actief-lage uitgangen.

De uitgangsdriver

De uitgangsdriver is getekend in **figuur 5**. Wanneer het GPIO-signal laag wordt, zal de P-MOSFET BSS84 geleiden. Hierdoor wordt de gate van de N-MOSFET BS170 hoog getrokken, zodat deze ook gaat geleiden en een stroom door de spoel van het relais loopt. Als het GPIO-signal hoog is, spert de P-MOSFET BSS84. De N-MOSFET BS170 spert nu ook vanwege de 1 kΩ pull-down weerstand aan de gate, en de relaisspoel krijgt geen stroom meer. De weerstand van 10 kΩ aan de gate van de BSS84 zorgt ervoor dat de GPIO-pin hoog wordt getrokken bij het opstarten.



Figuur 5. De uitgangsdriver: een P-MOSFET stuurt een N-MOSFET aan.

Laatsteloodjes

Voor de 3,3V-voeding heb ik eenvoudig een low dropout-regelaar opgenomen in de 5V-voedingslijn (**figuur 6**). In de praktijk zijn R1, R6 en LED1 niet nodig omdat ze al op de ESP-01 module zitten, daarom hebben ze 'NC' als waarde (dus niet gemonteerd). R5 is ook niet nodig, maar ik heb hem voor alle zekerheid gemonteerd. Voor R1 en R5 is een waarde van 10 kΩ geschikt, en voor R6 iets als 470 Ω. Voor de interface tussen de WiFi-module en de relaisprint heb ik een kleine print ontworpen (**figuur 7**). De bestanden zijn te vinden op [3].

Het YAML-bestand voor ESPHome

Nu de in- en uitgangen zijn gedefinieerd, kunnen we het YAML-configuratiebestand voor ESPHome schrijven (zie **kader**; het bestand is beschikbaar op [3]). Dit is iets

YAML-configuratie voor de WiFi-switch

```

esphome:
  name: wifiswitch
  platform: ESP8266
  board: esp01_1m

wifi:
  ssid: "my_ssid"
  password: "my_passphrase"

# Logging impossible as UART0 pins are
# used for switches & sensors.
logger:
  baud_rate: 0 # Hardware UART off.

# Enable Home Assistant API
api:

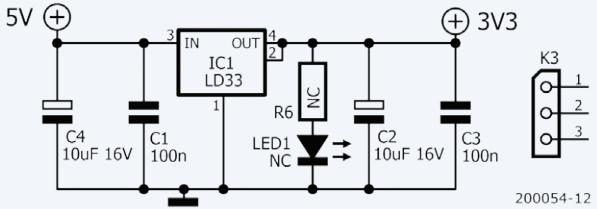
# Enable Over-the-Air programming
ota:

output:
  - platform: gpio
    id: relay_set
    pin: GPIO1 # TXD, may not be pulled low at startup.
    inverted: true
  - platform: gpio
    id: relay_reset
    pin: GPIO2 # Do not pull low at startup.
    inverted: true

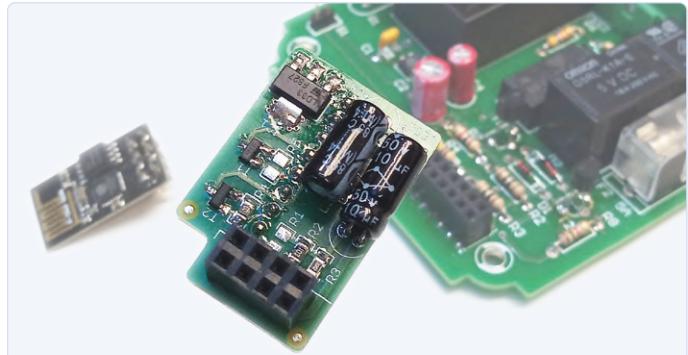
switch:
  - platform: template
    name: "Wi-Fi switch"
    id: wifiswitch
    turn_on_action:
      # Pulse the Set pin.
      - output.turn_on: relay_set
      - delay: 0.1s
      - output.turn_off: relay_set
    turn_off_action:
      # Pulse the Reset pin.
      - output.turn_on: relay_reset
      - delay: 0.1s
      - output.turn_off: relay_reset
    # Use optocoupler state as switch state.
    lambda: return id(optocoupler).state;

    # Pushbutton & optocoupler.
binary_sensor:
  - platform: gpio
    name: "Wi-Fi switch pushbutton"
    id: pushbutton
    pin:
      number: GPIO0 # Do not pull low at startup.
      inverted: true
    on_press:
      then:
        - if:
            condition:
              binary_sensor.is_on: optocoupler
            then:
              switch.turn_off: wifiswitch
            else:
              switch.turn_on: wifiswitch
  - platform: gpio
    name: "Wi-Fi switch optocoupler"
    id: optocoupler
    pin:
      number: GPIO3 # RXD, may be pulled low at startup.
      inverted: true

```



Figuur 6. Een simpele 5V-naar-3,3V adapter. K3 zorgt voor wat extra mechanische stabiliteit wanneer de WiFi-module op de connector van de relaisprint is geplikt.



Figuur 7. De interface tussen de EPS-01 WiFi-module en de relaisprint is op een klein printje gemonteerd.

gecompliceerder dan normaal vanwege het bistabiele relais dat twee pinnen gebruikt in plaats van één. ESPHome en Home Assistant kennen wel een zogenaamde 'cover'-component die dergelijke periferie kan aansturen, maar die is bedoeld voor garagedeuren en jaloezieën en dergelijke met een knop voor openen en sluiten. Deze hebben ook een speciaal pictogram. Onze schakelaar is gewoon een schakelaar en daarom willen we dat deze zich ook als een schakelaar gedraagt en er ook zo uitziet. Hiervoor gebruiken we een sjabloonenschakelaar waarmee het gedrag bij in- en uitschakelen afzonderlijk kan worden gespecificeerd.

Eerst declareren we GPIO1 (beter bekend als TXD) als een geïnverteerde uitgang met

de naam *relay_set*. GPIO2 wordt gedeclareerd als een geïnverteerde uitgang met de naam *relay_reset*. We definiëren GPIO3 ook als een actief-lage binaire sensor voor de optocoupler-uitgang.

Vervolgens specificeren we een korte puls van 100 ms op de *relay_set*-uitgang om de belasting in te schakelen, en doen we hetzelfde op de *relay_reset*-uitgang om de belasting weer uit te schakelen. Als schakelaarstand retourneren we de toestand van de optocoupler. Om drukknop S1 als verwacht te laten werken (dus door indrukken wordt de belasting in- en uitgeschakeld) definiëren we GPIO0 als een actief-laag binaire sensor. Wanneer S1 wordt ingedrukt, wordt eerst de toestand van de optocoupler gecontroleerd. Als de belasting

is ingeschakeld, wordt het relais afgeschakeld; als de belasting is uitgeschakeld, wordt het relais geactiveerd. Dit krijgen we met een *if-then-else* constructie voor elkaar.

Klaar

Na het compileren van ESPHome met dit YAML-bestand en het flashen van de ESP-01-module moet de WiFi-schakelaar in Home Assistant verschijnen en klaar zijn voor automatiseringsgebruik. We kunnen hem nu aansluiten voor het betere schakelwerk. 

200054-04

Vragen of opmerkingen?

Hebt u vragen of opmerkingen? Stuur een e-mail naar de auteur of naar de redactie van Elektor, via redactie@elektor.com.

Een bijdrage van

Idee, ontwerp, tekst en illustraties:
Clemens Valens
Vertaling: **Eric Bogers**
Layout: **Giel Dols**



GERELATEERDE PRODUCTEN

➤ **ESP8266 WiFi-module**
www.elektor.nl/17326

➤ **IoT Home Hacks with ESP8266**
www.elektor.nl/19158

➤ **LoRa-controlled Switch with State Feedback – Slave Unit kale print**
www.elektor.nl/180666-1

WEBLINKS

- [1] L. Lemmens en M. Claussen, "De Elektor LoRa Node", Elektor, maart/april 2020: www.elektrormagazine.nl/180666-04
- [2] C. Valens, "Domotica helemaal niet moeilijk", Elektor, september/oktober 2020: www.elektrormagazine.nl/200019-04
- [3] Projectbestanden op Elektor Labs: www.elektrormagazine.com/labs/wifi-switch

Geen angst voor de cellulaire module!

Tam Hanna (Slowakije)

Velen beschouwen het ontwerpen van een communicatiesysteem op basis van een cellulaire draadloze module als zwarte magie. Niets is echter verder bezijden de waarheid! Tam Hanna had onlangs het genoegen diverse van dit soort ontwerpen aan de tand te voelen. Hier presenteren we enkele van zijn notities, om u te helpen uw eigen modules te ontwerpen.

Waarom zou u uw eigen cellulaire module ontwerpen?

Kant-en-klare 3G- en 4G-communicatiemodules en USB-dongles worden te links en te rechts op het internet aangeboden, maar zoals altijd zijn het de 'kleine lettertjes' die ze ongeschikt maken voor uw ontwerp en die u dwingen uw systeem te herontwerpen. Sommige providers en applicaties gebruiken bijvoorbeeld ongebruikelijke frequentiebanden die niet door de module worden ondersteund (de US-band 13 is een notoir probleem), terwijl voor andere systemen toepassing van een COTS USB-dongle onaanvaardbaar is. En dan zijn er applicaties die een ongebruikelijke antenne (zoals SMA) of ook GPS-functionaliteit nodig hebben.

Van start!

Het begint allemaal met het vinden van een communicatiemodule die past bij de door u geselecteerde communicatieband(en) en die past bij uw criteria voor afmetingen en kosten. De kwaliteit van het *field applications*-team van de fabrikant en van de verkoper of de distributeur van de communicatiemodule is ook belangrijk, omdat u op een gegeven moment misschien ondersteuning nodig hebt. De auteur vond Quectel bijzonder open en behulpzaam. Gemalto's distributeur YAWiD in Duitsland en Telit's Poolse distributeur waren ook in orde. De auteur zet de populaire Zwitserse fabrikant U-blox echter aan het andere uiteinde van de schaal, omdat hij ten gevolge van hun e-mail-systeem geen reactie kon krijgen. Maar dat is natuurlijk een persoonlijke ervaring. Het hangt allemaal af van de regio waarin u zich bevindt, en misschien hebt u andere ervaringen.

De volgende stap is de aanschaf van een ontwikkel- of evaluatiekit voor de communicatiemodule. Als u die eenmaal hebt, moeten uw software-engineers de management-software van de module voor 100% aan de praat krijgen, met de nadruk op 100%. Hoed u voor losse eindjes die uw ontwerp later wellicht – volgens Murphy is dat zelfs waarschijnlijk – in de problemen kunnen brengen. De logica schrijft bovendien voor dat het field application-team van de leverancier uw eerste aanspreekpunt is voor de softwareproblemen die u kunt tegenkomen. De mate van samenwerking die u hier ondervindt, dient tegelijk als lakmoesproef voor de samenwerking op hardwaregebied.

Zodra de software op orde is, wordt het tijd om de documentatie van de communicatiemodule te verzamelen; die is meestal verborgen op de website van de fabrikant achter een formulier waar u een account moet aanmaken. Twee documenten zijn van bijzonder belang: het schema van het evaluatieboard en het document dat de hardware van het eigenlijke systeem beschrijft. Waar u ook nodig hebt is de

PCB-footprint van de module en een opsomming van alle pinnen en de daaraan gekoppelde functie.

Interface ahoy!

De meeste communicatiemodules kennen maar twee manieren om met de host te communiceren: een seriële poort en USB. Andere interfaces zoals I²C en SPI zijn meestal gereserveerd voor het aansluiten van randapparatuur en zijn voor ons hier niet interessant. Dus moeten we het doen met USB en serieel. Laten we beginnen met USB, de voorkeursinterface voor Linux-gebaseerde hostsysteem vanwege de eenvoudiger integratie van stuurprogramma's. De module manifesteert zich meestal als een of meer TTY-apparaten. Hoewel USB 2.0 al met een paar jumperkabeltjes kan werken, houdt dat niet in dat u hersenloos te werk kunt gaan: probeer de basisregels van de PCB-layout in acht te nemen en houd de connector zo dicht mogelijk bij de module. U hoeft zich echter geen zorgen te maken over de exacte impedantie van de USB-printsporen.

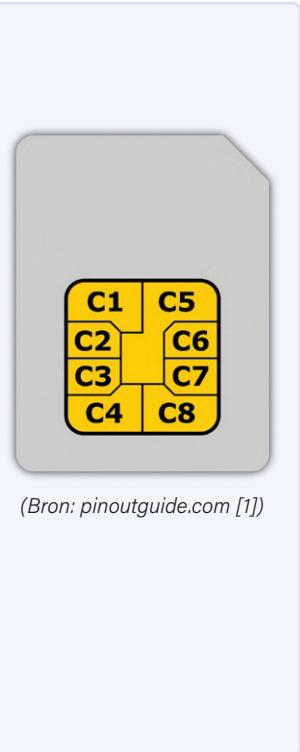
De seriële poort is een ander verhaal. Hoewel de auteur nog niet heeft gezien dat een op Linux gebaseerd systeem de seriële poort gebruikt voor de verbinding met een communicatiemodule, zijn ze op de boards van de auteur vaak wel toegankelijk. Fabrikanten van modules reserveren meestal een van de UART's als debug-interface en als deze niet toegankelijk is, kan een niet-functionerend ontwerp niet worden geanalyseerd wanneer het naar het laboratorium van de



Figuur 1. De twee niet-aangesloten soldeereilandjes links zijn verbonden met de debug-UART van de module. De connector wordt alleen op aanvraag gemonteerd.

Tabel 1. De standaard-pinout van een sim-kaart (smartcard).

Pin	Naam	Omschrijving
C1	VCC	+5 VDC voedingsspanning (optioneel gebruik door de kaart).
C2	Reset	Resetsignaal, gebruikt om de communicatie van de kaart te resetten. Ofwel zelf gebruikt (resetsignaal geleverd door het interface-apparaat) of in combinatie met een interne reset-stuurschakeling (optioneel gebruik door de kaart). Als een interne reset is geïmplementeerd, is voedingsspanning op VCC verplicht.
C3	Clock	Voorziet de kaart van een kloksignaal waarvan de timing van de datacommunicatietiming wordt afgeleid.
C4	Gereserveerd	AUX1, optioneel gebruikt voor USB-interfaces en andere toepassingen.
C5	GND	Massa (referentiespanning).
C6	Vpp	Programmeerspanningsingang (optioneel). Dit contact kan worden gebruikt om de spanning te leveren die nodig is om te programmeren of om het interne niet-vluchige geheugen te wissen. ISO/IEC 7816-3: 1997 noemt dit een programmeerspanning: een ingang voor een hogere spanning om permanent geheugen te programmeren (bijv. EEPROM). ISO/IEC 7816-3: 2006 noemt het SPU, voor standaard of eigen gebruik, als invoer en/of uitvoer.
C7	I/O	Ingang of uitgang voor seriële gegevens (half-duplex) naar het IC in de kaart.
C8	Gereserveerd	AUX2, optioneel gebruikt voor USB-interfaces en andere toepassingen.



leverancier wordt gestuurd. Hieruit volgt dat de seriële poort op zijn minst op de een of andere manier toegankelijk moet worden gemaakt. De auteur gebruikt vaak een connector die ergens geplaatst is waar er plaats voor is; wanneer het ontwerp dan in serieproductie gaat, wordt de betreffende connector niet gemonteerd (**figuur 1**).

Diversen

Bijna alle communicatiemodules maken deel uit van een familie. Hoewel het denkbaar is dat een ontwerp meerdere moduletypen uit een familie ondersteunt, is dit niet altijd even verstandig. Vooral wanneer u met low power-modules werkt (LTE M1 enzovoort), kunt u zonder 'interfamilie-compatibiliteit' een kleinere voeding gebruiken, wat kosten bespaart. Ook zijn resetschakelingen en dergelijke soms niet voor alle familieleden nodig, en dat kan ook wat kosten sparen.

Als een goed gedefinieerde resetpuls nodig is, gebruik dan een *microcontroller reset supervisor* zoals de STM1001 van STMicroelectronics. Deze SOT23-IC's zijn niet duur en zijn betrouwbaarder dan een RC-netwerkje dat u zelf in elkaar knutselt.

Over de voeding hoeft niet veel te worden gezegd – de keuze tussen

een schakelende of een lineaire regelaar is grotendeels aan uzelf. Bezuinig echter niet op ontkoppelcondensatoren. 4G-modules zijn bijvoorbeeld berucht om hun 'burststromen'.

Iets anders om op te letten: 'magic pins'. Lees de hardwarebeschrijving voor elke pin uiterst zorgvuldig, zodat u zeker weet wat ermee gedaan moet worden. Sommige moeten aan een specifieke spanning worden gelegd, terwijl andere (zoals de bootmodus-selectiepin) toegankelijk moeten zijn (soldeerelandje of tenminste een via). Als u niet zeker bent, kunt u een structuur zoals in **figuur 2** gebruiken om bij het prototype keuzemogelijkheid te hebben.

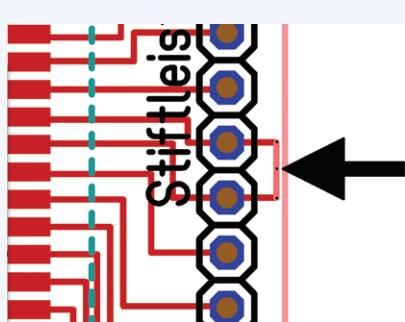
Sim-kaart

De verbinding van de sim-kaart met de communicatiemodule is altijd een interessante uitdaging. Sim-kaarthouders hebben meestal genummerde contacten, terwijl de module labels gebruikt (zoals Vcc, Rst, Data en Clk). Gelukkig zijn de contacten op de sim-kaart gestandaardiseerd (**tabel 1**). Afhankelijk van de schakeling kan een pullup-weerstand tussen Data en Vcc nodig zijn. Soms worden ontkoppelcondensatoren gebruikt bij Rst, Clk en Data. De auteur laat de ESD-protectiediodes meestal weg, omdat de gebruiker bij zijn ontwerpen nooit toegang krijgt tot de sim-kaarthouder.

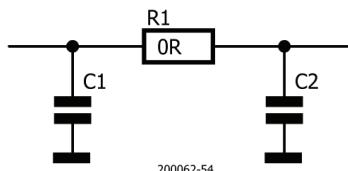
Bij sommige communicatiemodules loert hier gevaar. De aanwezigheid van een sim-kaart wordt gedetecteerd door een spanningsniveau op een bepaalde pin. Als u deze functie niet gebruikt, moet u die pin aan een spanning leggen die 'sim-kaart aanwezig' signaleert. En als u van plan bent deze functie te gebruiken, moet u controleren hoe die is geïmplementeerd bij uw sim-kaarthouder.

Houd ook rekening met de mechanische aspecten. Sommige sim-kaarthouders klappen open terwijl bij andere de (onbuigzame) sim-kaart in een bepaalde richting naar binnen moet worden geschoven.

En ten slotte moet u de gedachte van u af zetten dat communicatiemodules altijd op vierlaags-printen moeten worden gemonteerd. Met alle moderne PCB-ontwerptools kunt u kopervlakken 'gieten'. Massavlakken aan beide zijden van het board, met veel via's met elkaar verbonden, zijn meestal voldoende.



Figuur 2. Het spoor waar de pijl naar wijst, kan eenvoudig worden doorgesneden en indien nodig weer worden doorverbonden.



Figuur 3. Een pi-netwerk is vereist als antenne-aanpassing gepland is. Het kan tijdens de productie worden geneutraliseerd door voor R1 een 0Ω-exemplaar te gebruiken en C1 en C2 niet te monteren.

Antennekwesties

Dan moeten we het nog over de antenne hebben. Tot dusverre is de auteur erin geslaagd het ontwerpen en evalueren van PCB-antennes te vermijden. (De HP 8753C vector-netwerkanalyzer die hij ooit op verzoek van een klant aanschafte, staat sindsdien stof te verzamelen maar heeft zowel onderhoud als dure kalibratiekits nodig...). Zonder PCB-antenne is voor het aansluiten van een antenne op het bord een connector nodig. Of het daarbij om een SMA- of U.FL-type gaat: overleg altijd met de ontwerper van de behuizing.

Probeer de verbinding met de antenneconnector fysiek zo kort mogelijk te houden. Gebruik bovendien een transmissielijn-calculator (die zijn online beschikbaar) en probeer enigszins de aanbevolen geometrie aan te houden (dus goede massa aan weerszijden en **geen** via's in de signaalweg). Maar meestal zullen er weinig problemen zijn zolang de printsporen kort blijven.

De meeste referentieontwerpen van leveranciers raden aan om voor de zekerheid een pi-netwerkje te gebruiken – zoets als in **figuur 3**. Bij de productie wordt dat dan vaak geneutraliseerd met een 0Ω-weerstand terwijl de twee condensatoren niet gemonteerd worden. Als u niet van plan bent om de antenne exact aan te passen, kan in veel gevallen een betere resultaat worden verkregen door de afstand tussen de communicatiemodule en de antenneconnector te verkleinen en het pi-netwerk weg te laten.

Merk op dat er twee soorten GPS-antennes bestaan: actief en passief. Actieve antennes hebben een fantoomvoeding nodig die met een paar passieve componenten kan worden gemaakt; meestal is dat te vinden in het referentieschema van de fabrikant.

Nu wordt het spannend

Zodra al deze kwesties zijn geregeld, wordt het tijd om de echte hardware te produceren. Afhankelijk van de relatie die u hebt met de field application-engineers van de fabrikant waar we het eerder over hadden, kunt u het PCB-ontwerp en het schema opsturen voor beoordeling. De auteur was bijzonder onder de indruk van de service die Quectel in dit opzicht bood.

Het grootste probleem bij het handmatig bouwen van een prototype is het solderen van de onderdelen. Hoewel de auteur erin is geslaagd om een paar modules in zijn reflow-oven te solderen, maken de nauwkeurigheid van de plaatsing van stencils en onderdelen en

het gebrek aan apparatuur voor röntgeninspectie dit tot een klus die beter kan worden uitbesteed aan een bedrijf dat gespecialiseerd is in het bouwen van prototypes (inclusief een herinnering om een röntgen-inspectie uit te voeren).

Als het geassembleerde prototype een USB-dongle is, kunt u voor de eerste tests het beste een USB-hub met eigen voeding gebruiken – dit beschermt uw dure werkstation tegen beschadiging van het moederbord.

De klus klaren

Zoals bij zoveel andere ondernemingen is beginnen de belangrijkste horde die moet worden genomen. Bijna alle ontwerpen van de auteur werkten in eerste of tweede instantie. Het ontwerp aan de praat krijgen is echter slechts een klein deel van de complete oplossing – softwareproblemen kunnen bijvoorbeeld de systeemintegratie bemoeilijken. Wees echter niet bang. Gewapend met de ervaring die de auteur op de moeilijke manier heeft verzameld en hier heeft gepresenteerd, kunt u de klus ook klaren. 

200062-03

CERTIFICERING

Afhankelijk van de vraag of u een gehoorzaam burger bent, kan voor de afronding van het ontwerp overheidscertificering nodig zijn. Dat is echter weer een heel ander verhaal.

Vragen of opmerkingen?

Hebt u vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur of naar de redactie van Elektor, via redactie@elektor.com.

Een bijdrage van

Idee en tekst: **Tam Hanna**

Redactie: **Clemens Valens**

Vertaling: **Eric Bogers**

Layout: **Giel Dols**

WEBLINK

[1] Pinout sim-kaart:

https://pinoutguide.com/Memory/SmartCardIso_pinout.shtml



GERELATEERDE PRODUCTEN

- **SeeedStudio RF Explorer 3G Combo – Handheld Spectrum Analyzer**
www.elektor.nl/seeedstudio-rf-explorer-3g-combo-handheld-spectrum-analyzer

Houd de tijd bij met ESP32 en Toggl

werken met de M5Stack

Mathias Claußen (Elektor)

Het kan van pas komen de tijd bij te houden die u aan elektronica-projecten besteedt. Dat kan met verschillende services. We gebruiken hier Toggl.com om te demonstreren hoe u web-requests kunt doen met HTTPS en hoe u basale GUI-functies kunt implementeren. Dat doen we allemaal met de M5Stack, een snel prototype- en ontwikkelplatform voor ESP32-gebaseerde projecten.

Figuur 1. M5Stack Core.



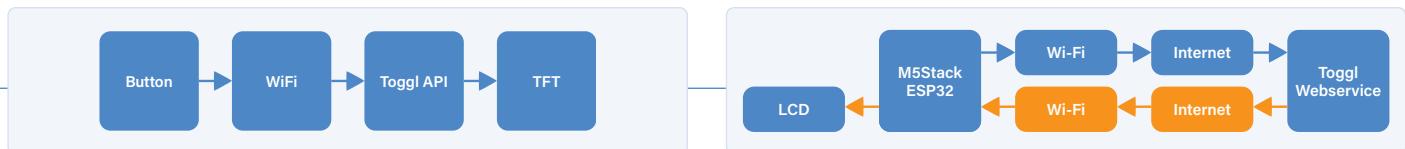
In deze moderne tijden, vooral als u van thuis uit werkt, kan het bijhouden van uw taken en de tijd die u eraan besteedt, u helpen uw aandacht erbij te houden. Het maakt het ook gemakkelijker om uw werk te bespreken – met collega's of met klanten. Er zijn veel oplossingen op de markt – zoals openTimetool, Toggl en Kimai – die tijd- en projectregistratie bieden die ook bruikbaar zijn om facturen voor klanten te genereren. Voor dit artikel heb ik de Toggl track-web-service gebruikt, een webgebaseerde oplossing die ook een plugin voor uw webbrowser of een app voor uw telefoon biedt om de tijd bij te houden. Het biedt een open API

om met een tijdregistratiesysteem te interfacen, waardoor u uw eigen app of hardware kunt bouwen voor het bijhouden van tijd met behulp van de services van het bedrijf.

KISS

“Keep it simple stupid” (KISS). Voor een tijdregistratiesysteem is dit het beste wat u een gebruiker kunt bieden. Het bijhouden van de tijd is voor sommigen van ons meer iets wat we moeten doen dan iets wat we graag doen. We hoeven maar op één knop te drukken als we aan het werk gaan, en op een andere als we stoppen – eenvoudiger kan haast niet. Aangezien

Toggl een online-service is, hebben we iets nodig dat verbinding kan maken met het internet (al dan niet draadloos) en onze request kan indienen. Voor zo'n taak denken we al snel aan de ESP32. Om een warboel van draden en componenten op een breadboard te vermijden, volgen we de KISS-route en pakken we een M5Stack Core Basic (**figuur 1**), waarvan de documentatie online beschikbaar is [1]. Naast de ESP32 krijgen we drie knoppen, een display, een batterij en wat andere periferie in een fraai compacte behuizing. Het enige dat we dan nog nodig hebben zijn een paar regels code.



Figuur 2. Datastream naar de Toggl-service.

Figuur 3. Datastream om de huidige status te verkrijgen.

Het basisconcept

Wat we nodig hebben is een apparaat dat toegang heeft tot de Toggl-webservice en dat op ten minste één knop kan reageren om aan te geven dat we aan het werk zijn, of juist niet natuurlijk. In het geval van de M5Stack betekent dit dat we WiFi gebruiken voor internetconnectiviteit, aangezien een access point binnen handbereik is en we ervan uit kunnen gaan dat we permanent online zijn. Zoals u in figuur 1 kunt zien, hebben we drie knoppen om uit te kiezen. En omdat er toch een LC-display is, maken we wat graphics om de status weer te geven (aan het werk of buiten spelen). Hoe moeilijk kan de toegang tot Toggl zijn? In theorie niet zo moeilijk, aangezien de API goed gedocumenteerd is. Het komt erop neer dat op een knop moet worden gedrukt en dat een passend commando naar de Toggl-webservice moet worden gestuurd. Vervolgens wordt de status gewijzigd en dit krijgt de gebruiker dan weer te zien. Het klinkt eigenlijk eenvoudig. **Figuur 2** toont de datastream naar de Toggl-API na een druk op een knop. De datastream voor een update van het LCD met de actuele informatie is te zien in **figuur 3**: een datastream naar de Toggl-server ergens in het net, en een datastream terug naar de ESP32. Dit klinkt eenvoudig en ziet er ook zo uit.

Voor het WiFi-gedeelte is dit al vele malen gedaan en we gebruiken code die al in veel ESP32-projecten van Elektor is toegepast. Dit verschafft ons een modulaire webserver, met een mechanisme om tijd en OTA-updates af te handelen – dat laatste is overigens meer luxe dan noodzaak. Een protocol-stack wordt gebruikt voor de communicatie met de Toggl-service.

Figuur 4 toont de communicatie-stack en de betrokken bibliotheken. Het ESP32 Arduino-framework zorgt voor de netwerk-communicatie met een WiFiClient-bibliotheek, die een verbinding mogelijk maakt met servers die bereikbaar zijn via de WiFi waarop de ESP32 is aangesloten. Daarbovenop hebben we de HTTPClient-bibliotheek die het HTTP-protocol en de HTTP-requests beheert, eveneens opgenomen in het ESP32 Arduino-framework. Aangezien Toggl vraagt om gegevens over HTTP uit te wisselen in JSON-formaat,

hebben we een extra bibliotheek nodig om JSON af te handelen. In dit geval gebruiken we de ArduinoJSON-bibliotheek.

Naast de communicatie hebben we ook een bibliotheek nodig om het display in de M5Stack aan te sturen. Een al bekende bibliotheek die met het M5Stack-display, is de TFT_eSPI die een ruime keuze aan kant-en-klare functies levert voor het tekenen van graphics. Aangezien deze compatibel is, alle vereiste functies biedt om tekst en grafische elementen te tekenen en al in vorige projecten is gebruikt, is het zonde die niet te gebruiken. Een kort overzicht van de grafische stack van de TFT_eSPI is te zien in **figuur 5**. Eerst echter gaan we nader op enkele belangrijke onderwerpen in, te beginnen met HTTP dat noodzakelijk is om toegang te krijgen tot de Toggl-service.

HTTP

Hypertext Transfer Protocol (HTTP), in 1989 bij CERN ontwikkeld, wordt gebruikt voor het aanvragen van een resource door een client (bijvoorbeeld uw webbrowser) en een antwoord van een webserver te krijgen. Als u bijvoorbeeld www.elektor.com wilt openen, stuurt uw client een request naar de webserver. Daartoe wordt een TCP/IP-verbinding tot stand gebracht en een request-string verzonden, zoals te zien in **listing 1**.

De eerste regel, een HTTP-header met **GET**, vertelt de server dat we iets vragen. De **/** vertelt de server dat we de standaard webpagina willen hebben. Met **HTTP/1.1**

Listing 1. HTTP-request.

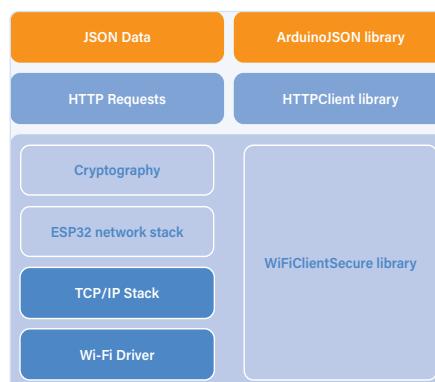
```
GET / HTTP/1.1
Host: www.elektor.com
User-Agent: curl/7.55.1
Accept: */*
```

vertellen we de server dat we HTTP-versie 1.1 gebruiken. Er wordt gewerkt aan HTTP/3 (de derde protocolgeneratie), maar voor de meeste van onze toepassingen volstaat HTTP/1.1.

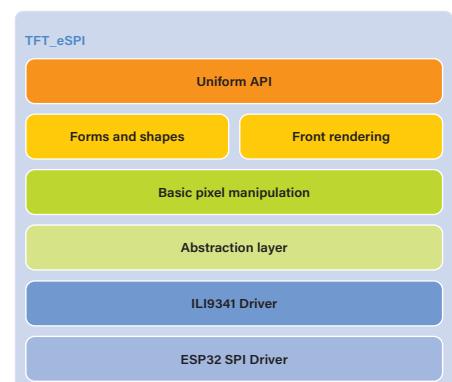
Met betrekking tot de tweede regel, de **Host**, moet u weten dat de namen die u in uw browser intypt door een DNS-service worden vertaald naar een IP-adres. Dit betekent dat elektor.com vertaald zal worden naar 83.96.255.227, het IP-adres dat wordt gebruikt voor de TCP/IP-verbinding. Achter dit IP-adres kan zich één webserver bevinden die niet alleen elektor.com afhandelt maar ook elektor.de of elektor.nl. Om de webserver te laten weten welke pagina u wilt, is de tweede regel met **Host** nodig.

Met **User-Agent** in regel 3, vertellen we de webserver wat voor HTTP-client we zijn, in dit geval **curl**. Hierdoor kan een webserver bijvoorbeeld pagina's presenteren die voor uw webbrowser geoptimaliseerd zijn, omdat Safari zich anders gedraagt dan Firefox. De laatste HTTP-header in regel 4 vertelt onze client om alle soorten inhoud te accepteren. Elk HTTP-request wordt afgesloten met een lege regel.

De webserver zal nu een antwoord op onze request formuleren, ook beginnend met



Figuur 4. Protocol-stack en bibliotheken.



Figuur 5. 'Stacked' tekenfuncties voor het LC-display.

Listing 2. HTTP-antwoord.

```
HTTP/1.1 200 OK
Server: unknown
Date: Tue, 17 Nov 2020 13:34:04 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=63072000
Pragma: no-cache
Expires: -1
Cache-Control: no-store, no-cache, must-revalidate, max-age=0
Accept-Ranges: bytes
Strict-Transport-Security: max-age=63072000
```

een reeks HTTP-headers zoals in **listing 2**. De eerste regel geeft de protocolversie en een antwoordcode. Aangezien we hebben gevraagd om HTTP/1.1, zal de server ook in deze protocolversie antwoorden. De antwoordcode die we krijgen is **200**, wat aangeeft dat ons request kon worden verwerkt. Dit wordt gevolgd door meer headers met aanvullende informatie.

Als we alleen geïnteresseerd zijn in de opgevraagde inhoud, kunnen we de header overslaan. De header wordt door een lege regel gescheiden van de inhoud die we hebben opgevraagd (niet getoond in listing 2). Voor het geval u zich dat afvraagt, de header zelf kan tot 8 KB of meer aan data bevatten. De body en de lengte kunnen worden bepaald door **Transfer-Encoding: chunked** of **Content-Length:<length>** waarbij de eerste wordt gebruikt voor inhoud van onbekende lengte, zoals dynamisch gegenereerde webpagina's, en de laatste voor inhoud waarvan de grootte van te voren bekend is, zoals plaatjes. Er zijn meer http-methodes, zoals POST, waarbij gegevens worden overgebracht van de client naar de server. Deze wordt gebruikt als u informatie indient via een webformulier. Ook de PUT-methode kan worden gebruikt om gegevens over te brengen naar een webserver. Meer informatie is onder andere bij het Mozilla Developer Network [2] te vinden voor andere gedefinieerde methoden.

Van HTTP naar HTTPS

Toen HTTP werd ontwikkeld, is niet voorzien in geïntegreerde beveiliging voor het communicatieprotocol. De informatie wordt verzonden als platte tekst en iedereen met een beetje netwerkkennt kan die lezen of wijzigen. HTTPS, waar de 'S' voor *secure* staat, voegde een extra laag toe aan de communicatiestack. De server en de client praten nog steeds via het HTTP-protocol, maar in plaats van communicatie in platte tekst via een TCP/IP-verbinding, wordt al het verkeer tussen beide partijen eerst verwerkt door *Transport Layer Security* (TLS), dat de uitwisseling van cryptografische sleutels en encryptie beheert.

Dat betekent dat wanneer we HTTPS gebruiken, we meer rekenkracht, flash en RAM nodig hebben voor encryptie en decryptie, en de extra routines voor TLS in onze software moeten worden opgenomen. Op een moderne computer is voldoende rekenkracht voorhanden. Kleinere embedded systemen zullen communicatievertra-

Index	Binair	Gecodeerd	Index	Binair	Gecodeerd
0	000000	A	33	100001	h
1	000001	B	34	100010	i
2	000010	C	35	100011	j
3	000011	D	36	100100	k
4	000100	E	37	100101	l
5	000101	F	38	100110	m
6	000110	G	39	100111	n
7	000111	H	40	101000	o
8	001000	I	41	101001	p
9	001001	J	42	101010	q
10	001010	K	43	101011	r
11	001011	L	44	101100	s
12	001100	M	45	101101	t
13	001101	N	46	101110	u
14	001110	O	47	101111	v
15	001111	P	48	110000	w
16	010000	Q	49	110001	x
17	010001	R	50	110010	y
18	010010	S	51	110011	z
19	010011	T	52	110100	0
20	010100	U	53	110101	1
21	010101	V	54	110110	2
22	010110	W	55	110111	3
23	010111	X	56	111000	4
24	011000	Y	57	111001	5
25	011001	Z	58	111010	6
26	011010	a	59	111011	7
27	011011	b	60	111100	8
28	011100	c	61	111101	9
29	011101	d	62	111110	+
30	011110	e	63	111111	/
31	011111	f			
32	100000	g			=
		OPVULLING			

Tabel 1. BASE64-coderingstabel (bron: <https://tools.ietf.org/html/rfc2045#section-6.8>).

gingen ondervinden als encryptie/decryptie en het beheer en de uitwisseling van sleutels uitsluitend in software worden gedaan. Daarom vindt u op moderne MCU's cryptografische accelerators die de encryptie van uw MCU overnemen. Bij de ESP32 is hardwareversnelling beschikbaar om dit proces te versnellen.

Request en antwoord

De Toggl-service biedt een open, gedocumenteerde API [3] om gegevens uit te wisselen. De API werkt via het indienen van POST-, PUT- en GET-requests naar een URL met gedefinieerde header en payload. Alle acties vereisen een authenticatie-sleutel die kan worden verkregen van de Toggle-webpagina. Met de verstrekte informatie kan het opbouwen van een request beginnen. We gebruiken de `WiFiClientSecure` klasse in combinatie met het http client-object. Dit is een bewezen manier om te communiceren met de server, zodat we niet opnieuw het wiel moeten uitvinden als het gaat om het verwerken van http-data.

We genereren van het autorisatie-token, een Base64-gecodeerde combinatie van gebruikersnaam en wachtwoord, met de hand. Als invoer voor de Base64-codering hebben we een API-sleutel en `:api_token` als een string. We kunnen de API-sleutel van de Toggl-service verkrijgen. Aangezien deze niet in platte tekst wordt verzonnen, kan hij tekens bevatten die problemen veroorzaken met het gereserveerde teken voor de header zelf. Om gegevens van welke aard dan ook in deze requests op te nemen, is de oplossing een BASE64-codering van data. Gewoonlijk kunnen onze gegevens in een byte een waarde tussen 0 en 255 hebben. BASE64 gebruikt slechts 64 waarden (zie **tabel 1** voor een vertaal tabel) die zijn toegewezen aan het equivalent van de ASCII-karakters **A-Z, a-z, 0-9, +, /** en **=** als opvulling. Als we bijvoorbeeld Elektor als een string in BASE64 coderen, krijgen we **RWxla3Rvcg==** (zoals te zien in **figuur 6**).

Opgemerkt moet worden dat e-mail-attachments om vergelijkbare redenen ook worden gecodeerd met BASE64, en zoals u in het voorbeeld kunt zien, wordt een attachment hierdoor ongeveer 37% groter dan het origineel.

Zodra de autorisatie aan de HTTP-client is doorgegeven, kan het request worden ingediend. Het volgende voorbeeld laat zien hoe een GET-request wordt geïmplementeerd en hoe de gereturneerde informatie wordt verwerkt, zie **listing 3**. De

Figuur 6. Elektor, gecodeerd in BASE64.

`GET()`-functie retourneert de antwoordcode van de server op onze request, of als we een communicatieprobleem hebben, een waarde kleiner dan nul. De antwoordcode (response code) kan waarden in verschillende bereiken hebben, zoals [4] laat zien. De bekendste code is **404 Resource not found**. We verwachten hier code **200**, Response **okay** en controleren daarop. Als er gegevens van de server worden ontvangen, kunnen deze worden opgehaald met `getString()` voor verdere verwerking.

Vergelijkbare code bestaat voor PUT en POST, met enkele kleine maar belang-

rijke verschillen, zoals te zien in **listing 4**. De `addHeader`-functie die voorafgaand aan `PUT()` wordt gebruikt zal twee extra headers toevoegen aan de request. De eerste is `Content-Type` die de webserver vertelt welk type gegevens we zullen versturen; de tweede is `Content-length`, en dit is waar HTTP plotseling een beetje anders wordt dan wat we eerder hebben gezien. Gewoonlijk zenden we na de header gegevens die tekst kunnen zijn, gescheiden door twee nieuwe regels en eindigend met twee nieuwe regels. Dit werkt voor tekst, maar bijvoorbeeld een JPEG-afbeelding

Listing 3. Code-snippet voor HTTP-request.

```
httpClient.begin(client,link);
    httpClient.setReuse(true);
    httpClient.setAuthorization((const char*)b64.c_str());
    int httpCode = httpClient.GET();
    if (httpCode > 0) { //Check for the returning code
        if(httpCode!=200){
            Serial.println("Response Error");
            String payload = httpClient.getString();
            Serial.println(httpCode);
            Serial.println(payload);
            *error = 1;
        } else{
            *Result = httpClient.getString();
        }
    } else {
        Serial.print("Request Error");
        Serial.println(httpClient.errorToString(httpCode));
    }
    httpClient.end();
```

Listing 4. HTTP-header toegevoegd.

```
httpClient.setAuthorization((const char*)b64.c_str());
    httpClient.addHeader("Content-Type","application/json");
    httpClient.addHeader("Content-length","0");
    int httpCode = httpClient.PUT((uint8_t*)(NULL),0);
    if (httpCode > 0) { //Check for the returning code
        ...
    }
```

Listing 5. Hergebruik van de verbinding.

```
httpClient.begin(client,link);
httpClient.setReuse(true);
httpClient.setAuthorization((const char*)b64.c_str());
httpClient.addHeader("Content-Type","application/json");
int httpCode = httpClient.POST(payload);
if (httpCode > 0) {
    ...
}
```

Listing 6. Voorbeeld JSON-string.

```
{ "name": "Test", "value": 1351824120,"array": [ 123.45, 321.89 ] }
```

Listing 7. JSON genereren.

```
DynamicJsonDocument doc(capacity);
JsonObject time_entry = doc.createNestedObject("time_entry");
if(description.length()>0){
    time_entry["description"] = description.c_str();
}
time_entry["created_with"] = "ESP32";
serializeJson(doc, payload);
```

Listing 8. DeserializerJSON.

```
DeserializationError er = deserializeJson(doc, Result.c_str());
if (er) {
    //something went wrong
} else {
    if(false == doc["data"].isNull() ){
        JsonObject data = doc["data"];
        uint32_t data_id = data["id"];
        const char* data_start = data["start"];
        if(false == data["description"].isNull()){
            const char* data_description = data["description"];
            Element->description = String(data_description);
        } else {
            Element->description = "";
        }
    }
}
```

Listing 9. Toggle JSON-string.

```
{
"data": {
    "id":436694100,
    "pid":123,
    "wid":777,
    "billable":false,
    "start":"2013-03-05T07:58:58.000Z",
    "duration":1200,
    "description":"Brew some coffee",
    "tags":["billed"]
}
}
```

kan bestaan uit elke combinatie van willekeurige bytes. Dit is waar *Content-length* in beeld komt. Headers worden gerangschikt en gescheiden van onze data door twee nieuwe regels. Met *Content-length* kunnen we elk type karakter of binaire data versturen. De webserver weet hoeveel bytes hij kan verwachten dankzij de opgegeven *Content-length*.

Voor de POST hoeven we alleen het content-type op te geven dat zal worden overgedragen. Er wordt geen eigenlijke payload verzonden. Als we een payload of inhoud zouden leveren die groter is dan 0 bytes, zou de http-client zelf een *Content-Length* header toevoegen. In dit speciale geval, waarbij de lengte op nul is gezet omdat we geen inhoud verzenden, wordt de header voor de content-lengte helemaal niet toegevoegd. Aangezien de server klaagt wanneer deze lengte ontbreekt, moeten we die zelf toevoegen met de *addHeader*-functie, zoals in listing 4.

Zoals u kunt zien in **listing 5**, is *setReuse()* op true gezet. Na dit request, hetzij POST, PUT of GET, zal de verbinding met de server normaliter door de client worden beëindigd. Als *setReuse()* true is, blijft de verbinding bestaan en wordt deze hergebruikt wanneer een volgende keer een request wordt ingediend. Dit spaart tijd omdat de verbinding niet opnieuw hoeft te worden opgebouwd als een volgende request wordt ingediend. Ook verbergt dit, op het moment van schrijven, een bug ergens in de netwerkstack, waardoor geheugen niet correct wordt vrijgegeven. Vroeg of laat kan dit ertoe leiden dat verbindingen mislukken omdat de ESP32 geen vrije geheugen meer heeft.

Gegevensuitwisseling met JSON

JSON staat voor JavaScript Object Notation en beschrijft een manier om efficiënt gegevens uit te wisselen tussen twee systemen. Alle gegevens worden verpakt in een string die er uit kan zien als die in **listing 6**. Deze string moet worden gegenereerd en weer gedecodeerd. De meeste omgevingen die gegevens voor webapplicaties verwerken en uitwisselen, gebruiken JSON en hebben ingebouwde functionaliteit of extra kant-en-klare bibliotheken voor genereren en opsluiten van JSON. Dit geldt ook voor kleine apparaten zoals een Arduino of ESP32.

Een bibliotheek die met het Arduino Framework en de ESP32 kan worden gebruikt is ArduinoJSON; deze kan

JSON-strings ontleden en genereren voor uitwisseling met andere systemen. Aangezien Toggl JSON gebruikt om gegevens uit te wisselen, komt deze bibliotheek nu goed van pas. Van de Toggl-API kunnen we bepalen welke informatie verwacht wordt in JSON-strings. Deze documentatie beschrijft ook hoe gegevens moeten worden gegenereerd als informatie moet worden verzonden. Voor een nieuw tijd-item moeten we een JSON-string genereren met daarin een `time_entry` object. Binnen deze `time_entry` hebben we een `description`-string nodig en een optionele `created_with`-string voor de nieuwe entry. De code in **listing 7** zal een nieuw `DynamicJsonDocument` op de heap van de ESP32 aanmaken met een vastgelegde capaciteit. Dit betekent dat geheugen in het niet-statisch gebruikte geheugen van de ESP32 wordt toegewezen met `malloc()` en later wordt vrijgegeven met `free()`. Het tegenovergestelde is `StaticJsonDocument` dat statisch in RAM wordt ondergebracht of, indien lokaal in een functie gebruikt, op de stack wordt geplaatst. `StaticJsonDocument` is sneller, maar omdat het lokaal gebonden is aan de stack betekent dit dat de grootte kleiner kan zijn dan wat we zouden kunnen gebruiken met `DynamicJsonDocument`. De functie `serializeJson()` zal de gewenste string creëren die kan worden verzonden. Het decoderen van JSON kan worden gedaan met `deserializeJson()`, dat elke fout retourneert die het tijdens het ontleden tegenkomt. Het codegedeelte staat in **listing 8**. Toegang tot de elementen binnen de JSON kan hierboven worden gezien. De string die door Toggl wordt geretourneerd, ziet eruit als in **listing 9**. De code controleert na het ontleden of het object `data` bestaat, dat wil zeggen of we een geldig resultaat hebben. Daarna benadert de code de elementen in `data` en pakt alleen de waarden die later nodig zijn voor weergave en gegevensverwerking. Dat zijn `id`, `start` en `description`. Die laatste is bijzonder, alleen als dit veld een waarde heeft binnen de Toggl-database wordt het geïntegreerd in `data`. Dit vereist dat met `data["description"].isNull()` moet worden gevalideerd dat de entry bestaat voordat we proberen er toegang tot te krijgen; anders zal de bibliotheek een exception genereren.

De volledige gegevensuitwisseling is ingekapseld in de `toggleClient` bibliotheek. Hier ondersteunt het de basisbehoefthen, het tonen van de huidige invoer, het starten van

een nieuwe en het stoppen van de huidige lopende invoer. De code is niet perfect en iedereen is welkom om verbeteringen en foutcorrecties aan te dragen, aangezien dit meer bedoeld is als een uitgangspunt dan als een afgewerkte bibliotheek. Naast het verzamelen en verzenden van gegevens, moeten die ook worden weergegeven en dat is wat nu volgt.

Pixels tekenen

Voor het tekenen wordt de `TFT_eSPI` bibliotheek gebruikt, die u misschien nog kent van een ouder Elektor project, *Touch-GUI voor ESP32 en Raspberry & co* [5], en die met verschillende displays en ESP32-combinaties kan worden gebruikt. Omdat deze bibliotheek het display van de M5Stack ondersteunt kan sommige code uit andere projecten hergebruikt worden. In de bibliotheekmap moet het bestand `User_Setup_Select.h` aangepast worden om de ingesloten configuratie voor de M5Stack te gebruiken. Binnen `ToggleButtonM5.ino` zal de regel `TFT_eSPI tft = TFT_eSPI();` een nieuw `TFT_eSPI` object aanmaken dat later gebruikt kan worden om te tekenen. Voor de initialisatie van het display hebben we slechts vier regels code nodig in onze `setup()`-functie, zoals te zien in **listing 10**. Het item `inverted display` moet op `true` worden gezet om de juiste kleurvolgorde voor het M5Stacks-display door te geven. Met `setRotation(1)` zal het display zijn onderkant hebben waar de knoppen van de M5Stack zich bevinden. Hierna kunnen alle mogelijke pixelbewerkingen gebruikt worden die de bibliotheek biedt.

Een kleine tip met betrekking tot de tekenprestaties, speciaal voor displays met de ILI9341. Deze displays zijn geoptimaliseerd voor het verwerken van volledige schermbepalen of grotere brokken schermdata. Pixelgewijs benaderen zal het tekenen minstens zeven keer trager maken. Dat komt door de manier waarop commando's en de toegang tot individuele pixels werken. De `TFT_eSPI` bibliotheek heeft het tekenen op de ESP32 al geoptimaliseerd en zal, waar mogelijk, de toegangstijden optimaliseren. Naast het instellen van pixels of het monochroom vullen van het scherm, voegt de bibliotheek ook andere tekenfuncties toe zoals lijnen, rechthoeken en printfuncties om strings op het scherm te zetten. En als iemand vraagt waarom u een bibliotheek gebruikt voor deze taak en geen driver schrijft, is het antwoord dat we het niet opnieuw willen uitvinden.

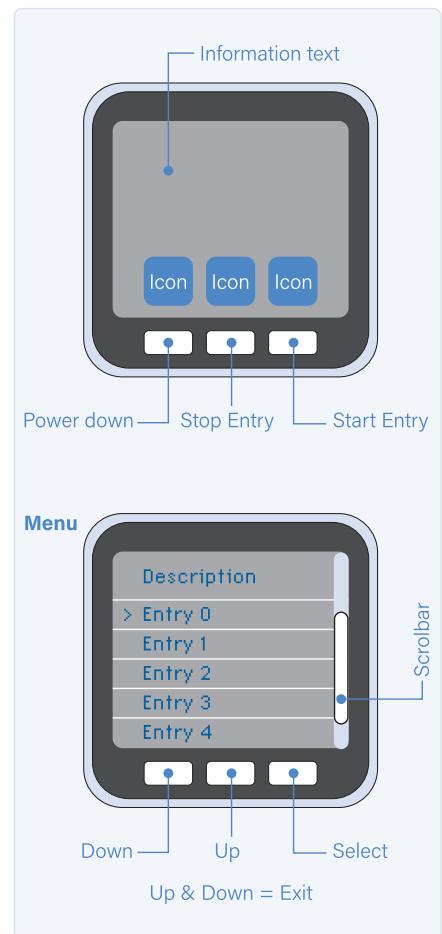
Listing 10. LCD-initialisatie.

```
tft.init();
tft.invertDisplay( true );
tft.setRotation(1);
tft.fillRect(TFT_WHITE);
```

De GUI, en een menu

De M5Stack heeft een display en drie knoppen. Time-tracking oplossingen hebben meestal slechts één knop voor het starten en stoppen van een entry. Aangezien de M5Stack ook op een batterij kan werken, kan een uitschakelknop erg handig zijn. **Figuur 7** toont het GUI-concept en ongeveer de positie van de enkele weer te geven items.

Aangezien de weergave van tekst vrij eenvoudig is, wordt de vraag natuurlijk wat we willen weergeven. De essentiële gegevens zijn daarbij de `description` (als we die hebben) en de begintijd. Voor de icoontjes gebruiken we die uit de open icon



Figuur 7. Ontwerp tekening voor de GUI.



Figuur 8: Hoofdmenu van de GUI.



Figuur 9: Description-selectie in de GUI.

library [6]; het resulterende hoofdscherm is te zien in **figuur 8**. Hiermee kunnen we een nieuwe entry starten en stoppen, en het apparaat uitschakelen. Zo blijft het geheel

gebruiksvriendelijk.

Als we een nieuwe Toggl-entry starten, zal die geen description bevatten. In principe hebben we die ook niet nodig, maar het zou

fijn zijn als op zijn minst kunnen kiezen uit een aantal vooraf gedefinieerde descriptions. Het invoeren van descriptions met slechts drie knoppen is weliswaar mogelijk maar zeker niet gebruikersvriendelijk. Een totale herziening op basis van het project ‘Wekker met drievoudige weergave’ [7] resulteerde in een eenvoudig menusysteem dat met de TFT_eSPI-bibliotheek kan worden gebruikt. De schermresolutie en de oriëntatie van het menusysteem komen ook overeen met de configuratie die we hebben voor de Toggl-knop. Code kan in dit project worden getransplanterd voor een menu zoals in **figuur 9**.

Dit is geen universeel, flexibel menu zoals u in een GUI-framework zou verwachten. Het is simpel, werkt met de TFT_eSPI en is gebouwd om met slechts een paar knoppen te worden gebruikt. Als ik meer tijd had gehad, zou ik de LVGL (*Light and Versatile Graphics Library*) [8] gebruikt hebben, omdat die ook werkt voor displays zonder aanraakfunctionaliteit. Nu we een zelfgemaakt menusysteem hebben, gaan we eens kijken hoe het werkt en hoe de graphics worden getekend. De logica voor het menu zit in een `RenderMenu()`-functie. Afhankelijk van de toestand, zal deze het menu tekenen en `true` retourneren. Als er geen menu wordt getekend, zal de functie `false` retourneren, wat aangeeft dat de inhoud van het scherm niet is veranderd. Als `true` wordt gereturneerd, moet worden vermeden om daarna nog meer te tekenen. Een vereenvoudigd stroomdiagram is te zien in **figuur 10**.

`RenderMenu()` bevat de menulogica, maar het tekenen doen we met `ShowMenuSettingsList()`. Deze functie tekent de menukop en maximaal vijf menu-items. De menulogica zorgt ervoor dat het geselecteerde item in een geldig bereik blijft. De tekenroutine berekent de startpositie en tekent vijf items als een lijst op het scherm. Ook wordt een schuifbalk (zoals bekend van veel Windows-gebaseerde systemen) berekend en getekend met een kleine indicator voor de positie binnen de lijst. De code van `ShowMenuSettingsList()` staat in **listing 11** en kan worden opgesplitst in drie delen. Het eerste is de koptekst. Om verwerkingstijd te besparen wordt deze uitsluitend opnieuw getekend wanneer dat nodig is.

Het centreren van de koptekst gebeurt handmatig door de halve lengte van de tekst in pixels te berekenen en die waarde af te trekken van de halve weergavebreedte. Dit

Listing 11. ShowMenuSettingsList.

```
void Menus::ShowMenuSettingsList(uint8_t selected_idx, bool refresh){

    /* Draw Headline */
    if(true == refresh){
        _lcd->fillRect(0,0,320,40,_lcd->color565( 45,47,50 ) );
        _lcd->setTextColor(_lcd->color565( 112,116,122 ),TFT_BLACK);
        _lcd->setFreeFont(FSB18);
        _lcd->setCursor(160- ( _lcd->textWidth("Description") / 2 ),30);
        _lcd->print("Description");
    }
    /* Headline done */

    /* Draw scrollbar */
    _lcd->fillRect(303,43,15,197,TFT_WHITE);
    _lcd->drawRect(300,40,20,200,_lcd->color565( 112,116,122 ));
    _lcd->drawRect(301,41,18,198,_lcd->color565( 112,116,122 ));
    _lcd->drawRect(302,42,16,196,_lcd->color565( 112,116,122 ));
    _lcd->fillRect(305,45 +(196/GetUsedEntryCount())*selected_idx
        ,10,(196/GetUsedEntryCount())-2 , TFT_DARKGREY);
    /* Scrollbar done */

    /* Menu entries */
    uint8_t startidx=0;
    uint8_t endindex=0;
    if(selected_idx>4){ //We can display 5 Elements
        startidx = selected_idx - 4 ;
    }
    if((startidx+5)>GetUsedEntryCount()){ //Limit last drawn item
        endindex=GetUsedEntryCount();
    }else{
        endindex=startidx+5;
    }
    /*Draw up to five elements*/
    for(uint8_t i=startidx;i<endindex;i++){
        DrawSettingsMenuEntry( (40*(i-startidx))
            ,DescriptionArray[i].c_str() ,( i==selected_idx ) );
    }
    /* Menu entries drawn */
}
```

werkt als de tekst kleiner is dan 160 pixels en kan onbekende resultaten opleveren als de tekst langer is. De tweede is de scrollbar, die aan de rechterkant van het menu wordt getekend. De scrollbar bestaat uit een aantal rechthoeken. De lengte van de scrollbar wordt berekend uit het aantal items in het menu. Dit werkt zolang er minder dan 196 items in de lijst staan, anders kunnen er rare dingen gebeuren.

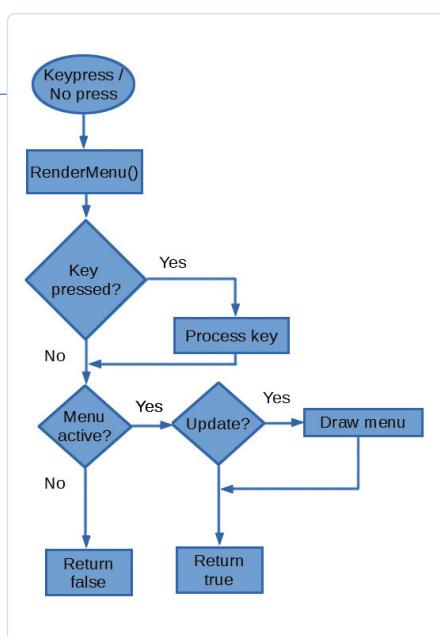
Voor het derde deel worden de stat-index en de eind-index voor de te tekenen lijst-items berekend. Aangezien we slechts vijf elementen kunnen tekenen, moeten we dat bereik dienovereenkomstig verschuiven ten opzichte van de door de gebruiker geselecteerde index. De drie delen zijn verantwoordelijk voor de lijst zoals die wordt weergegeven. Omdat elk item in de lijst er hetzelfde uitziet, wordt dit item getekend met zijn eigen functie die alleen een offset nodig heeft om op de juiste hoogte te tekenen, plus een vlag als dit item geselecteerd is. Dit genereert een slank menusysteem dat voor dit doel geschikt is en enkele basale menuconcepten demonstreert.

Webgebaseerde configuratie

De software is niet vanaf nul geschreven. Ik heb delen van eerdere projecten hergebruikt. Dit geldt ook voor de webgebaseerde configuratie. Als de twee linkerknopen worden ingedrukt tijdens het opstarten, zal de software een access-point starten waarmee u verbinding kunt maken met een computer of een draagbaar WiFi-apparaat. Anders zal de software proberen verbinding te maken met het WiFi-netwerk dat was geconfigureerd (indien aanwezig). Het starten van een webserver op een ESP32 kan eenvoudig zijn, en wordt gedaan door de software nadat WiFi is gestart en ons een webinterface presenteert, waarmee we het WiFi-netwerk kunnen instellen (**figuur 11**) en een Toggl-API-sleutel kunnen invoeren voor gebruikersauthenticatie (**figuur 12**).

Verdere ontwikkeling en wat we al weten

Hoewel de Toggl-knop werkt en het effectief bijhouden van de tijd mogelijk maakt, is er nog ruimte voor verbetering. In dit artikel heb ik slechts een paar van de vele aspecten van moderne IoT-apparaten aangestipt,



Figuur 10. Stroomdiagram menusysteem.

vooral als ze interageren met een webserver of -browser. Termen als WiFi, HTTP-server, HTTP-client, TLS, JSON, JavaScript, HTML, GUI en C/C++ klinken wellicht modieus, maar het zijn in feite de essentiële ingredi-

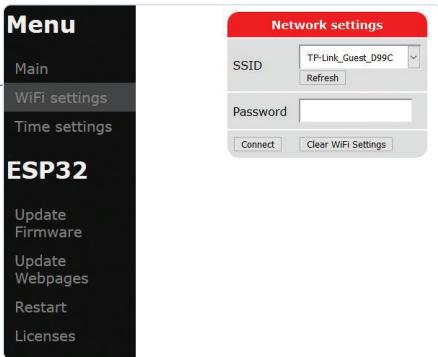
Advertentie

Ontwikkelingstools op één plaats

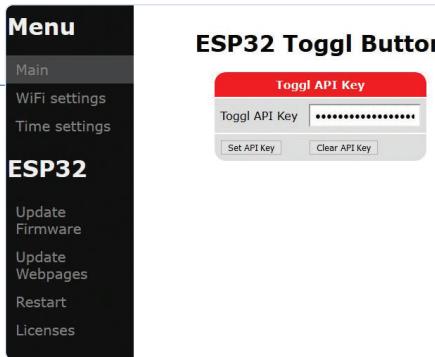
Duizenden tools van honderden vertrouwde fabrikanten

nl.mouser.com/dev-tools

MOUSER ELECTRONICS



Afbeelding 11. WiFi-configuratiepagina.



Figuur 12. Toggl API-sleutel invoeren.

enten voor dit eenvoudige project. Omgaan met HTTP en HTTPS en JSON aan de ene kant en GUI-ontwerp aan de andere, terwijl ook een webgebaseerde configuratie-interface wordt geboden, was ongebruikelijk in de beginlagen van het IoT. Maar vandaag de dag is dit het absolute minimum voor een *connected* apparaat. Deze mengeling van verschillende vakgebieden – waaronder de basis van webontwikkeling, JSON, HTML, C/C++ en JavaScript – zou een heel boek kunnen vullen. Als zo'n boek verschijnt,

zal Elektor daar melding van maken. Tijdens het schrijven van dit artikel heb ik enkele ideeën opgedaan om de knop nog gebruiksvriendelijker te maken en om functies toe te voegen die de registratie zullen verbeteren. De eerste verbetering zou de overschakeling van de GUI naar LVGL zijn. Voor de GUI brengt dit ons snel naar wat we al weten. Als u een GUI aan uw project wilt toevoegen, zoek dan naar iets dat zich bewezen heeft en beschikbaar is. Er zijn zelfs oplossingen die een vriende-

lijke licentie hebben voor uw project. Uw eigen menusysteem implementeren voor het plezier van het leren kan leuk zijn, maar het wiel opnieuw uitvinden is dat niet.

Als u voor de ESP32 het Arduino-framework en de Arduino-bibliotheken gebruikt, kijk dan eens naar de GitHub-repository en de nog niet opgeloste problemen. Dit kan een goede bron zijn om problemen te vermijden of bekende beperkingen te omzeilen. De code voor deze Toggl-knop staat op de Elektor GitHub-pagina [9] en biedt een handige manier om de code te klonen en te verkennen omdat de geschiedenis bewaard blijft. Als u bugs vindt of suggesties hebt voor de code, kunt u de GitHub issues-functie gebruiken. Aangezien dit project veel onderwerpen raakt, kunt u me feedback geven als er onderwerpen zijn die in een toekomstig artikel meer in detail moeten worden uitgelegd. ↗

200631-03

🛒
GERELATEERDE PRODUCTEN

- **M5Stack - ESP32 Basic Core Development Kit**
www.elektor.nl/m5stack-esp32-basic-core-development-kit
- **M5Stack ESP32 Arduino MPU9250 Development kit**
www.elektor.nl/m5stack-esp32-arduino-mpu9250-development-kit
- **W. Gay, FreeRTOS for ESP32-Arduino (Elektor 2020)**
www.elektor.nl/freertos-for-esp32-arduino



Een bijdrage van

Ontwikkeling, tekst en diagrammen:

Mathias Claußen

Illustraties: **Patrick Wielders**

Vertaling: **Jelle Aarnoudse**

Layout: **Giel Dols**

Vragen of opmerkingen?

Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur via mathias.claussen@elektor.com of naar de redactie van Elektor, via redactie@elektor.com.

WEBLINKS

- [1] **M5Core Basic:** <https://docs.m5stack.com/#/en/core/basic>
- [2] **HTTP | MDN:** <https://developer.mozilla.org/en-US/docs/Web/HTTP>
- [3] **Toggl.com API-documenten:** https://github.com/toggl/toggl_api_docs
- [4] **HTTP Response Status Codes | MDN:** <https://developer.mozilla.org/en-US/docs/Web/HTTP/Status>
- [5] **Touch-GUI voor ESP32 en Raspberry & co:** www.elektrormagazine.nl/magazine/elektor-144/57038
- [6] **Open Icon Library:** <https://sourceforge.net/projects/openiconlibrary/>
- [7] **Wekker met drievoudige weergave:** <http://bit.ly/elektor-3display-alarm-clock>
- [8] **Light and Versatile Graphics Library:** <https://lvgl.io/>
- [9] **GitHub-projectpagina:** https://github.com/ElektorLabs/200631_ESP32_Toggl_Button

Kennismaking met het Raspberry Pi Pico-board en de RP2040



Mathias Claußen (Elektor) en Luc Lemmens (Elektor)

De Raspberry Pi staat bekend als een single-board computer (SBC) die Linux draait, maar de nieuwe Raspberry Pi Pico is anders. Hoewel het nog steeds een Raspberry Pi is, betreedt de Pico een wereld waar de Arduino, het STM32 BluePill-board en de ESP32 de standaard zetten qua prijs, software-ecosysteem en connectiviteit. De Raspberry Pi Pico slaat een brug van uw krachtige Linux-gebaseerde SBC's naar de wereld van embedded microcontrollers.

Bekijk Elektor's "Raspberry Pi Pico Review" op Elektor.TV!



Figuur 1. Bovenaanzicht van de Raspberry Pi Pico.

Weer een Raspberry Pi?

Ja, maar nu is het een nieuw Raspberry Pi-product – de Raspberry Pi Pico (**figuur 1**). We zeggen het op voorhand, de Raspberry Pi Pico is niet bedoeld om een bestaande Raspberry Pi te vervangen. Hij is bedoeld als een uitbreiding op en een metgezel voor de bestaande Raspberry Pi-familie. Hoewel het een Raspberry Pi is, is hij totaal anders dan wat we tot nu toe hebben gezien. De Raspberry Pi Pico is een op een dual-core ARM gebaseerd board dat alleen de minimale schakelingen aan boord heeft om het te laten werken. Ook al klinkt het als een aangepaste Pi Zero, dat is het niet. De Raspberry Pi Pico gebruikt geen Broadcom-CPU zoals de andere Raspberry Pi-boards die we kennen. Deze gebruikt een chip, de RP2040, die in eigen

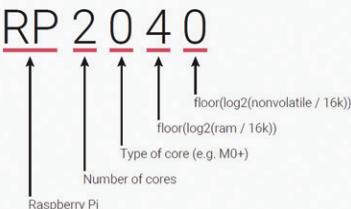
beheer is ontworpen door de Raspberry Pi Foundation zelf. De RP2040 is een dual-core ARM Cortex-M0+ microcontroller. Deze core is geen voor Linux geschikte SoC zoals we eerder hebben gezien. De Raspberry Pi Pico betreedt het gebied van de microcontrollers, waar Arduino's, STM32 BluePill-boards en op ESP32 gebaseerde boards tegenwoordig het meest gebruikt worden, terwijl op RISC-V gebaseerde SBC's (zoals de GD32VF103) steeds populairder worden. Hij is betaalbaar, met een prijskaartje van rond de € 5, wat hem in de categorie brengt van een gekloonde Arduino Nano en de STM32 BluePill. En hij kost minder dan een ESP32 Pico-kit. In dit artikel bekijken we de specificaties en kenmerken van de Raspberry Pi Pico en zijn RP2040. Bekijk zeker ook de unboxing-

1.1. Why is the chip called RP2040?

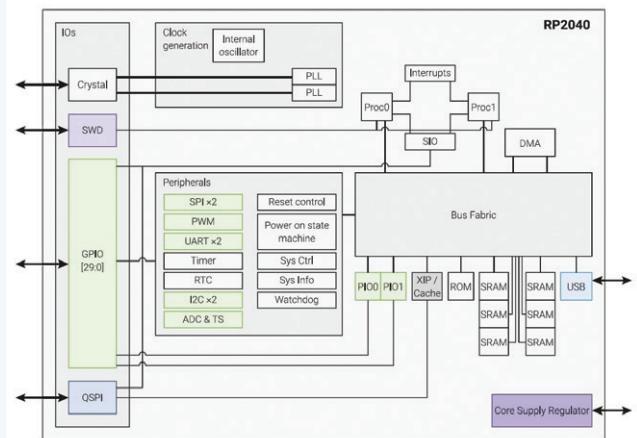
The post-fix numeral on RP2040 comes from the following,

1. Number of processor cores (2)
2. Loosely which type of processor (M0+)
3. $\text{floor}(\log_2(\text{ram} / 16k))$
4. $\text{floor}(\log_2(\text{nonvolatile} / 16k))$ or 0 if no onboard nonvolatile storage

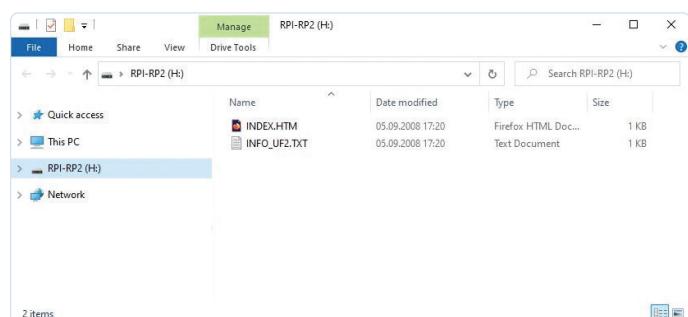
see [Figure 1](#).



Figuur 2. Zo zit de naam in elkaar [9].



Figuur 3. Binnenin de RP2040 [9].



Figuur 4. De Raspberry Pi Pico als apparaat voor massaopslag.

Tabel 1. Raspberry Pi Pico: specificaties

CPU type	Cortex M0+
Aantal kernen	2
Max. snelheid	133 MHz
SRAM	264 kB in 6 banken
Flash intern	0 kB
Flash extern	2 MB QSPI flash (tot 16 MB ondersteund)
GPIO's	26 (incl. 4 ADC)
USB	1.1 Host / Slave
ADC	12-bit @ 500 kSps
ADC-kanalen	5 (incl. temperatuursensor)
SPI	2
UART	2
I2C	2
PWM	16 kanalen
Timer	1x 64-bit
RTC	Ja
Unieke eigenschappen	programmeerbaar IO-state-machine, boot ROM met USB-massaopslag bootloader

video van Clemens Valens op het YouTube-kanaal van Elektor [1] en dit handige overzicht [2].

Een eerste blik op de Raspberry Pi Pico

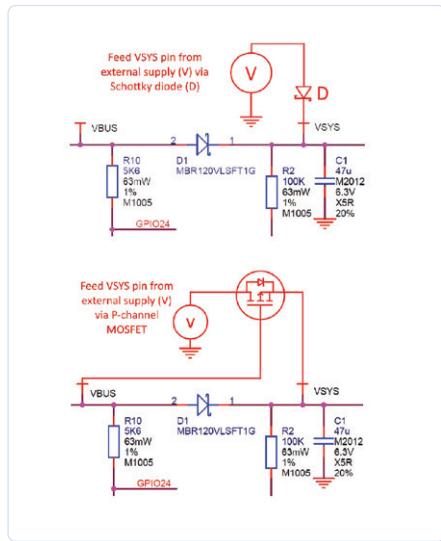
Zoals we al hebben vermeld, is de Raspberry Pi Pico uitgerust met een dual-core Cortex-M0+ microcontroller met een kloksnelheid van 133 MHz. Hoewel dual-core Cortex-M MCUs tegenwoordig niet ongewoon zijn, hebben ze meestal geen twee Cortex-M0+ kernen, maar gebruiken ze de krachtigere Cortex-M4, Cortex-M7, of de nieuwere Cortex-M33 kernen.

De RP2040-MCU die in de Raspberry Pi Pico wordt gebruikt, volgt het in [figuur 2](#) getoonde naamschema. We kunnen alleen maar raden naar welke andere versies zullen volgen, aangezien dit suggereert dat er mogelijk meer varianten komen. We hebben de Raspberry Pi Foundation gevraagd of er meer varianten zullen volgen, maar we hebben nog geen antwoord gekregen. De specificaties van deze MCU staan in [tabel 1](#).

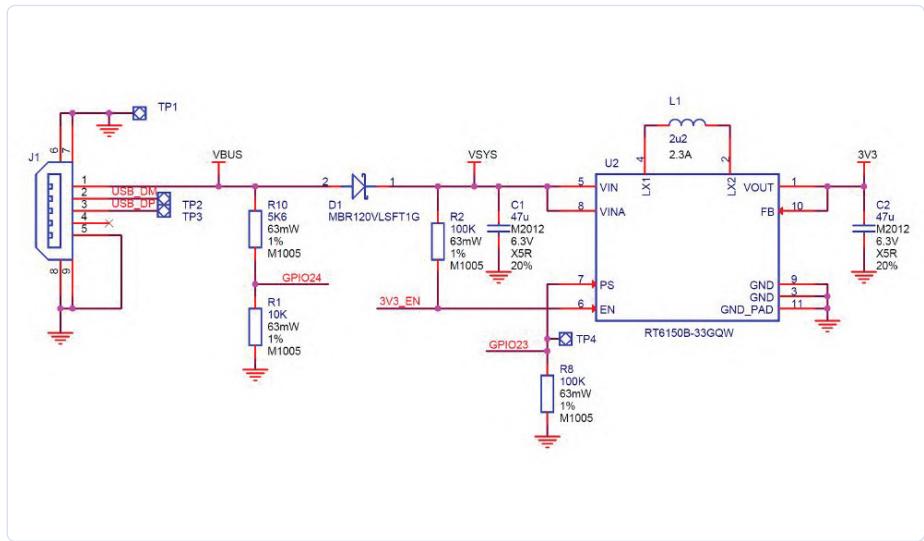
Een overzicht van de periferie en functieblokken in de RP2040 is te zien in [figuur 3](#). Een van de unieke dingen die in eerste instantie over het hoofd kan worden gezien, is de afwezigheid van flash-geheugen in de MCU. Op de Raspberry Pi Pico bevindt zich naast de MCU een klein IC – een W25Q16JUXIQ NOR-flash met plaats voor 2 MB. De RP2040 ondersteunt tot 16 MB extern geheugen, dus het vergroten van het geheugen is een kwestie van het vervangen van de flash-chip. De Raspberry Pi Pico heeft een USB-poort die kan worden geconfigureerd als host of slave in USB 1.1-modus (12 Mbit/s), dus om USB-periferie aan te sluiten of om zelf als USB-apparaat te fungeren.

De RP2040 bevat een boot-ROM, waar de gebruiker nieuwe code naar kan uploaden vanaf een computer. Om speciale programmeertoools te vermijden, meldt de geïntegreerde bootloader zich als apparaat voor massaopslag en kan de RP2040 worden geprogrammeerd door simpelweg nieuwe firmwarebestanden naar het apparaat te kopiëren ([figuur 4](#)), een eenvoudige en handige manier om apparaten te programmeren.

Ook de voeding is gebruiksvriendelijk. U kunt het apparaat van 5 V voorzien via micro-USB. Het accepteert ook spanningen van 1,8 V tot



Figuur 5. Verschillende manieren om het board te voeden [3].



Figuur 6. De onboard buck/boost converter [3].

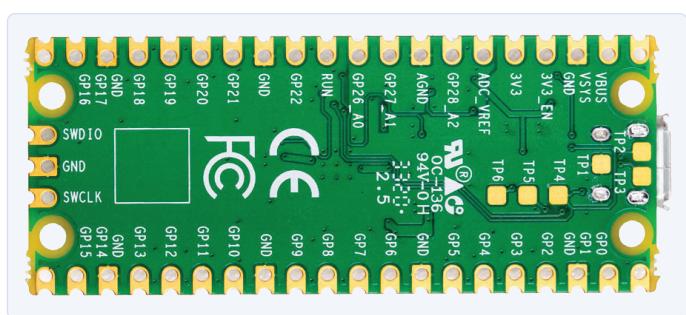
5,5 V via een DC/DC buck-boost converter op de VSYS-pin. Dit maakt ook het gebruik van een oplaadbare lithium-batterij of gewoon een set van twee of drie NiMH-batterijen mogelijk. Overgenomen uit de datasheet [3] toont **figuur 5** de verschillende voedingsmogelijkheden. Het DC/DC buck-boost IC is hier een RT6150, waarvan de datasheet te vinden is op [4]; dit IC heeft genoeg aan slechts een paar componenten om te werken, zoals te zien in het schema van **figuur 6**.

De 40 pinnen van de Raspberry Pi Pico zitten aan de lange zijkanten van het board, waardoor 26 GPIO's inclusief drie ADC-inputs van de RP2040-IO's toegankelijk zijn. De andere pinnen zijn voor voeding en massa. De maximale IO-spanning voor de Raspberry Pi Pico is 3,3 V op alle GPIO-pinnen. Aan de onderzijde van de print (**figuur 7**) is elke pin gelabeld. Er zijn drie extra pinnen op het Pico-board (SWCLK, SWIO en GND) voor de Serial Wire Debug-poort, waarmee de RP2040 kan worden geprogrammeerd en ook gedebugd.

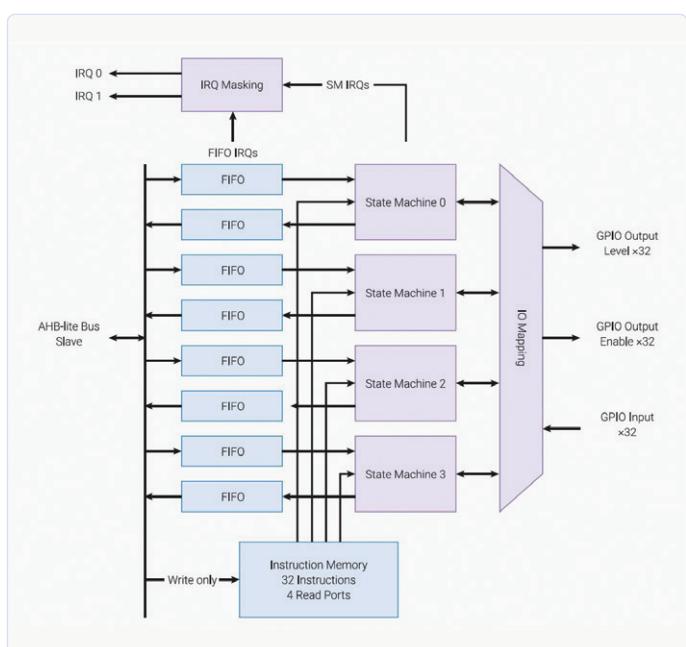
Periferie en het Programmable Input/Output-blok

De Raspberry Pi Pico met zijn RP2040 omvat, zoals te verwachten, een verzameling van de meestgebruikte periferie: twee UART's, twee I²C-controllers, twee SPI-controllers, 16 PWM-kanalen, een 12-bit ADC met 500 kSps, een geïntegreerde temperatuursensor, een real-time klok, een timer en de basis-GPIO-functies. Ook inbegrepen en minder gebruikelijk is de USB-interface, werkend als host en slave, en acht Programmable IO (PIO) toestandsmachines.

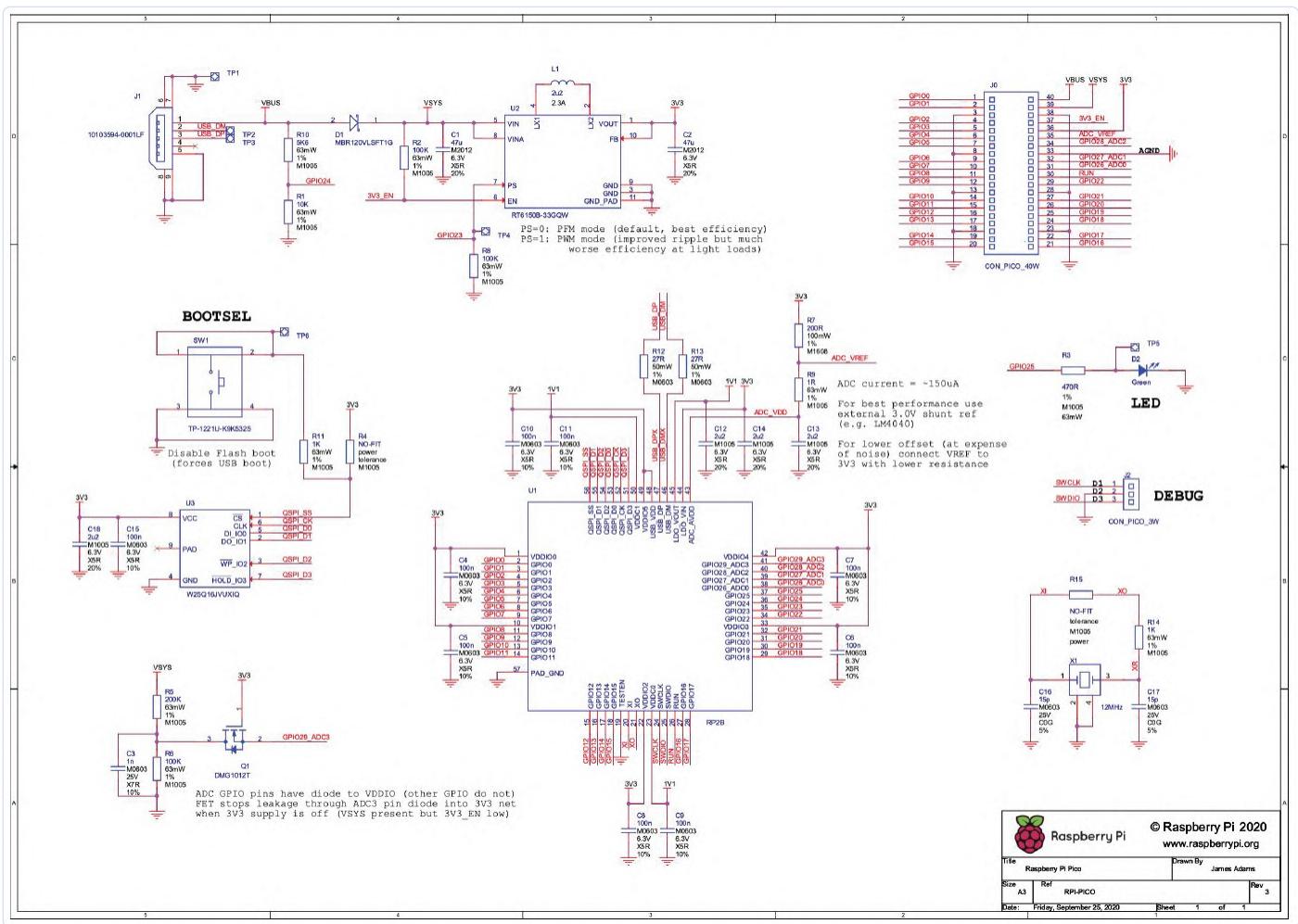
Deze laatste kunnen worden gebruikt om diverse interfaces te implementeren, zoals (extra) UART, I²C, I²S en SPI, maar ook SD-Card, VGA, DPI en nog veel meer. De datasheet van de RP2040 bevat een speciale sectie met uitleg over het gebruik, plus voorbeeldprogramma's. **Figuur 8** toont een blok met vier toestandsmachines. Deze kunnen een set van negen commando's uitvoeren: JMP, WAIT, IN, OUT, PUSH, PULL, MOV, IRQ, en SET. Omdat deze toestandsmachines onafhankelijk van de beide CPU-kernen kunnen werken, kunnen interfaces worden gevormd die niet door de onboard-hardware worden ondersteund, waarbij de CPU-kernen ontlast worden. De native UART's in de RP2040 zijn gebaseerd op de ARM Primecell UART (PL011), die ook wordt gebruikt op andere Raspberry Pi-boards, waardoor een maximale snelheid tot 961,6 kBaud.



Figuur 7. Onderaanzicht van de print met de pinlabels.



Figuur 8. I/O met programmeerbare toestandsmachine [9].

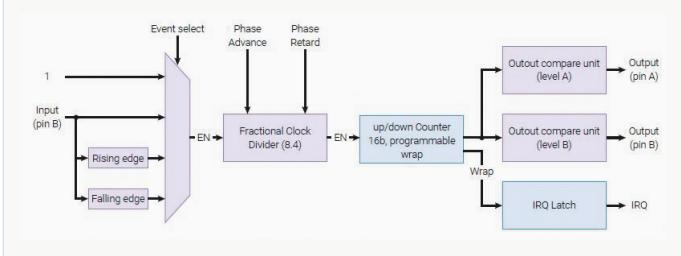


Figuur 9. Schema van het board [3].

mogelijk is. Voor de twee SPI-interfaces wordt een ARM Primecell SPI (PL022) gebruikt, die ook in andere Raspberry Pi's is ingebouwd. Deze interface kan data overdragen tot F_CPU/2 SPI klok in master-modus. De aanwezige I²C-controller ondersteunt standaardmodus (100 kHz), fast-modus (400 kHz) en fast-modus plus (1 MHz) kloksnelheden met 7- en 10-bit adressering als master of slave. Deze drie interfaces bieden connectiviteit met de meeste gebruikte externe hardware en vormen een goede balans tussen functies en complexiteit van de apparaten.

De ADC in de RP2040 is een basis-SAR met 500 kSps bij een 12-bit resolutie. Hij heeft vier inputs en een temperatuursensor. Als u

gewend bent aan de ADC's van bijvoorbeeld een AVR, treft u normaal gesproken een bandgap-referentie aan die 2,56 V of 1,1 V aan de ADC levert als referentiespanning; hier wordt de ADC_AVDD-pin op de RP2040 gebruikt. Op het Raspberry Pi Pico board betekent dit dat een externe spanning als referentie vereist is. De implementatie op de Raspberry Pi Pico is te zien in figuur 9, waar deze pin via een weerstand met 3,3 V en een filternetwerkje is verbonden. Een andere interessante eigenschap is de USB-interface die op de RP2040 en dus ook op de Raspberry Pi Pico aanwezig is. De Pico kan worden geconfigureerd als host of slave en biedt een overdrachts-snelheid tot 12 Mbit/s in USB1.1-modus. Dit maakt het gebruik van USB-apparaten zoals muis en toetsenbord en USB-connectiviteit met een PC of Raspberry Pi mogelijk. Voor de USB host-modus ondersteunt de controller het gebruik van USB-hubs, zodat u zich niet tot het aansluiten van slechts één apparaat hoeft te beperken. De RP2040 heeft één dedicated 64-bit timer die werkt met een tijdbasis van 1 µs. Hiermee kunnen tot vier alarmtimers worden ingesteld, of kunnen vertragingen in het µs-bereik worden gegenereerd. Voor het herhalen van getimedede gebeurtenissen of interrupts kunt u een van de acht 16-bit PWM-units gebruiken, elk met twee kanalen. Figuur 10 toont het blok dat in één PWM-unit is opgenomen. De GPIO-pinnen voor PWM zijn ook te gebruiken voor frequentie- en duty cycle-metingen, voor het genereren van



Figuur 10. Een RP2040 PWM-blok [9].

interrupts, DMA-requests en ze hebben een fractionele klokdeling voor een nauwkeurigere frequentie-aanpassing.

Voor meer geavanceerde programmering bevat de RP2040 een DMA-eenheid die gegevensoverdracht in het systeem mogelijk maakt zonder tussenkomst van de CPU-kernen. Zo is het mogelijk AD-omzettingen uit te voeren en de resultaten naar een voorgedefinieerde buffer in het geheugen te verplaatsen zonder tussenkomst van de CPU, of gegevens uit het geheugen naar een UART te sturen. Ook het kopiëren van grote aantallen buffers binnen de RP2040 van de ene plaats naar de andere kan zonder de CPU's en gaat zelfs sneller dan met de CPU's. Dit komt ook van pas als u gegevens naar een extern aangesloten display moet verplaatsen. Aangezien de DMA-unit ook kan worden gekoppeld aan de Programmable IO (PIO) toestandsmachines, vormt dit een krachtige manier voor snelle gegevensuitwisseling met externe apparaten.

Schema's, handleidingen en ontwerpbestanden

Het Raspberry Pi Pico-board is het eerste RP2040-board. **Figuur 9** toont het schema. De MCU is omringd door de DC/DC-converter, extern QSPI NOR flash-geheugen en wat schakelingen om de voedingsspanning te meten en de DC/DC-converter in de low-power modus te zetten, en een USB-interface.

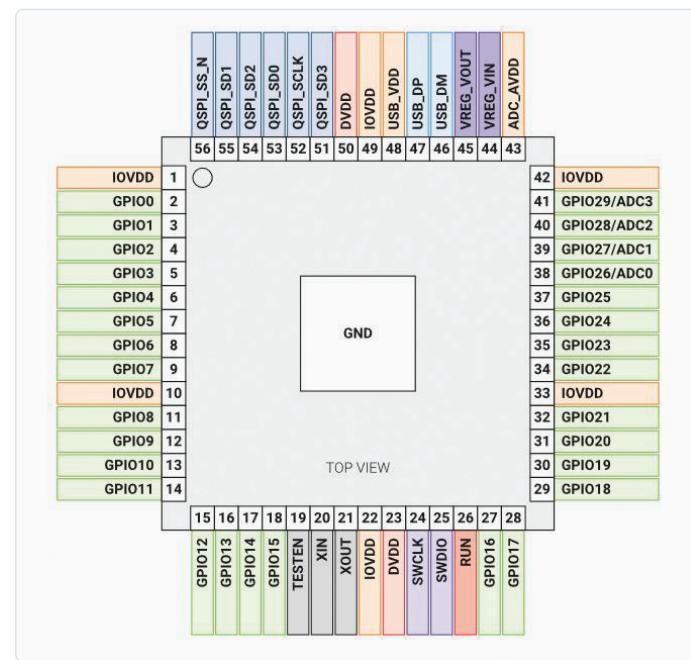
De Raspberry Pi Foundation levert ook KiCad- en Fritzing-bestanden om aan de slag te gaan met uw eigen ontwerpen. Het is interessant om te zien dat u een referentie-ontwerp krijgt dat niet van de Raspberry Pi Pico is, maar een demonstratie van hoe u uw eigen board met de RP2040 kunt maken. Als u hardware ontwerpt, weet u dat bij de meeste andere microcontrollers geen speciale aandacht aan de pinout is besteed. Maar met de RP2040 zullen PCB-ontwerpers de Raspberry Pi Foundation dankbaar zijn voor het in groepen combineren van bij elkaar horende pinnen, die gemakkelijk te routeren zijn naar externe hardware. **Figuur 11** toont de pinout; u kunt duidelijk zien dat er ook aan het hardware-ontwerp is gedacht.

Op het moment van schrijven zijn zowel de handleidingen als de software nog niet definitief, dus er kunnen nog kleine wijzigingen of toevoegingen komen. Nietemin ziet u duidelijk dat dit een Raspberry Pi-product is met documentatie die van meet af aan klaar en volledig is en dat het open hardware is. Daarnaast betekent dit ook dat er van meet af aan een open SDK wordt geleverd die applicatieontwikkeling op uw Raspberry Pi, uw Linux- of Windows-PC of een Intel-gebaseerde Mac mogelijk maakt. Op dit moment kunt u voor het ontwikkelen van uw eigen applicaties kiezen tussen MicroPython en C/C++.

Programmeren met MicroPython

Veel RP2040-programmeurs zullen de voorkeur geven aan C/C++ (of in de nabije toekomst aan de Arduino IDE, want ondersteuning is al aangekondigd). MicroPython is echter ook een optie als u deze microcontroller wilt programmeren. Allereerst hebt u een Raspberry Pi 4 nodig met Raspberry Pi OS of – volgens de handleiding – een gelijkwaardige op Debian gebaseerde Linux-distributie die op een ander platform draait. U kunt uw eigen Raspberry Pi Pico MicroPython-port builden volgens de instructies in de handleiding [5] of een kant-en-klare binary downloaden via de *Pico getting started* pagina's [6].

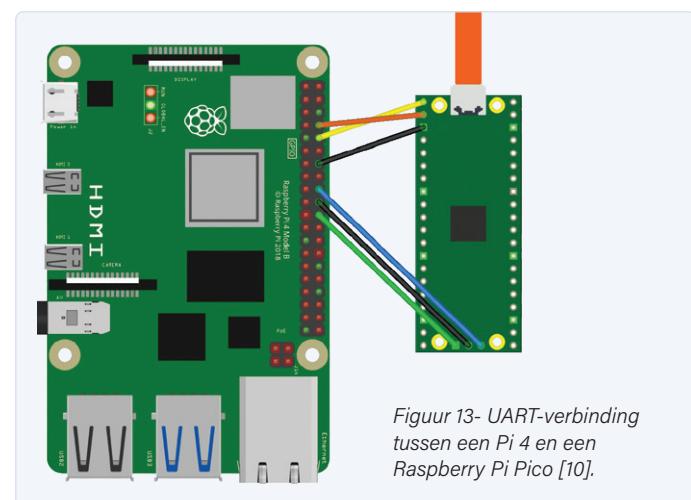
Door het board aan te sluiten op de Raspberry Pi 4 met een micro-USB kabel terwijl de BOOTSEL-knop ingedrukt wordt gehouden, wordt het Pico-board aangemeld als USB-apparaat voor massa-



Figuur 11. Pinout van de RP2040 [9].



Figuur 12. Thonny IDE op een Pi 4.



Figuur 13- UART-verbinding tussen een Pi 4 en een Raspberry Pi Pico [10].

MIST U DE PARAGRAAF OVER LAAG VERMOGEN IN DIT ARTIKEL?

Deze vraag leidde tot wat discussie onder de redacteuren die al met de Raspberry Pi Pico aan het spelen waren. Normaal gesproken is een van de eerste dingen die men merkt bij een nieuwe MCU het lage stroomverbruik, en dat is meestal maar een paar honderd nA of een paar μ A die nodig is voor de slaap-modus. Als u de datasheet van de RP2040 bekijkt, dan ziet u een gemiddeld stroomverbruik van 0,18 mA in deep sleep. Dat klinkt misschien niet zo slecht, en 180 μ A is een waarde waar u mee kunt leven; maar voor de Raspberry Pi Pico zit er een addertje onder het gras. Op het board zit niet alleen de RP2040, maar ook de externe flash en de DC/DC-converter. Terwijl de flash-chip met 50 μ A in standby-mode kan gaan of zelfs kan terugschakelen naar 15 μ A, moet de DC/DC-converter bij zeer lage belastingen werken. Deze resultaten uit de datasheet van de Raspberry Pi Pico komen uit op een standby-stroom van 0,8 mA bij

25 °C. Dit is gewoon iets waar u rekening mee moet houden als u van plan bent om het board voor een langere periode op batterijen te laten werken. Dus de paragraaf over laag vermogen is niet vergeten, maar op een gegeven moment moet men een soort compromis hebben wat betreft functionaliteit, stroomvereisten en beoogde toepassing. De perfecte microcontroller bestaat niet, maar de eerste chip van de Raspberry Pi Foundation biedt een veelbelovende vooruitblik op wat we kunnen verwachten. Met betrekking tot de optimalisatie van het stroomverbruik zal in de komende tijd moeten blijken of en hoe het ontwerp kan worden aangepast om er nog een paar μ A meer uit te halen voor een langere runtime. Dit geldt ook voor de softwareontwikkeling en hoe u uw code kunt optimaliseren, niet alleen voor snelheid maar ook om energie-efficiënt te werken op apparaten met batterijen.

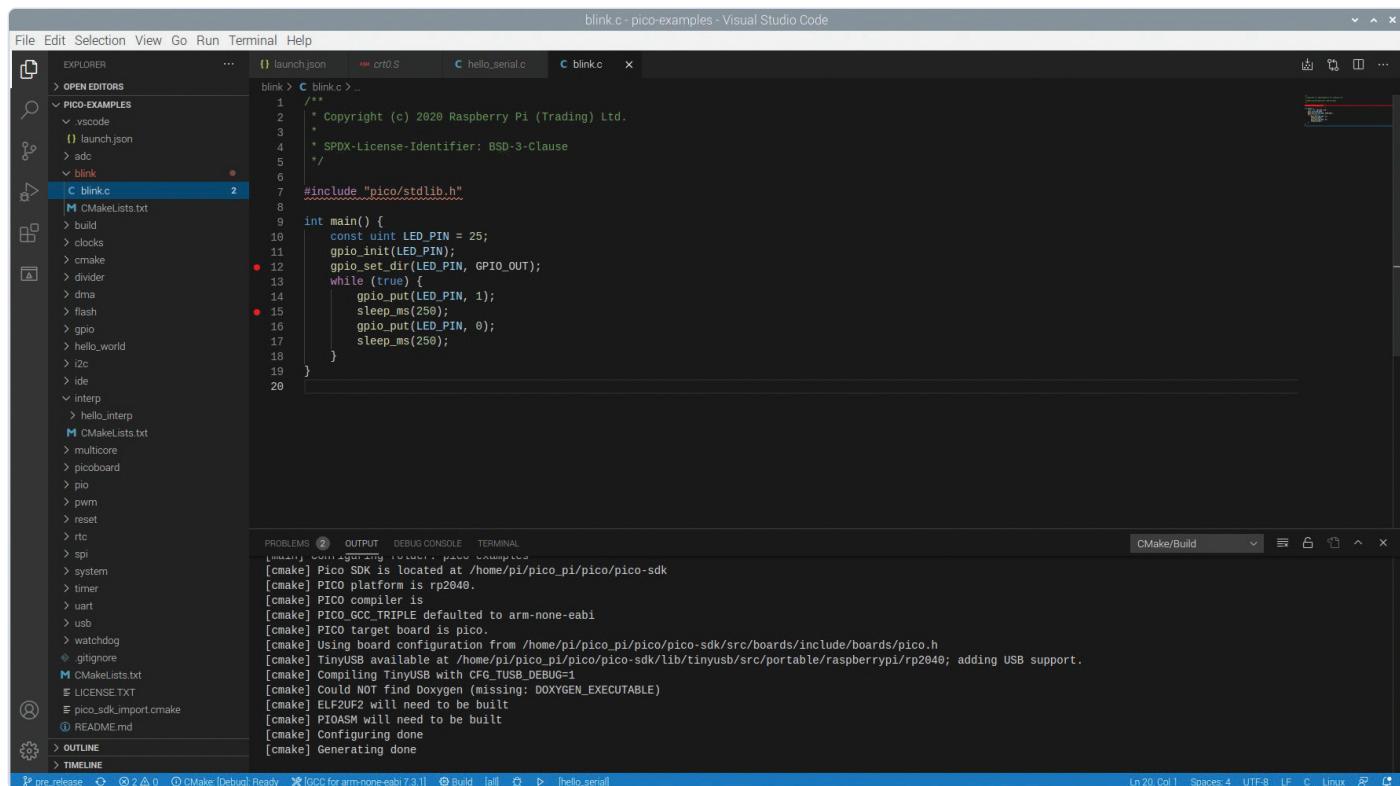
opslag. U kunt het binair bestand dan naar het board slepen om het in het flash-geheugen te zetten. Na deze installatie kunt u verbinding maken met de Raspberry Pi Pico in de MicroPython Interactive Interpreter Mode (ook REPL genoemd) via USB of de UART van de Pico.

Daarnaast werkt de MicroPython-port naar de Raspberry Pi Pico en andere op de RP2040 gebaseerde boards met veelgebruikte Integrated Development Environments (IDE's) zoals Thonny (**figuur 12**). De handleiding bevat instructies voor het installeren en configureren van deze IDE voor het programmeren van de Pi Pico in MicroPython.

Bent u geïnteresseerd in MicroPython? In het boek *Get Started with MicroPython on Raspberry Pi Pico* (G. Halfacree en B. Everard, Raspberry Pi Foundation 2021) leert u hoe u MicroPython kunt gebruiken om programma's te schrijven en hardware aan te sluiten om uw Raspberry Pi Pico te laten samenwerken met de wereld om hem heen. (Zie het kader **Gerelateerde producten**).

C/C++ ontwikkeling op een Pi met een Pi

Uit de uitgebrachte beschrijving en tools blijkt dat programmeren met C/C++ bedoeld is voor op een Raspberry Pi 4. Het opzetten van een ontwikkelomgeving op andere systemen staat beschre-



Figuur 14. Visual Studio Code IDE op een Pi 4.

ven in de handleiding [7], maar voor de Raspberry Pi kunt u een handig script downloaden [8] dat alle stappen voor zijn rekening neemt die nodig zijn voor de installatie. De verbinding tussen de Raspberry Pi 4 en de Raspberry Pi Pico gaat volgens **figuur 13**. In deze opstelling zal de Raspberry Pi 4 ook fungeren als debug-interface met een gepatchte versie van OpenOCD. Als u graag een PC of Mac gebruikt, kunt u een debug-probe bouwen met een tweede Raspberry Pi Pico zoals beschreven in de handleiding.

Als u liever een IDE gebruikt dan alleen programmeren in een shell, kunt u Visual Studio Code (**figuur 14**) installeren via het bijgeleverde script, dan hebt u een IDE om mee te beginnen. Op het moment van schrijven konden de voorbeelden behoorlijk goed worden gevolgd, maar de projectgenerator om nieuwe projecten te starten zonder van alles handmatig op een commandoregel te moeten doen was nog niet helemaal klaar. Zelfs als u bij de lancering wat kleine problemen zou kunnen ervaren, zoals we hebben gezien bij de Raspberry Pi 4, zullen er na verloop van tijd zeker verbeteringen komen naarmate meer gebruikers de meegeleverde software gaan gebruiken en bugs melden.

Voor de RP2040 staat een uitgebreide set bibliotheken en voorbeelden klaar om u op weg te helpen met de chip. U hoeft dus niet zelf drivers te schrijven voor elk onderdeel van de chip. Ook is voor de USB de TinyUSB-bibliotheek opgenomen die is uitgebreid om de RP2040 te ondersteunen, zodat u snel kunt instappen. Het mooie is dat u, omdat alle code open is, in de bestanden kunt duiken en kunt bestuderen hoe alles in elkaar zit. Als u meer gewend bent aan een framework dat compatibel is met Arduino Wiring: dit is op het moment van schrijven nog niet uitgebracht, maar net als voor andere niet-Arduino platforms die interessant zijn, zal dat slecht een kwestie van tijd zijn.

Als u een PC of Mac gebruikt en wilt beginnen met programmeren, kunt u het beste een virtuele machine gebruiken om de ontwikkelomgeving op te zetten. Als u dan op sommige punten problemen ondervindt, kunt u terugschakelen naar een opgeslagen snapshot om in korte tijd weer een goed werkend systeem te hebben.

Nog wat gedachten over de Pico

Hebt u microcontrollers links laten liggen omdat ze te duur lijken of minder goed gedocumenteerd? Nu hebt u daarvoor geen excus meer. Koop een paar Raspberry Pi Pico's nu ze nog goed verkrijgbaar zijn en begin met programmeren. Voor een prijs van € 5 per stuk

en met alle ondersteuning en documentatie van de Raspberry Pi Foundation is het niet de vraag of u er een moet kopen, maar veeleer hoeveel Pico-boards u moet kopen. Dankzij de beschikbaarheid van datasheets, programmeertools, voorbeelden en laagdrempelige handleidingen is het werken met een Pico heel plezierig. Hij is niet voorbehouden aan ervaren ontwikkelaars; hij kan worden gebruikt om kinderen en studenten te onderwijzen. Als u nieuwe projecten start met het board, laat het ons dan weten en deel uw ervaringen. We zijn benieuwd waar u mee op de proppen komt. 

21045-03

Vragen of opmerkingen?

Hebt u vragen of opmerkingen naar aanleiding van dit artikel?
Stuur een e-mail naar de auteurs via
mathias.claussen@elektor.com.

Een bijdrage van

Tekst: **Mathias Claußen** en
Luc Lemmens

Redactie: **Jens Nickel**

Vertaling: **Jan Mulder**
Layout: **Giel Dols**



GERELATEERDE PRODUCTEN

➤ **Raspberry Pi Pico microcontrollerboard**
www.elektor.nl/rpi-pico-board

➤ **Get Started with MicroPython on Raspberry Pi Pico**
www.elektor.nl/micropython-on-rpi-pico

➤ **Raspberry Pi 4 B (8 GB RAM)**
www.elektor.nl/rpi4b8

➤ **Officiële EU-voeding voor Raspberry Pi 4 (zwart)**
www.elektor.nl/eu-power-rpi4-blck

➤ **Officiële HDMI-kabel voor Raspberry Pi 4 (zwart, 1 m)**
www.elektor.nl/hdmi-rpi4-b-1m

WEB LINKS

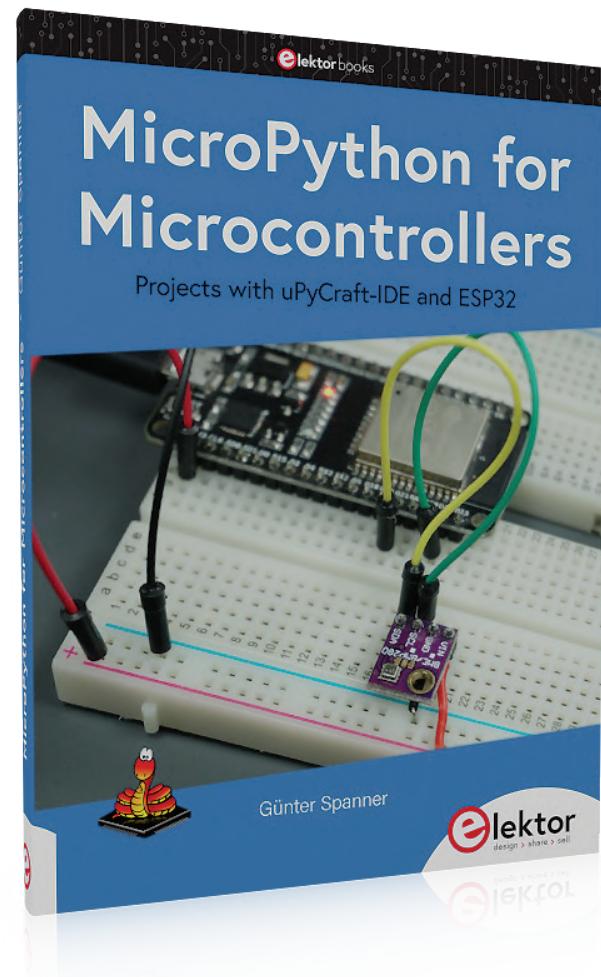
- [1] [C. Valens, "Raspberry Pi Pico Review", Elektor.TV, 1/21/2021: <https://youtu.be/ijn-QDAgZss>](https://youtu.be/ijn-QDAgZss)
- [2] [C. Valens, "Now Playing at a Theatre Near You: RP2040 in Raspberry Pi Pico", ElektorMagazine.com, 1/22/2021: \[www.elektormagazine.com/news/now-playing-rp2040-raspberry-pi-pico\]\(http://www.elektormagazine.com/news/now-playing-rp2040-raspberry-pi-pico\)](http://www.elektormagazine.com/news/now-playing-rp2040-raspberry-pi-pico)
- [3] [Raspberry Pi Pico datasheet: \[https://datasheets.raspberrypi.org/pico/pico_datasheet.pdf\]\(https://datasheets.raspberrypi.org/pico/pico_datasheet.pdf\)](https://datasheets.raspberrypi.org/pico/pico_datasheet.pdf)
- [4] [RT6150 datasheet: \[www.richtek.com/assets/product_file/RT6150A=RT6150B/DS6150AB-04.pdf\]\(http://www.richtek.com/assets/product_file/RT6150A=RT6150B/DS6150AB-04.pdf\)](http://www.richtek.com/assets/product_file/RT6150A=RT6150B/DS6150AB-04.pdf)
- [5] [Pico MicroPython manual: \[https://datasheets.raspberrypi.org/pico/sdk/pico_python_sdk.pdf\]\(https://datasheets.raspberrypi.org/pico/sdk/pico_python_sdk.pdf\)](https://datasheets.raspberrypi.org/pico/sdk/pico_python_sdk.pdf)
- [6] [Pico Getting Started: \[www.raspberrypi.org/documentation/pico/getting-started/\]\(http://www.raspberrypi.org/documentation/pico/getting-started/\)](http://www.raspberrypi.org/documentation/pico/getting-started/)
- [7] [C/C++ setup: \[https://datasheets.raspberrypi.org/pico/sdk/pico_c_sdk.pdf\]\(https://datasheets.raspberrypi.org/pico/sdk/pico_c_sdk.pdf\)](https://datasheets.raspberrypi.org/pico/sdk/pico_c_sdk.pdf)
- [8] [IDE setup script: \[https://github.com/raspberrypi/pico-setup/blob/master/pico_setup.sh\]\(https://github.com/raspberrypi/pico-setup/blob/master/pico_setup.sh\)](https://github.com/raspberrypi/pico-setup/blob/master/pico_setup.sh)
- [9] [RP2040 datasheet: \[https://datasheets.raspberrypi.org/rp2040/rp2040_datasheet.pdf\]\(https://datasheets.raspberrypi.org/rp2040/rp2040_datasheet.pdf\)](https://datasheets.raspberrypi.org/rp2040/rp2040_datasheet.pdf)
- [10] [Pico Getting Started datasheet: \[https://datasheets.raspberrypi.org/pico/getting_started_with_pico.pdf\]\(https://datasheets.raspberrypi.org/pico/getting_started_with_pico.pdf\)](https://datasheets.raspberrypi.org/pico/getting_started_with_pico.pdf)

MicroPython voor microcontrollers

technieken voor kleine displays

Dr. Günter Spanner (Duitsland)

Dit is een deel van een hoofdstuk uit het boek MicroPython for Microcontrollers (Elektor 2021) van Günter Spanner. Het materiaal dat hier is afgedrukt heeft betrekking op de geslaagde combinatie van het populaire ESP32-microcontrollerboard met een SSD1306 OLED display onder gebruikmaking van de programmeertaal MicroPython. Ontdek de kracht en het gebruiksgemak van MicroPython, ga programmeren en maak prototypes van een ECG-display en een digitale klok.



Als het om tests en eenvoudige toepassingen gaat, voldoet de console volledig voor het weergeven van tekst en getallen. Deze methode heeft echter het grote nadeel dat er altijd een PC of in ieder geval een laptop bij nodig is. Bij permanent gebruik gedurende dagen of weken veroorzaken deze apparaten onnodig elektriciteitsverbruik. Bovendien is het niet altijd wenselijk of praktisch om voor elke µC-applicatie een extra computer te gebruiken. Als het alleen een kwestie is van het weergeven van een tijd of de meetwaarden van klimaatsensoren, dan is het veel beter om een apart klein display op de controller aan te sluiten. Een veelgebruikt display is de SSD1306. Dit meet slechts 0,96 inch (2,4 cm) en biedt een resolutie van 128 × 64 pixels.

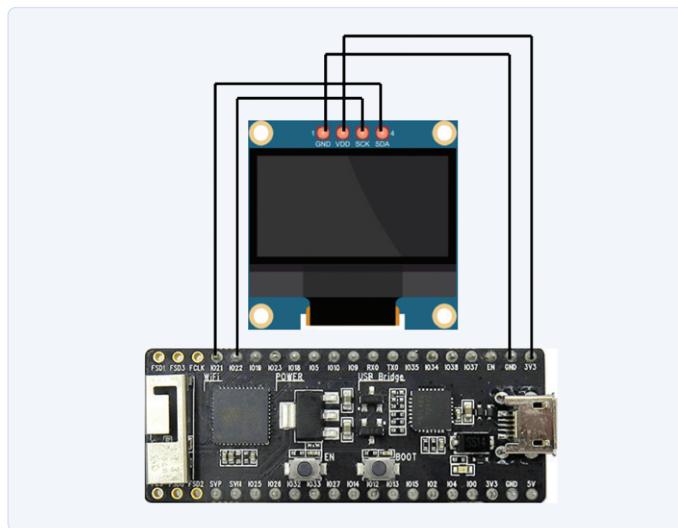
MicroPython wordt standaard geleverd met een stuurprogramma voor dit displaytype. Dit stuurprogramma is al geladen wanneer de bestanden worden overgebracht naar de ESP32. Het stuurprogramma maakt het mogelijk tekst en numerieke gegevens weer te geven, alsmede eenvoudige graphics. Het SSD1306-display heeft een interne RAM en een ingebouwde oscillator. Dit betekent dat het geen externe componenten nodig heeft. Bovendien beschikt het over een helderheidsregeling met 256 instelbare niveaus.

De belangrijkste kenmerken van het SSD1306-display zijn:

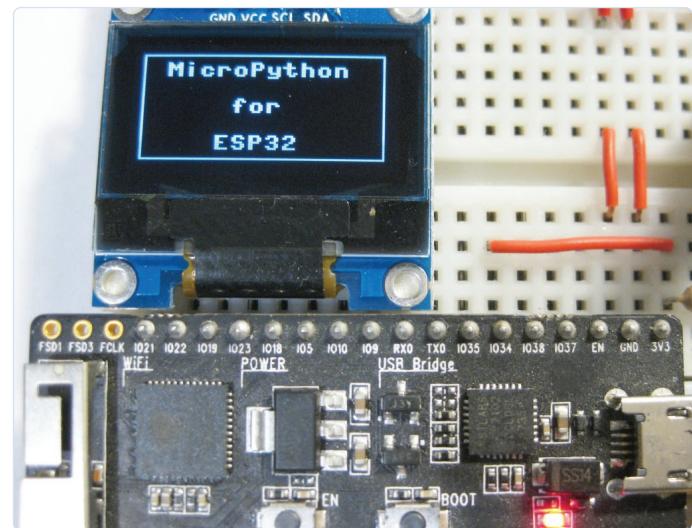
- resolutie 128 × 64 dot matrix;
- voedingsspanning 1,65...3,3 V;
- temperatuurbereik -40 °C tot +85 °C;
- geïntegreerde 128 x 64-bit SRAM displaybuffer;
- continue scroll-functie, zowel horizontaal als verticaal;
- on-chip oscillator.

OLED-displays hebben geen achtergrondverlichting nodig omdat elke pixel licht kan uitstralen. Daarom is deze variant ook bij ongunstige lichtomstandigheden goed afleesbaar. Bovendien is het contrast aanzienlijk hoger in vergelijking met LCD's. Omdat een pixel alleen energie verbruikt als hij daadwerkelijk oplicht, zijn OLED-schermen zeer energiezuinig.

De eenvoudigste versies van SSD1306-boards hebben slechts vier aansluitpinnen. Dat is echter genoeg om het display via de I²C-bus te besturen. Andere varianten hebben reset-pinnen of een extra SPI-interface. Voor de meeste toepassingen is de eenvoudige versie



Figuur 1. SSD1306-display aangesloten op het ESP32-microcontrollerboard.



Figuur 2. SSD1306-display met tekstuele en grafische uitvoer.

echter voldoende. De onderstaande tabel toont alle benodigde aansluitingen.

OLED-pin	ESP32
VDD	3V3
GND	GND
SCK	GPIO 22
SDA	GPIO 21

Het bijbehorende bedradingsschema is afgebeeld in **figuur 1**. Het volgende script geeft een tekstbericht en een eenvoudig grafisch element in de vorm van een kader op het display weer:

```
# SSD1306_DEMO.py

from machine import Pin, I2C
from ssd1306 import SSD1306_I2C

i2c=I2C(-1,scl=Pin(22),sda=Pin(21))
# I2C Pin assignment
oled_width=128
oled_height=64
oled = SSD1306_I2C(oled_width, oled_height, i2c)
lin_hight = 9
col_width = 8
def text_write(text,lin,col):
    oled.text(text,col*col_width,lin*lin_hight)

oled.fill(0)
text_write("MicroPython", 1, 2)
text_write("for", 3, 6)
text_write("ESP32", 5, 5)

oled.rect(5, 5, 116, 52, 1)
oled.show()
```

Figuur 2 toont de resulterende uitvoer. Om het display aan te sturen, moeten vervolgens de benodigde modules worden geïmporteerd. Zoals al opgemerkt zijn de bibliotheken *machine* en *SSD1306* normaal gesproken al beschikbaar als standaardbibliotheeken. Indien nodig kan echter een *ssd1306.py*-bestand afzonderlijk in het board worden geladen. De pinspecificatie voor de I²C-bus verloopt via:

```
i2c = I2C (-1, scl = Pin (22), sda = Pin (21))
```

Het aantal beschikbare pixels in de aangesloten module wordt vastgelegd met behulp van deze variabelen:

```
oled_width = 128
oled_height = 64
```

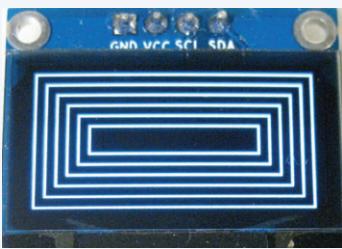
De parameter '-1' geeft aan dat de gebruikte module geen reset- of interruptpinnen heeft. Met deze informatie kan een *SSD1306_I2C* object met de naam *oled* worden gemaakt. Hier worden de eerder gedefinieerde gegevens gekopieerd:

```
oled = ssd1306.SSD1306_I2C (oled_width, oled_height,
                             i2c)
```

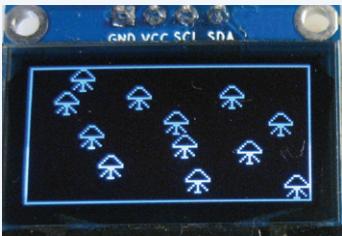
Het display is nu klaar voor gebruik. De functie *text()* wordt gebruikt om informatie weer te geven op het display. Met de methode *show()* wordt het display bijgewerkt. De functie *text()* accepteert de volgende argumenten:

- bericht (type String);
- X- en Y-positie van het tekstveld in pixel-eenheden;
- optionele tekstkleur: 0 = zwart (donker) en 1 = wit (licht).

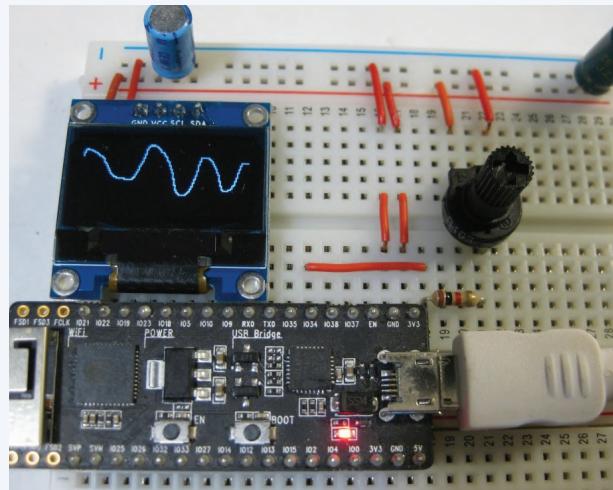
De volgende instructie geeft een bericht in witte of blauwe kleur op een donkere achtergrond uit. De tekst begint op positie x = 0 en y = 0:
`oled.text ('MicroPython!', 0, 0)`



Figuur 3. Concentrische frames op het OLED-display getekend.



Figuur 4. Grafische elementen op het OLED-display.



Figuur 5. Gegevensuitvoer op het OLED-display.

Listing 1: SSD1306 Bitmap Demo.

```
# SSD1306_bitmap_DEMO.py

from machine import Pin, I2C
import ssd1306
import urandom

i2c = I2C(-1, scl=Pin(22), sda=Pin(21))
oled_width = 128
oled_height = 64
oled = ssd1306.SSD1306_I2C(oled_width, oled_height,
    i2c)

# frame
oled.hline(0, 0, oled_width-1, 1)
oled.hline(0, oled_height-1, oled_width-1, 1)
oled.vline(0, 0, oled_height-1, 1)
oled.vline(oled_width-1, 0, oled_height, 1)
oled.show()
```

```
ICON = [
    [0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0],
    [0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0],
    [0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0],
    [0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
    [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
    [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0],
    [0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0],
    [0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0],
    [0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0],
]
for n in range(12):
    xofs = urandom.randint(1, oled_width-12)
    yofs = urandom.randint(1, oled_height-12)
    for y, row in enumerate(ICON):
        for x, c in enumerate(row):
            oled.pixel(x+xofs, y+yofs, c)
oled.show()
```

Listing 2: Rolling ECG Display.

```
# Rolling_ECG_display.py
from machine import Pin, ADC, I2C from time import
    sleep
import ssd1306

i2c = I2C(-1, scl=Pin(22), sda=Pin(21))
oled_width = 128
oled_height = 64
oled = ssd1306.SSD1306_I2C(oled_width, oled_
    height, i2c)

pot = ADC(Pin(34))
pot.atten(ADC.ATTN_11DB) #Full range: 3.3v vMax =
```

```
3.4
dotPos_old = int(oled_height/2)
while True:
    pot_value = pot.read()
    voltage = 0.000816*pot_value + 0.037822 #
        print(voltage)
    dotPos_new = int(voltage/vMax*oled_height)

    oled.line(0, dotPos_new, 0, dotPos_old, 1)
    oled.scroll(1, 0)
    oled.line(0, dotPos_new, 0, dotPos_old, 0)

    dotPos_old = dotPos_new oled.pixel(0, int(oled_
        height/2), 1)
    oled.show()
```

De `show()` methode maakt de wijzigingen zichtbaar op het display. De bibliotheek bevat ook andere handige methoden. De functie `fill(1)` creëert een volledig wit scherm, dat wil zeggen alle pixels zijn verlicht. Met `oled.fill(0)` worden alle pixels zwart of donker gemaakt.

De `pixel()` methode maakt grafische weergaven mogelijk. Hij accepteert de volgende argumenten:

- X-coördinaat: horizontale pixelpositie;
- Y-coördinaat: verticale pixelpositie;
- pixelkleur: 0 = zwart, 1 = wit.

Zo kan een enkele witte pixel in de linkerbovenhoek worden gemaakt:

```
oled.pixel(0, 0, 1)
```

Met behulp van `oled.invert(True)` worden de OLED-kleuren geïnverteerd. Wit wordt zwart en omgekeerd. Om terug te keren naar de originele kleuren, kunt u `oled.invert(False)` gebruiken.

Grafische weergave

Naast de pixelinstructies zijn er ook andere grafische commando's beschikbaar. Met behulp van `.hline()` of `.vline()` kunnen horizontale en verticale lijnen worden getekend. De XY-startpositie evenals de lijnlengte en kleur worden gespecificeerd.

Het volgende kleine programma tekent bijvoorbeeld concentrische rechthoekige kaders op het display. Het resultaat kan worden bewonderd in **figuur 3**.

```
# SSD1306_frames.py
from machine import Pin, I2C
import ssd1306

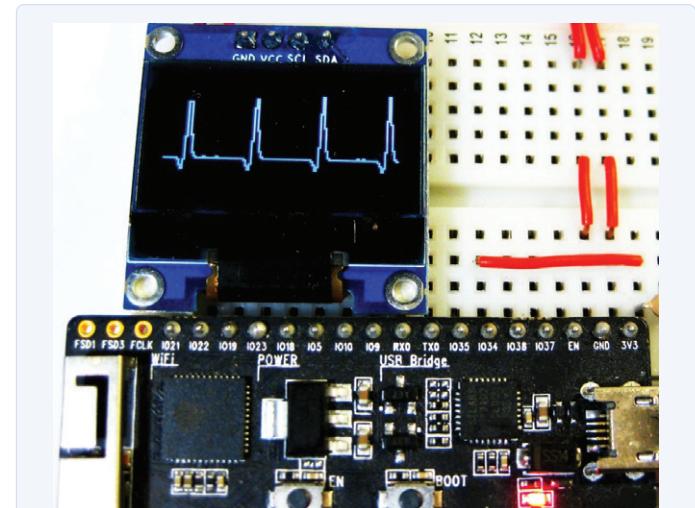
i2c = I2C(-1, scl=Pin(22), sda=Pin(21))
oled_width = 128
oled_height = 64
oled = ssd1306.SSD1306_I2C(oled_width, oled_height,
    i2c)

for n in [0,5,10,15,20,25]:
    oled.hline(n, n, oled_width-1-2*n, 1-2*n)
    oled.hline(n, oled_height-1-n, oled_width-1-2*n,
        1-2*n)
    oled.vline(n, n, oled_height-1-2*n, 1-2*n)
    oled.vline(oled_width-1-n, n, oled_height-2*n, 1-2*n)
oled.show()
```

Diagonalen worden getekend met `.line(x1, y1, x2, y2, c)` tussen de twee punten (x_1, y_1) en (x_2, y_2). De parameter `c` bepaalt de kleur van de getekende lijn. Voor eenvoudige grafische afbeeldingen kunnen bitmaps pixel voor pixel in de weergavebuffer worden geschreven. Het programma in **listing 1** toont een voorbeeld, resulterend in de weergave van **figuur 4**.

OLED-display als plotter

Naast bitmaps kunnen ook meetwaarden grafisch worden weergegeven op het OLED-display. Dit betekent dat u niet langer afhankelijk bent van de plotterfunctie in Thonny (de IDE die voor MicroPython wordt gebruikt) maar ook stand-alone apparaten kunt gebruiken.



Figuur 6. Elektrocardiogram op het display.

ken om grafische uitvoer weer te geven. De software in **listing 2** toont een doorlopende weergave zoals bekend van een ECG-monitor in een ziekenhuis.

Op de ADC-ingang 34 kan een potentiometer worden aangesloten voor het registreren van de meetwaarde. Na het starten van het programma worden de spanningswaarden continu op het display weergegeven. Dit werkt dankzij de ingebouwde scroll-functie. Met behulp van de instructie

```
oled.scroll(1, 0)
```

wordt de volledige scherminhoud één pixel verplaatst. Als u geen losse pixels maar een doorlopende curve wilt opnemen, moet u de punten met lijnen verbinden. Hiervoor zijn twee variabelen nodig:

```
dotPos_old and dotPos_new
```

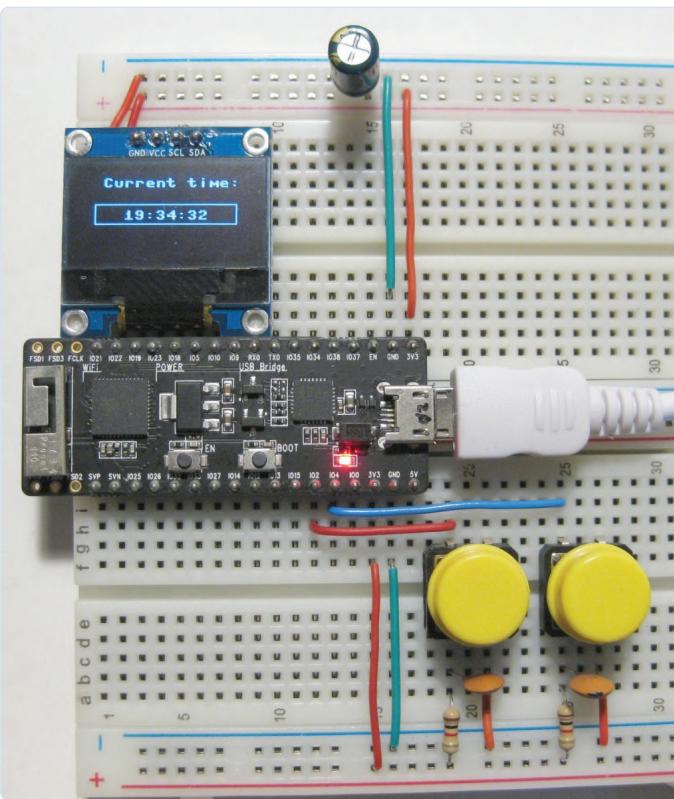
Hiermee wordt een lijn getekend tussen de huidige en de laatst gemeten waarde. Vervolgens wordt de scherminhoud met één pixel verschoven. Ten slotte wordt de nieuwe positie overgebracht naar het buffergeheugen:

```
dotPos_old = dotPos_new
```

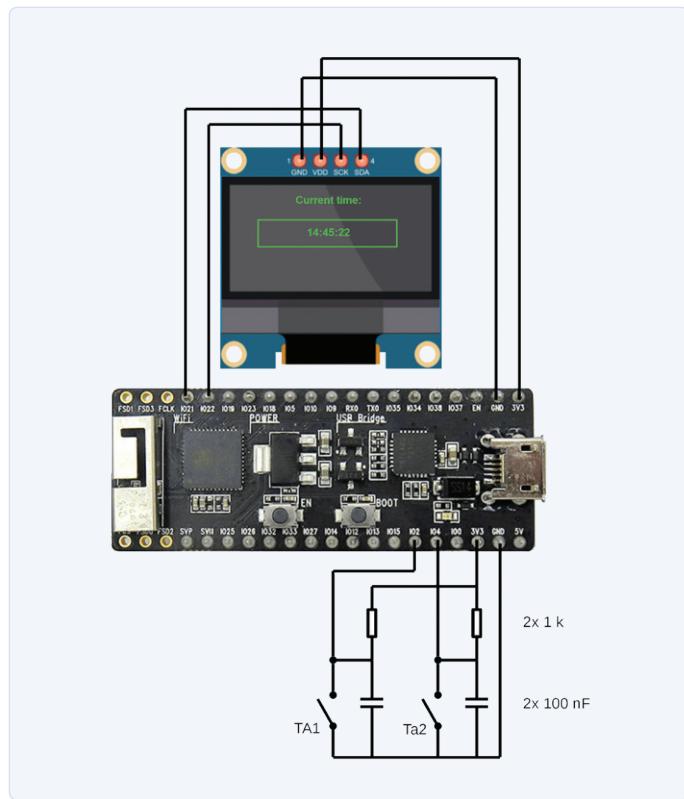
en het spel begint opnieuw. **figuur 5** toont een voorbeeld van continu veranderende potentiometerwaarden. Als u een ECG-versterker hebt, kunt u op deze manier ook de elektrische signalen van het menselijk hart opnemen. Het resultaat is het typische elektrocardiogram dat we kennen van de intensive care-afdelingen in ziekenhuizen (**figuur 6**).

Digitale klok met OLED-display

Een andere nuttige toepassing is het tonen van de tijd. Dit verandert de ESP32 samen met de SSD1307 in een praktische digitale klok. De functie `time()` van de tijdmodule geeft het aantal seconden weer sinds het inschakelen of resetten van het board. Met behulp van een variabele



Figuur 7. Digitale klok met OLED-display.



Figuur 8: Schema van de in MicroPython geprogrammeerde digitale klok.

Listing 3. ESP32/SSD1306-klok in MicroPython.

```
# setable_clock.py

from machine import Pin,I2C
import time
from ssd1306 import SSD1306_I2C

i2c = I2C(-1,scl=Pin(22),sda=Pin(21))
oled_width=128
oled_height=64
oled = SSD1306_I2C(oled_width,oled_height,i2c)

time_offset=20*3600+00*60+0 # hh:mm:ss
lin_height=5
col_width=8

def handle_interrupt_min(pin):
    global time_offset
    time_offset+=60
    time.sleep(.2)

def handle_interrupt_hr(pin):
    global time_offset
    time_offset+=3600
    time.sleep(.2)

button_min = Pin(4, Pin.IN)
```

```
button_min.irq(trigger=Pin.IRQ_RISING,
    handler=handle_interrupt_min)

button_hr = Pin(2, Pin.IN)
button_hr.irq(trigger=Pin.IRQ_RISING,
    handler=handle_interrupt_hr)

def text_write(text, lin, col=0):
    oled.text(text, col*col_width, lin*lin_height)

def time_text(time):
    secs=time%60
    mins=(time//60)%60
    hours=(time//3600)%24
    return "{:02d}:{:02d}:{:02d}".
        format(hours,mins,secs)

def show():
    oled.fill(0)
    text_write("Current time:",1,2)
    current_text = time_text(current_time)
    text_write(current_text,6,4)
    oled.rect(10,25,108,16,1)
    oled.show()

while True:
    current_time=time_offset+time.time()
    show()
```

```

time_offset=20*3600+00*60+0 # hh:mm:ss
kan een starttijd kan worden gespecificeerd. In dit geval begint de
klok om 20:00:00. De routine time_text():
def time_text(time):
secs=time%60
mins=(time//60)%60
hours=(time//3600)%24
return "{:02d}:{:02d}:{:02d}".format(hours,mins,secs)

```

zet de opeenvolgende seconden om in uren, minuten en seconden, dus in de gebruikelijke hh:mm:ss-weergave, zoals in **figuur 7**. De klok kan worden ingesteld met behulp van twee drukknoppen (Ta1 en Ta2) op poorten 02 en 04, aangesloten zoals in **figuur 8**. Over de knoppen zijn 100nF-condensatoren aangesloten ter ontdeerring. De 1kΩ-weerstanden dienen als pull-ups. De bijbehorende interruptroutines

```

def handle_interrupt_min(pin):
global time_offset
time_offset+=60
time.sleep(.2)

en
def handle_interrupt_hr(pin):
global time_offset
time_offset+=3600
time.sleep(.2)

```

zorgen ervoor dat bij elke druk op de toets de minuten (60 seconden) of uren (3600 seconden) met één worden opgehoogd. Het volledige programma voor de digitale klok is in **listing 3** afgedrukt. 

200581-03

Opmerking van de redactie: een Engelstalige versie van het boek is beschikbaar op www.elektor.nl/micropython-for-microcontrollers. Het softwarepakket bij het boek is gratis beschikbaar op de projectpagina bij dit artikel [1].

Vragen of opmerkingen?

Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur of naar de redactie van Elektor, via redactie@elektor.com

Een bijdrage van

Tekst en illustraties:
dr. Günter Spanner

Redactie: **Jan Buiting**

Vertaling: **Hans Adams**

Layout: **Giel Dols**

WEBLINK

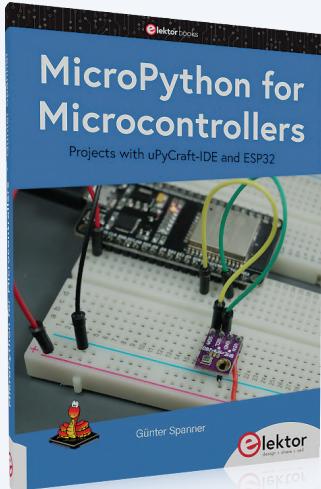
[1] Software-archief bij het boek **MicroPython for Microcontrollers**: www.elektrormagazine.nl/200581-03



GERELATEERDE PRODUCTEN

MicroPython for Microcontrollers

Projects with uPyCraft-IDE and ESP32



De programmeertaal Python heeft de afgelopen jaren een enorme opleving doorgemaakt en diverse systemen (zoals de Raspberry Pi) hebben bijgedragen aan de populariteit ervan. Maar Python wordt ook op grote schaal gebruikt op andere gebieden, zoals kunstmatige intelligentie en machine learning. En zowel Python als de MicroPython-variant zijn beslist ook goede kandidaten voor gebruik in SoC's (Systems on Chip). Krachtige controllers zoals de ESP32 van Espressif Systems bieden uitstekende prestaties plus WiFi- en Bluetooth-functionaliteit voor een betaalbare prijs. Deze functies hebben

de makerscene snel veroverd. In vergelijking met andere controllers heeft de ESP32 een veel groter flash- en SRAM-geheugen en is de CPU veel sneller. Dit boek werkt naar de toepassing van de moderne single-chip systemen toe. Naast de technische achtergrond ligt de nadruk vooral op MicroPython zelf. Na een kennismaking met de taal worden de verworven programmeervaardigheden direct in de praktijk gebracht. De afzonderlijke projecten zijn zowel geschikt voor gebruik in het laboratorium als voor alledaagse toepassingen. Naast het daadwerkelijke leereffect ligt de focus ook op het plezier van het bouwen van complete en bruikbare apparaten. Door gebruik te maken van breadboards kunnen allerlei soorten schakelingen met weinig moeite worden gerealiseerd, waardoor het testen van zelfgemaakte apparaten een leerzaam plezier wordt.

➤ **Koop het boek (papieren versie)**
www.elektor.nl/micropython-for-microcontrollers

➤ **Koop het e-boek**
www.elektor.nl/micropython-for-microcontrollers-pdf

12-200 V DC/DC-converter voor buizenversterkers

Ton Giesberts (Elektor)

Dit is een (relatief) nabouwvriendelijke DC/DC-converter voor buizenversterkers, met through-hole componenten (afgezien van één SO-8 IC) met galvanisch geïsoleerde hoogspanningsuitgang. De transformator maken we zelf; de wikkelverhouding bepaalt de uitgangsspanning. Ingang is een 12 VDC/5 A netadapter, het maximum uitgangsvermogen bedraagt 50 W.



Het hoogspanningsgedeelte van een buizenversterker is een belangrijk maar ook gevaarlijk onderdeel van de schakeling. Aan de ingang hebben we de – potentieel dodelijke – netspanning; aan de uitgang een hoge gelijkspanning die ook tamelijk 'schokkend' kan zijn.

De eenvoudigste van de netspanning afgeleide voeding bestaat uit een goed geïsoleerde transformator, een gelijkrichter en een afvlakcondensator. De uitgangsspanning is afhankelijk van de netspanning, de transformator, de spanningsval over de diode(s), de belasting en de resulterende rimpelspanning. De hoogspanning in buizenversterkers wordt vaak op deze manier gemaakt. Een nadeel daarbij is de vaste wikkelverhouding van commerciële transformatoren, waardoor de spanning ofwel te laag ofwel te hoog is, hoewel dit in veel ontwerpen niet echt een probleem is.

In dit project gebruiken we een 12V-netadapter en een op maat gemaakte DC/DC-converter om de lage gelijkspanning op te krikken tot 200 V. Op deze manier wordt directe aansluiting op de netspanning vermeden en kan de lagere DC-ingangsspanning ook voor de gloeidraden worden gebruikt.

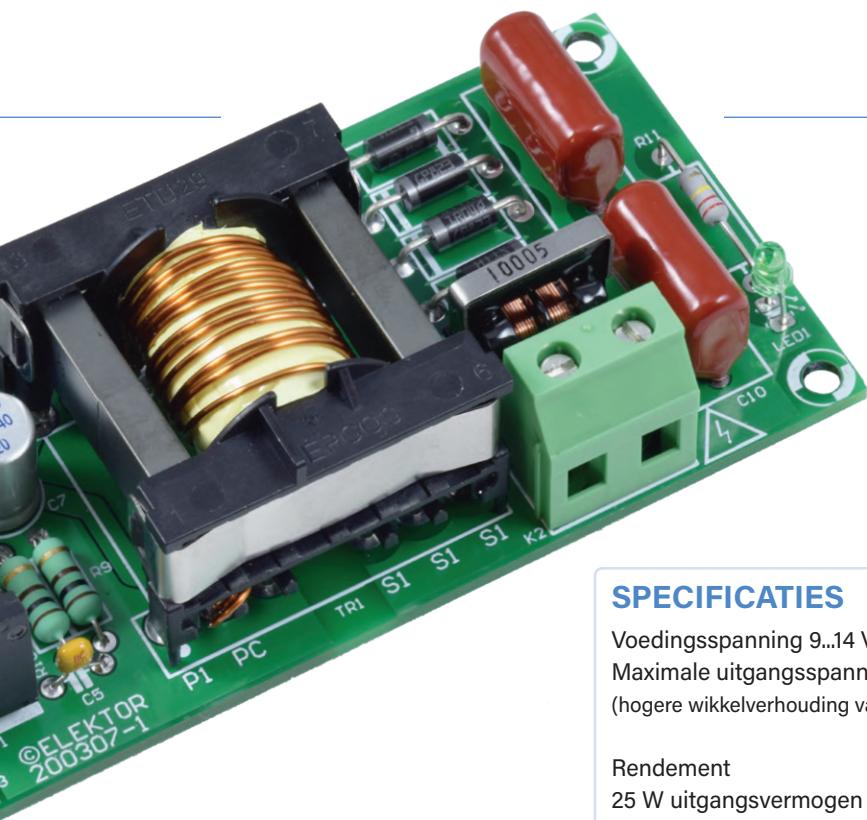
Hier presenteren we een samenvatting van veel uitgebreidere – zowel theoretische als praktische – informatie die de ontwerper heeft verzameld op de Elektor Labs-pagina van dit project. Als u alleen geïnteresseerd bent in wat achtergrondinformatie en de nabouw

van deze 12 VDC/200 VDC-omzetter, biedt dit artikel alles wat u nodig hebt. Maar als u meer wilt weten over de ontwerpoverwegingen die bij het bouwen van een push/pull DC/DC-converter een rol spelen en de (transformator)berekeningen die daarbij komen kijken, volg dan de link bij [1].

De werking van de DC/DC-converter

Het schema van de DC/DC-converter is te zien in **figuur 1**. Om de bouw van dit ontwerp te vergemakkelijken, zijn voor alle onderdelen through-hole componenten gebruikt, behalve voor één IC in SO-8 SMD-behuizing. Dat is een primaire push/pull-oscillator met dode-tijd regeling, de UCC28089 (IC1). De uitgangsdrivers kunnen 1 A opnemen (sink) en 0,5 A leveren (source) en zijn ideaal voor het aansturen van MOSFET's. Het ontbreken van een regellus in het ontwerp betekent dat er geen instabiliteit of ruis optreedt door PWM-regeling. De duty cycle is op maximum ingesteld, met andere woorden: de dode tijd tussen het inschakelen van een van de MOSFET's (T1 of T2) bedraagt een absoluut minimum (zie verderop). Hierdoor wordt de gelijkgerichte secundaire uitgangsspanning bijna tot een gelijkspanning, en is slechts een kleine afvlakcondensator nodig. De UCC28089 heeft ook overstroomdetectie, maar de uitschakeldrempel is relatief hoog, 0,725 V typ. Bij 5 A, waar we voor deze

schakeling de grens hebben gelegd, zou dit neerkomen op $5 \text{ A} \times 0,725 \text{ V} = 3,625 \text{ W}$! De shuntweerstand moet groot zijn en een fors vermogen aan kunnen, in dit geval ten minste 5 W. Bij 12 V, 50 W uitgangsvermogen en 86% rendement zou dit neerkomen op 7% rendementsafname. Een manier om dit verlies in de shunt te verminderen is door een spanningsdeler (R3 en R4) te gebruiken vanaf de voedingsspanning om een kleine offset toe te voeren aan de stroomdetectie-ingang. De oscillatorfrequentie van de UCC28089 kan met instelpot P1 worden ingesteld van ongeveer 200 kHz tot 560 kHz. De schakelfrequentie is de helft van de oscillatorfrequentie, dus instelbaar van 100 kHz tot 280 kHz. Volgens de datasheet bedraagt de ON-weerstand van de TK30A06N1 MOSFET's 15 mΩ max. (V_{GS} 10 V), bijna verwaarloosbaar. Dit type heeft een zeer lage reverse transfer-capaciteit (gate/drain-capaciteit of 'Miller-capaciteit') van 33 pF en snelle schakeltijden (t_{on}/t_{off} 21/28 ns). Bovendien is de ingangscapaciteit van 1050 pF lager dan gebruikelijk. Bij maximaal vermogen is geen extra koeling van de transistoren nodig, hoewel constant maximaal vermogen niet wordt aangeraden. In een push/pull-converter wordt de energie rechtstreeks naar de secundaire zijde overgebracht. Wanneer één schakelaar in geleiding is, is de andere uitgeschakeld en wordt de energie rechtstreeks overgedragen. Tussen



PROJECT-INFO

Tags

voeding, DC/DC-converter, buizenversterker, hoogspanning, through-hole, zelfgemaakte trafo

Niveau

beginners – gevorderden – experts

Tijd

ongeveer 8 uur

Gereedschap

soldergereedschap, kleine punt voor SO-8 solderen

Kosten

ongeveer € 45

SPECIFICATIES

Voedingsspanning 9...14 VDC
Maximale uitgangsspanning 350 V
(hogere wikkelaarshouding van transformator)

Rendement

25 W uitgangsvermogen 88%
50 W uitgangsvermogen 86%

Schakelfrequentie

100/150/280 kHz P1 min/mid/max

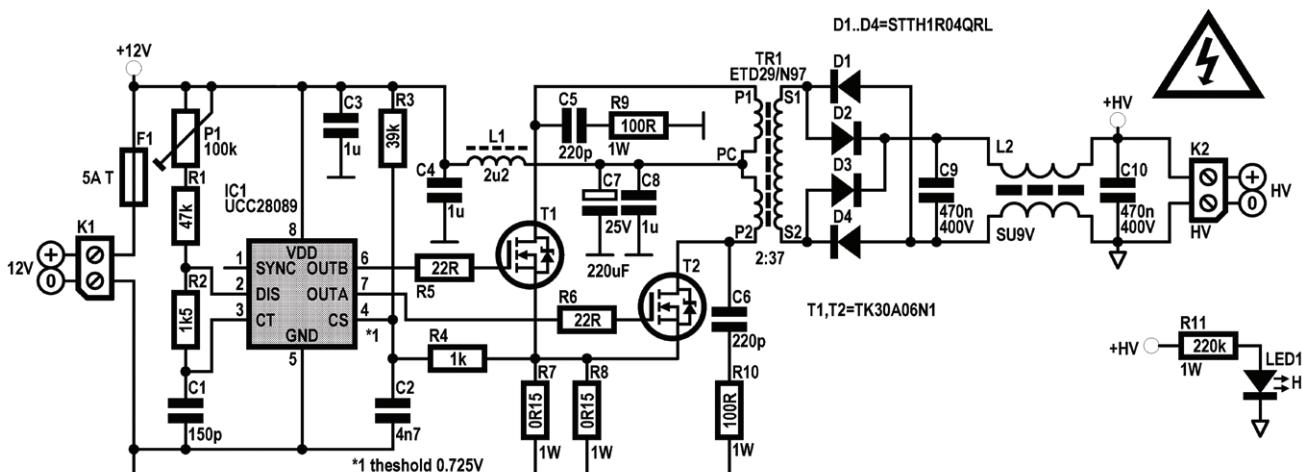
Geen laststroom 12 VDC
300/190/120 mA P1 min/mid/max

Bij een hoge belasting zal de converter niet aanlopen. Deze converter is ontworpen voor buizenversterkers waar de belasting van de hoogspanning zeer laag is bij het inschakelen!

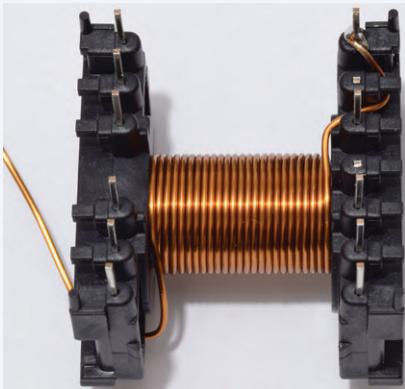
De transformator

Het lastigste onderdeel van deze DC/DC-converter is de transformator; deze moet voor een eerste prototype zelf worden gemaakt. Bij 150 kHz kan het uitgangsvermogen theoretisch meer dan 170 W bedragen bij gebruik van een ETD29-spoelvorm [2] en N97-materiaal, maar in de werkelijke wereld zijn er ook koperverliezen. Er wordt geen luchtspleet gebruikt in de transformator en dit vermindert het aantal windingen voor de primaire en secundaire wikkeling, waardoor de constructie van

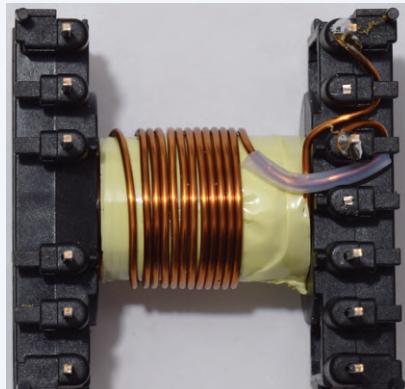
doorbranden, en als de kortsluiting aanhoudt zullen ook de MOSFET's sneuvelen (zelfs bij een kleine overlap!). Om de 12V-voeding te beschermen wordt een 5A-zekering gebruikt. Een voordeel van een push/pull-converter is dat de piekstroom door de schakelaars slechts iets hoger is dan de gemiddelde belastingsstroom aan de primaire zijde (de stroom die de netadapter moet leveren). Kleinere piekstromen in de schakelaars betekent minder geleidingsverliezen en een hoger rendement van de converter.



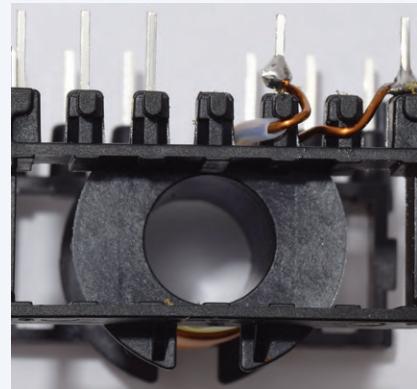
Figuur 1. Schema van de DC/DC-converter.



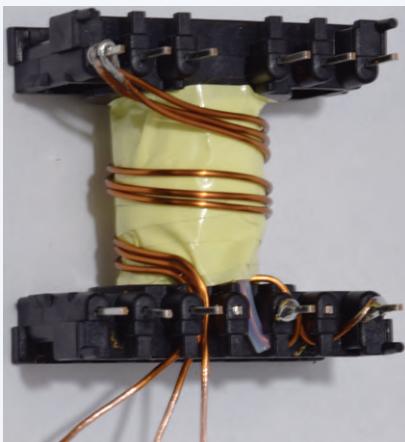
Figuur 2a. Eerste secundaire wikkellaag.



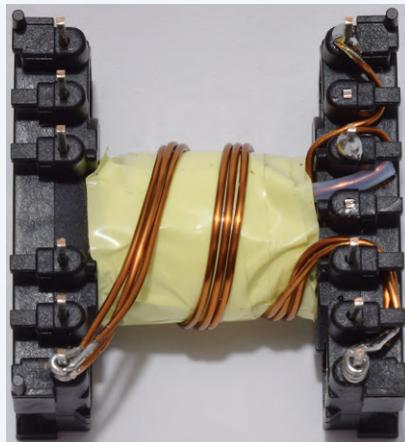
Figuur 2b. Tweede secundaire wikkellaag.



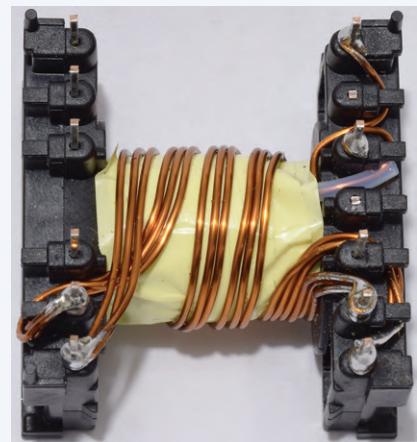
Figuur 2c. Secundaire wikkeling gesoldeerd.



Figuur 2d. Eerste primaire wikkeling, let op de drie parallelle draden.



Figuur 2e. Eerste primaire, vóór het solderen.



Figuur 2f. En de tweede primaire toegevoegd, transformator klaar.

Figuur 2: De bouw van de transformator.

de transformator eenvoudiger is. Geen luchtspleet betekent een hogere inductantie en een betere koppeling tussen de wikkelingen. U kunt dezelfde draad gebruiken voor alle wikkelingen: draad van 0,7 mm maakt het gemakkelijk deze om de spoelhouder te wikkelen en het aantal windingen bij te houden.

Het idee van de schakeling is dat de wikkelverhouding van de transformator de uitgangsspanning bepaalt, in het geval van een

push/pull-converter de verhouding van één primaire wikkeling en de secundaire. In dit project worden twee wikkelingen met elk twee windingen en een enkele secundaire wikkeling (37 windingen) gebruikt. Met een primaire spanning van 12 V zou een wikkelverhouding van 2:37 theoretisch resulteren in een secundaire spanning van 222 V. Aangezien geen enkele transformator ideaal is, zal deze verhouding niet nauwkeurig de

gewenste uitgangsspanning bij een specifieke belasting opleveren. Vergelijken met de hoge uitgangsspanning is de spanningsval over de gelijkrichter en het filter te verwaarlozen. Om wijzigingen aan de voltooide transformator uitvoerbaar te maken, is het logisch om, als de afwijking van de uitgangsspanning groter is dan verwacht, de constructie te beginnen met de primaire wikkelingen, deze dan te isoleren met speciale isolatietape en -kousen. De

EEN BEETJE TRANSFORMATORTHEORIE

De in een spoel geïnduceerde spanning is volgens de wet van Faraday (vereenvoudigd):

$$E = N * B * A / t$$

met:

E = spanning [V]

N = aantal windingen

B = fluxdichtheid [T]

A = oppervlakte van de spoel [m^2]

$t = T/2 = 1/2f$

In een push/pull-converter zijn de spanningen die op de primaire wikkelingen worden aangelegd – nagenoeg – blokgolven. In een halve periode kan de kern per wikkeling gemagnetiseerd worden van B_{max} naar $-B_{max}$ en omgekeerd. De andere wikkeling doet het omgekeerde; in de tijd dat de magnetisatie $2 * B_{max}$ verandert, vinden we voor de geïnduceerde spanning:

$$E = N * 2B_{max} * A * 2f$$

Voor één van de primaire wikkelingen krijgen we dan de volgende vergelijking:

$$NP = EP * 10^4 / (4 * f * B_{max} * A)$$

met oppervlakte A in cm^2 (vandaar de 10^4)

KERNGROOTTE EN MAXIMAAL UITGANGSVERMOGEN

Dit onderwerp lijkt het lastigste te zijn bij het ontwerpen van een DC/DC-converter met een transformator. Op het internet is veel informatie te vinden over het berekenen van transformatoren, zelfs complete calculators die het ontwerp sterk vereenvoudigen. Maar op de meeste sites kan de kerngrootte niet worden berekend maar wordt verwezen naar steeds weer ongeveer dezelfde tabellen voor verschillende kern- en spoelvormen zoals Exx, EExx, EFxx, EFDxx, Elxx, ETDxx en EERxx.

Die tabellen zijn meestal ingedeeld naar het maximaal haalbare vermogen, maar de schakelfrequentie ontbreekt vaak. Om een indruk te krijgen hoe groot de kern moet zijn, kan een formule worden gebruikt die gebruikmaakt van het kernoppervlak-product WaAc (kerndoorsnede maal het voor de wikkelingen beschikbare oppervlak). Maar factoren zoals topologieconstante

(hangt af van het type converter), stroomdichtheid (hangt af van de maximaal toegestane temperatuurstijging) en vultactor moeten in de formule(s) worden ingevoerd. Het is echter geen exacte berekening, er zijn te veel onderlinge afhankelijkheden.

Vele jaren geleden waren bij Block bouwpakketten van transformatoren verkrijgbaar. Eén bouwpakket bevatte een rolletje isolatiefolie, isolatiekousjes, 2 kartonnetjes voor een specifieke luchtspleet, een spoelhouder, 2 halve kernen en 2 clips. Koperlakdraad moest natuurlijk elders worden aangeschaft. In de doos zat ook een folder met de naam "Berechnungsbogen für Schaltnetzteil-Übertrager EB" (rekenblad voor transformatoren voor schakelende voedingen). Hier stonden formules voor de berekening van de minimale kerngrootte, primaire en secundaire wikkelingen en primaire en secundaire draaddiameter.

Het kernmateriaal in die tijd was N27. Het grootste verschil met het huidige N97 is dat de relatieve kernverliezen bij N97 aanzienlijk geringer zijn, 300 kW/m^3 in plaats van 920 kW/m^3 (bij 100 kHz/200 mT/100 °C). De formule om dan de minimale kerngrootte te berekenen luidt:

$$\text{Minimale kerngrootte [mm]} = \\ 4,7 * 10^6 * \frac{\text{maximaal totaal uitgangsvermogen}}{\text{frequentie [W/Hz]}}$$

Met deze formule komt het maximale vermogen bij 150 kHz en een kerngrootte van 5350 mm^3 uit op 170 W, dat bedoelde ik met de opmerking dat de ETD29-kern is groter dan strikt noodzakelijk.

Als iemand weet hoe de formule of een soortgelijke (betere) formule kan worden afgeleid en/of inzicht weet te geven hoe de kerngrootte nauwkeuriger kan worden berekend, deel die kennis dan met ons alstublieft!

secundaire wikkeling is dan de wikkeling aan de buitenkant en het aantal windingen kan in een vroege testopstelling gemakkelijk worden gewijzigd. Deze procedure werd gebruikt om de trafo voor het eerste prototype te maken. In een tweede prototype bleek echter een transformator met de secundaire wikkeling aan de binnenzijde en beide primaire aan de buitenzijde betere eigenschappen te bezitten dan de eerste (zie [1]), zodat hier alleen deze tweede transformator wordt beschreven.

Bouw van de transformator

Om de transformator gemakkelijker te kunnen reproduceren, is gekozen voor de kleinste ETD-versie van de spoelvorm die een grotere kern heeft dan nodig is voor het uitgangsvermogen (60 W of zo, bij een theoretisch rendement van 100%).

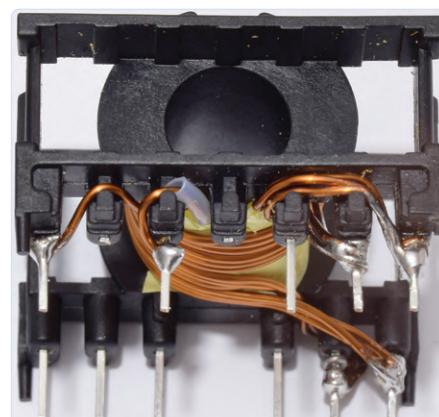
Twee pinnen van de spoelhouder worden afgeknipt (pin 8 en 10 volgens de nummering op de spoelhouder van TDK/Epcos) om de juiste stand van de transformator op de print te verzekeren. Kijk op de print-layout om er zeker van te zijn dat u de juiste pinnen afknijpt. De primaire en secundaire aansluitpinnen zijn op de print goed gescheiden, evenals de uitgangen van de secundaire wikkeling op de print. Oorspronkelijk zitten er 13 pennen op een spoelvorm van het type ETD29, er is al ruimte tussen pin 3 en 4. Een aansluitpunt van de secundaire wikkeling heeft twee mogelijke aansluitingen. Afhankelijk van het aantal windingen en de dikte van de gebruikte koperdraad, kan het aantal benodigde lagen meer dan één zijn en kan het uiteinde van de wikkeling ofwel dichter bij de tegenoverlig-

gende zijde liggen, ofwel aan dezelfde zijde als van waaruit is begonnen. Om deze reden zijn de pennen 4, 5, 6 en 9 op de print doorverbonden. Pen 7 is de andere aansluiting van de secundaire wikkeling. Begin de secundaire wikkeling altijd bij pen 7.

Pinnen 1 en 13 zijn de aansluitingen voor de eerste primaire wikkeling, pinnen 2 en 12 voor de tweede primaire wikkeling. Op de print zijn naast de transformator namen afgedrukt. P1/PC voor de eerste primaire wikkeling, P2/PC voor de tweede primaire wikkeling. PC staat voor de gemeenschappelijke aansluiting van de beide primaire wikkelingen. S1 en S2 zijn de aansluitingen voor de secundaire wikkeling. Voor de secundaire wikkeling, beginnend bij pin 7, worden 24 windingen om de spoelhouder gewikkeld (**figuur 2a**) en met tape geïsoleerd. Wikkel vervolgens de tweede laag met 13 windingen om de in totaal 37 secundaire windingen te voltooien (**figuur 2b**). De tweede verbinding is geïsoleerd met een stukje PTFE-kous, zoals te zien is in **figuur 2c**. Het zou beter zijn geweest om ook de eerste aansluiting van de secundaire wikkeling te isoleren (ontbreekt op de foto's van TR2). Vervolgens wordt de tweede secundaire laag geïsoleerd met tape. **Verhit de draden bij de pinnen van de spoelhouder niet te lang. Hierdoor kan het plastic van de spoelhouder smelten. Daarom moeten de draadeinden goed worden vertind voordat ze om de pennen worden gedraaid!**

Daarna komt de eerste winding van de eerste primaire wikkeling aan pin 1. Gebruik een scherp mes om de isolatie van elke draad te verwijderen en vertin dit uiteinde alvorens het

om een pin te wikkelen. Gebruik een tang om het draadeinde strak rond de pen te knijpen voordat de twee windingen rond de spoelhouder worden gelegd. Maak de draad (iets meer dan) lang genoeg, om de tegenoverliggende pin te bereiken, maar sluit deze nog niet aan. Doe dit voor de volgende twee windingen van deze primaire wikkeling (drie draden parallel, zie **figuur 2d**). Verwijder vervolgens de isolatie van de drie uiteinden. Vertin eerst de uiteinden en knijp ze alle drie om de aansluitpin, zodat de wikkeling optimaal komt te liggen (zo vlak en gelijkmatig mogelijk verdeeld, **figuur 2e**). Herhaal deze procedure voor de tweede primaire wikkeling, te beginnen bij pin 2 (**figuur 2f**). **Figuur 3** toont de transformator met alle wikkelingen. Vergeet niet de kern te monteren!



Figuur 3. Aansluitingen van de voltooide transformator.



ONDERDELENLIJST

Weerstanden:

R1 = 47k
 R2 = 1k5
 R3 = 39k
 R4 = 1k
 R5,R6 = 22 Ω
 R7,R8 = 0,15 Ω, 1W
 (Multicomp Pro MCKNP01SJ015KA10)
 R9,R10 = 100 Ω, 1 W (Multicomp Pro MCKNP01SJ0101A10)
 R11 = 220k, 1 W, 350V
 P1 = instelpotmeter 100k, liggend

Condensatoren:

C1 = 150p, 50 V, steek 5 mm
 C2 = 4n7, 50 V, steek 5 mm
 C3,C4,C8 = 1μ, 50 V, steek 5 mm
 C5,C6 = 220p, 100 V, steek 5 mm
 C7 = 220μ, 25 V, 20 %, polymer-aluminium, steek 3,5 mm, Ø 8 mm, ESR 15mΩ
 A750KS227M1EAAE015 Kemet
 C9,C10 = 470n, 400 V, 5 %, polypropyleen, steek 15 mm (19x9 mm max.)

Spoelen/transformatoren:

TR1 = 2x kern ETD29, N97, spleetloos, B66358G0000X197 TDK/Epcos
 TR1 = spoelvorm, ETD29, B66359W1013T001, TDK/Epcos

TR1 = 2x clip, B66359S2000X000, TDK/Epcos
 L1 = 2,2μH, 10%, 5,6 ARMS, 21mΩ, steek 5 mm, Ø 8,5 mm max., RLB0913-2R2K Bourns
 L2 = common-mode smoorspoel 500μH, 1 A, 0,3 Ω, SU9V-10005 Kemet

Halfgeleiders:

D1,D2,D3,D4 = STTH1R04QRL
 LED = LED groen, 3 mm
 T1,T2 = TK30A06N1, TO-220SIS
 IC1 = UCC28089D, SOIC-8

Overig:

K1 = printkroonsteen 2-polig, steek 5,08 mm
 K2 = printkroonsteen 2-polig, steek 7,68 mm
 F1 = glaszekerling, 5x20 mm, 5 A traag met zekeringhouder 500V/10A en afdekkap
 TR1 = isolatietape, polyesterfilm, 3M 1350 12 MM
 TR1 = koperdraad 0,71 mm voor alle wikkelingen, 49 windingen in totaal, 3 meter
 TR1 = PTFE-beschermkous, binnendiameter 1,02 mm min., Pro Power STFE 18 CLR, 10 cm

Print 200307-1 v1.1

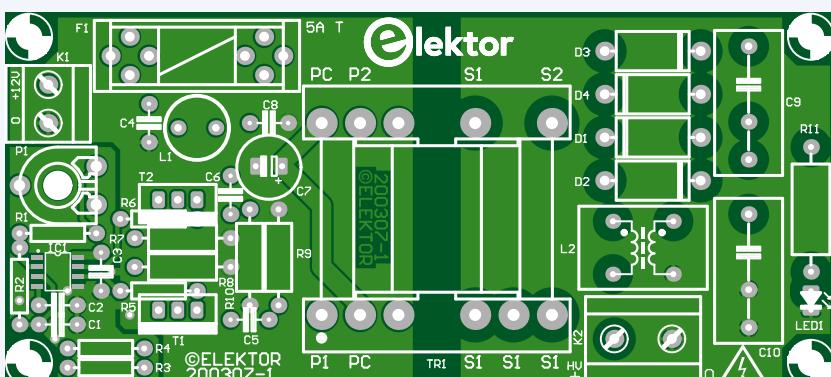
Bouw van de DC/DC-converter

De Gerber- en boorbestanden voor de productie van de print (**figuur 4**) kunnen worden gedownload van [3]. Bestel de print bij uw favoriete leverancier. De print op de foto's van het prototype is versie 1.0, in versie 1.1 is een kleine correctie aangebracht in de vorm van de aanduidingen '+HV' en '0' naast printkroonsteen K2.

Soldeer eerst IC1 op de print, de kleine SMD-chip (SO-8). Om ruimte te sparen zijn de 1W-vermogensweerstanden (R7...R10) aan de primaire zijde kleine versies (weerstandslichaam 10 mm x 3,5 mm); als u nog enkele oudere exemplaren hebt of willekeurige types koopt passen ze misschien niet. Kijk in de BOM voor de referentie van de fabrikant van de weerstanden die we hebben gebruikt. C7 moet een polymeer aluminium type zijn, gebruik geen gewoon elektrolytisch exemplaar omdat dat hoogstwaarschijnlijk zal doorbranden! Het genoemde type (A750KS227M1EAAE015 van Kemet) kan een rimpelstroom van 4,42 A aan bij 100 kHz en heeft een zeer geringe serieverstand van 15 mΩ (100 kHz/20 °C). De montage van de rest van de componenten is vrij eenvoudig. De weerstand voor de LED (R11) die de aanwezigheid van de hoogspanning (HV) aangeeft, moet ten minste een 350V-type zijn en mag een grotere 1W-weerstand zijn. Merk op dat de schakeling niet voldoet aan de Klasse-II-specificaties voor netgevoede apparatuur! Het ontwerp van de converter heeft grotere afstanden en ook de transformatoren (indien juist geconstrueerd) levert een uitstekend galvanisch geïsoleerde uitgangs-hoogspanning. Zet instelpot P1 in de middenstand voor een schakelfrequentie van 150 kHz.

Zorg er bij het gebruik van deze DC/DC-converter voor dat er zich geen elektrisch geleidende voorwerpen in de buurt van hoogspanning voerende componenten bevinden. Hoe groter de afstand hoe beter!

De uitgangsspanning kan worden gewijzigd door het aantal secundaire windingen aan te passen, maar ook door het aantal primaire windingen te veranderen, mits de juiste schakelfrequentie in acht wordt genomen. Veranderen van C1 (en R2) voor een andere oscillatorfrequentie zal de dode tijd veranderen! Kijk goed naar de formules in de datasheet van de UCC28089 alvorens waarden te wijzigen.



Figuur 4. Print-layout.





GERELATEERDE PRODUCTEN

➤ **M. van der Veen, Vanderveen Trans Buizenversterkers , Elektor (e-boek)**
www.elektor.nl/vanderveen-trans-tube-amplifiers-e-book

➤ **M. van der Veen, Ontwerpen van buizenversterkers, Elektor (e-boek)**
www.elektor.nl/ontwerpen-van-buizenversterkers-e-book

Een bijdrage van

Idee, ontwerp, tekst: **Ton Giesberts**

Redactie: **Luc Lemmens**

Vertaling: **Jelle Aarnoudse**

Illustraties: **Ton Giesberts,**

Patrick Wielders

Layout: **Giel Dols**

Het ontwerpen van push-pull DC/DC-convertisers en – misschien nog in sterkere mate – de berekening en constructie van de benodigde 'op maat gemaakte' transformator is behoorlijk gecompliceerd. Zoals eerder vermeld, is meer achtergrondinformatie over dit onder-

werp te vinden op [1], waar u ook vragen, opmerkingen, suggesties en ervaringen met deze converters of transformatorontwerp kunt delen met andere lezers.



200583-03

Vragen of opmerkingen?

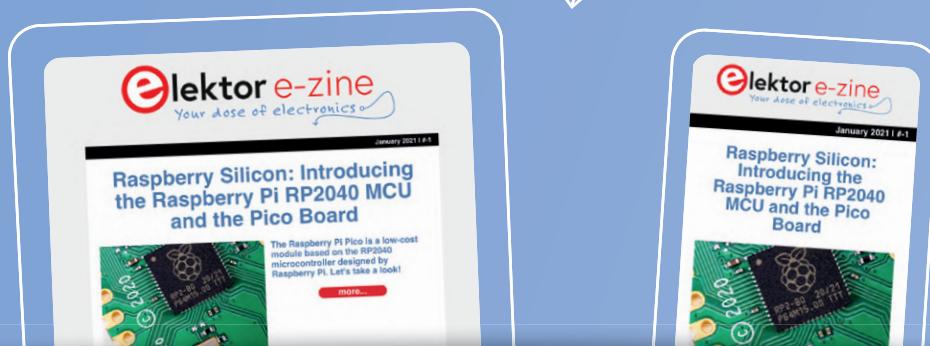
Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur via ton.giesberts@elektor.com of naar de redactie van Elektor via redactie@elektor.com.

WEBLINKS

- [1] **Labs-pagina bij dit artikel:** www.elektermagazine.nl/labs/12v-200v-dc-dc-converter-for-valve-amplifiers
- [2] **Datasheet TDK-spoelvorm:** https://product.tdk.com/info/en/documents/data_sheet/80/db/fer/etd_29_16_10.pdf
- [3] **Downloads van dit project:** www.elektermagazine.nl/200583-03

elektor e-zine

Your dose of electronics



Elke week dat u zich niet inschrijft voor ons Elektor E-zine mist u leuke elektronica-gerelateerde artikelen en projecten!

Dus, waarom nog langer wachten? Abonneer u vandaag nog op www.elektor.nl/ezine en ontvang bovendien een gratis Raspberry Pi projectboek!



elektor
design > share > sell

Warmtezoeker

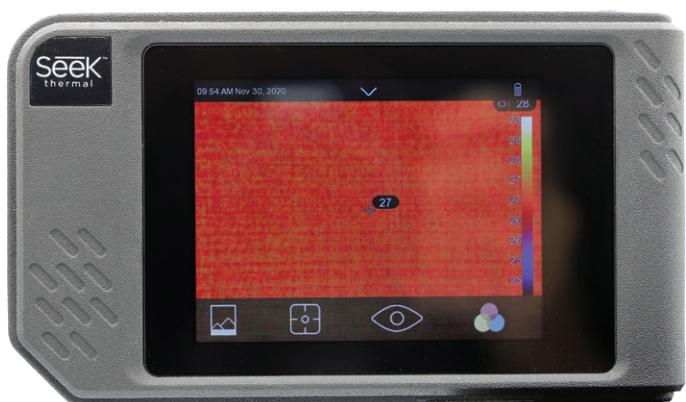
de Seek Shot Pro warmtebeeldcamera

Alfred Rosenkränzer (Duitsland)

We hebben onlangs een aardige thermische camera aan de tand gevoeld om de bruikbaarheid ervan in het lab te testen. We ontdekten dat deze vestzak-camera in de dagelijkse praktijk snel problemen kan opsporen. Maar het is een *kleine* camera – is het display eigenlijk wel groot genoeg? Laten we dat van dichterbij bekijken.



Figuur 1. Vooraanzicht met IR-lens, normale lens en LED-lichtbron.



Figuur 2. Achteraanzicht met ingeschakeld display.

Het in Santa Barbara, Californië gevestigde Seek Thermal heeft een reeks professionele IR-beeldvormingsproducten in haar catalogus. De Seek Shot Pro warmtebeeldcamera werd geleverd in een stevige kartonnen doos van 173 x 115 x 44 mm. Meegeleverd waren een USB-kabel, een draagriem en een meertalig informatieblad.

De camera meet 140 x 80 x 25 mm en lijkt meer op een smartphone dan op een 35mm-camera (**figuur 1**). De behuizing heeft een rubberen oppervlak dat goed in de hand ligt, zodat hij niet zo snel uit je hand glipt, en dat tegelijkertijd mechanische bescherming biedt. Op de behuizing zitten slechts twee mechanische drukknoppen: een aan/uit-knop en een ontspanknop voor het maken van foto's of video's. Voor alle andere instellingen wordt het aanraakscherm gebruikt.

Aan de buitenkant

De USB-poort van de camera is aan de onderkant verborgen onder een klepje. Deze wordt gebruikt om de batterij van de camera op te laden en om beelden over te brengen naar een PC. Het interne geheugen kan niet worden uitgebreid omdat er geen slot voor een SD-kaart is. Het grootste verschil met een smartphone is de 0,25"-adapter voor het bevestigen van de camera op een standaard statief.

Figuur 1 toont de voorzijde. In de rechterbovenhoek bevindt zich de grote IR-beeldvormingslens. Daaronder zit een kleinere lens voor zichtbaar licht. Links van de IR-lens bevindt zich een LED die wordt gebruikt voor extra flitsverlichting bij weinig licht. De LED kan ook onafhankelijk worden in- en uitgeschakeld om als zaklamp te fungeren als u de weg in het donker mocht kwijtraken.

De Seek Shot-warmtebeeldcamera's zijn in twee versies verkrijgbaar. De budgetversie produceert een IR-beeld met een resolutie van 206 x 156 pixels, terwijl de Pro-versie 320 x 240 pixels biedt. Dat lijkt wellicht bescheiden in vergelijking met de pixelresolutie die u normaaliter van een smartphonecamera zou verwachten, maar IR-warmtebeeld-arrays hebben nu eenmaal een veel geringere resolutie. Maar voor de meeste toepassingen levert een IR-warmtebeeld met een hogere resolutie geen echte voordelen op.

De Pro-versie biedt een gezichtsveld van 57° wat overeenkomt met een brandpuntsafstand van ongeveer 40 mm bij een 35mm-camera. Dit is iets meer dan het gezichtsveld van het menselijk oog. De budget-versie van de camera heeft een gezichtsveld van 36° (≈66 mm bij een 35mm-camera) waardoor deze iets meer op een tele-objectief lijkt. Het enige direct zichtbare verschil tussen de twee camera's van buitenaf is de kleur van de rand rond de IR-lens. In de budgetversie

is die grijs terwijl hij in de Pro-versie rood is. Zoals u in figuur 1 ziet, test ik hier de Pro-versie.

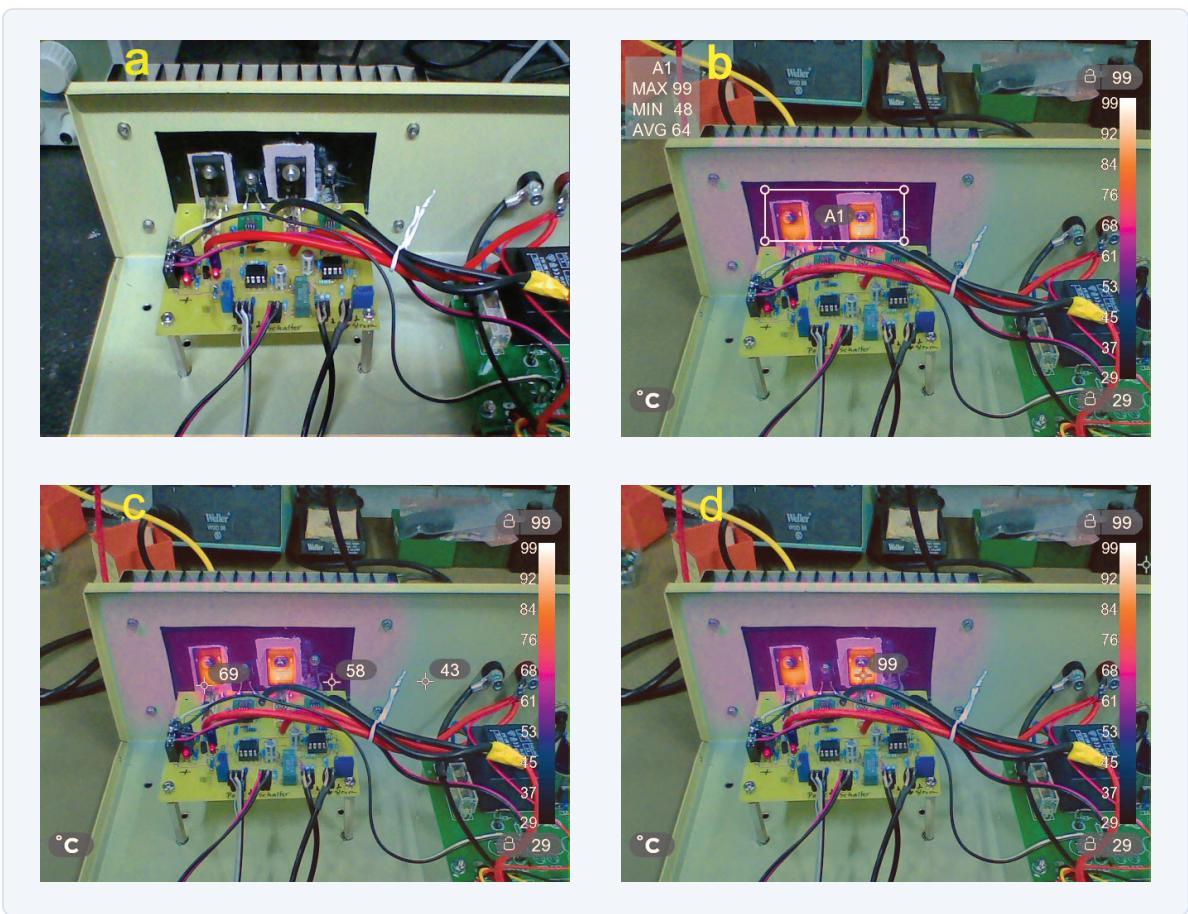
Erste indruk

Allereerst moet u de USB-kabel aansluiten om de accu van de camera op te laden. Dat geeft u wat tijd om vertrouwd te raken met de bedieningsinstructies en om enkele instructievideo's te bekijken op de website van het bedrijf [1] – dat is tenminste wat ik deed. Het is de

lastingsgebeurtenis vast te leggen, kunt u een reeks foto's maken en die later bestuderen.

Het display

Na inschakelen duurt het een paar seconden voordat de tijd en datum links in de bovenste balk van het display verschijnen (**figuur 2**). De ladingstoestand van de accu is rechts te zien. In het midden van deze balk staat een naar beneden wijzende pijl. Deze geeft toegang tot het



Figuur 3. Het prototype van een elektronische belasting normaal visueel (3a), als gecombineerd beeld (3b), gecombineerd beeld met drie meetpunten (3c) en gecombineerd beeld met weergave van de maximale temperatuur (3d).

moeite waard om de camera en zijn functies te leren kennen voordat u hem serieus gaat gebruiken.

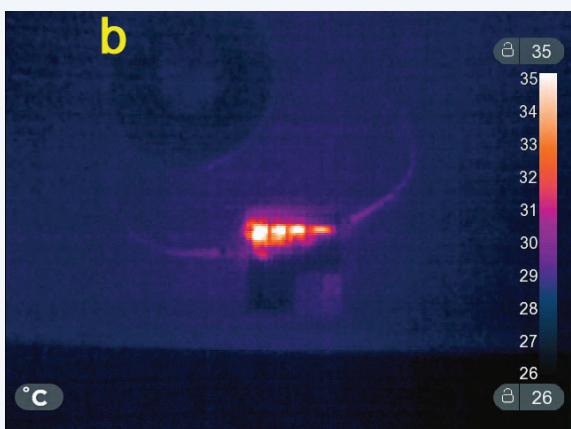
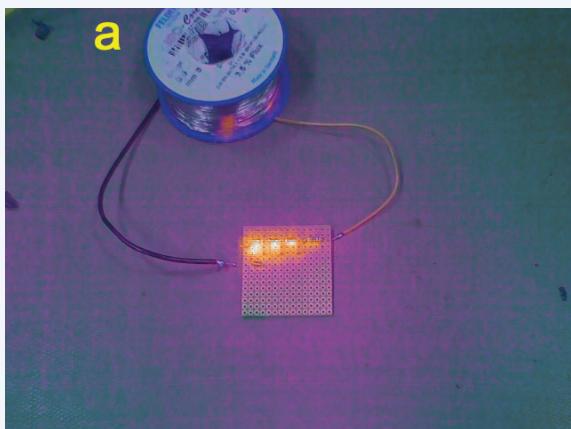
De camera is grotendeels van het point-and-shoot type. De meeste instellingen vinden automatisch plaats – u kunt diafragma of belichtingstijd niet wijzigen terwijl beide lenzen een vaste brandpuntsafstand hebben – daar valt dus ook niets in te stellen. Volgens de specificaties bedraagt de minimale beeldafstand ongeveer 10 tot 15 cm. Dat geldt voor beide lenzen, dus de Seek Fusion-beeldcombinatie werkt ook tot deze afstand.

Voor mij is een van de handigste functies van het systeem de mogelijkheid om een afbeelding achteraf te analyseren en deze vervolgens als kopie in de galerij op te slaan. Hierdoor kunt u een groot aantal verschillende versies van een enkele afbeelding maken en de resultaten naar behoefte opslaan. Om een enkele thermische of overbe-

menu waar u zaken als de helderheid van het display kunt aanpassen en waar u de eenheid (*celsius*, *fahrenheit* of *kelvin*) van de weergegeven temperatuur kunt instellen. U kunt hier ook de emissiviteit van de te fotograferen oppervlakken instellen. Dat is belangrijk om de foto's de juiste temperaturen te laten tonen.

Het zaklamp-pictogram rechts kan worden gebruikt om de witte LED aan de voorzijde in te schakelen; een flitspictogram activeert de LED-flitsfunctie. Met de ingebouwde helpfunctie kan de WiFi-link worden ingesteld of kan een beschrijving van de camera worden opgeroepen. Een standaard tandwiel-pictogram brengt u naar het menu met geavanceerde instellingen (tijd, datum, menutaal en andere configurerbare functies). Na terugkeer uit het menu ziet het display er ongeveer uit als in figuur 2.

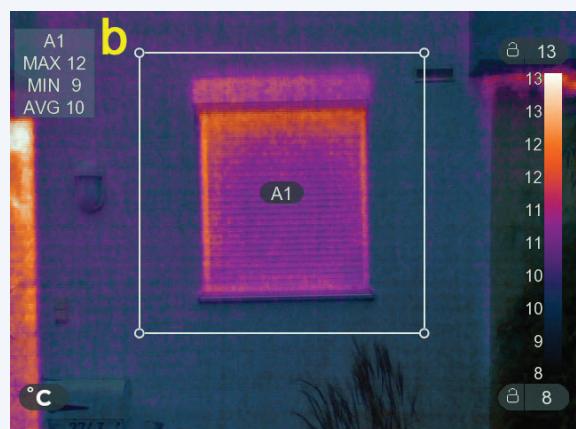
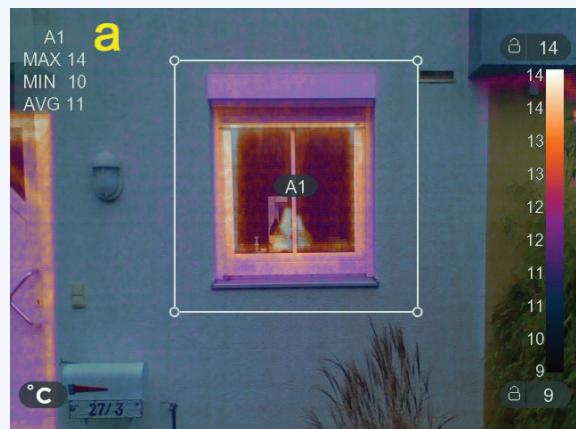
Langs de onderkant ziet u vier pictogrammen. Aan de linkerkant brengt



Figuur 4. Figuur a toont de schakeling op gaatjesprint als een gecombineerd beeld, terwijl b alleen het thermische beeld toont.

het landschapspictogram u naar de fotogalerij met eerder opgeslagen foto's. Deze kunnen worden opgeroepen en bewerkt. Daarover later meer. Het tweede pictogram van links opent een ander menu waarin de verschillende temperatuurmeetopties kunnen worden geselecteerd. Een klik op het linker dradenkruis-pictogram toont de temperatuur van het oppervlak in het midden van het actieve scherm. Door de camera te bewegen kunnen we de temperatuur aflezen, maar de temperatuurweergave is ietwat vertraagd zodat een vaste hand van voordeel is. Het driepuntspictogram geeft de spot-temperatuur van maximaal drie posities in het beeld weer. Het pictogram met een vierkant met stippen op de hoeken creëert maximaal drie van belang zijnde interessegebieden met aanpasbare en verplaatsbare positie in de actieve scène. De camera kan dan de maximum-, minimum- en gemiddelde temperaturen voor elk van deze velden weergeven. Het pictogram met plus- en mintekens uiterst rechts activeert de meting van het warmste en koudste punt in het beeld. Bekijk de trainingsvideo's om te zien hoe deze functies werken.

Het oogpictogram opent een menu om de weergave te wijzigen. Als het met ononderbroken lijnen weergegeven oogpictogram is geselecteerd, wordt het normale optische beeld weergegeven. Het met stippellijnen weergegeven oogpictogram rechts wordt geselecteerd om het warmtebeeld weer te geven. Het pictogram met beide 'ogen' over elkaar selecteert de Fusion View. Aanklikken resulteert in twee bedie-



Figuur 5. Het is koud buiten, met het rolluik omhoog (a) en met het rolluik neergelaten (b).

ningselementen. De linker past de beeldmix aan tussen optisch en thermisch beeld. Met de rechter kunnen beide beelden verticaal worden verschoven om de verticale parallaxfout te compenseren; deze fout wordt storender naarmate beide lenzen zich dichter bij het object bevinden. Als u het kleurpictogram aanklikt, wordt het kleurenpalet weergegeven waarmee u verschillende kleurenschema's op de actieve beeld kunt toepassen. Afhankelijk van het temperatuurbereik in het warmtebeeld kan een aangepast palet beter geschikt zijn om temperatuurverschillen te onderscheiden.

Testbeelden

De temperatuurschaal is meestal 'autoranging', maar kan naar wens worden aangepast zodat beelden gemakkelijker kunnen worden vergeleken. **Figuur 3a** toont een normaal visueel beeld van een elektronische belasting met de kap verwijderd. In **figuur 3b** zien we een combinatiebeeld dat bestaat uit het visuele beeld met het thermische beeld daar overheen – dat is de Seek Fusion-weergave. De maximumtemperatuur in het geselecteerde rechthoekige gebied is 99 °C. Langs de rechterkant van het beeld zien we de temperatuur-kleurenschaal. **Figuur 3c** toont de temperatuur van drie geselecteerde meetposities op het koellichaam en de behuizingsoppervlakken. In **figuur 3d** kunnen we ontdekken waar de maximale temperatuur is geregistreerd. In een ander voorbeeld (**figuur 4a**) heb ik een kleine testschakeling

op gaatjesprint opgebouwd: vijf 100Ω-weerstanden in serie. De eerste weerstand is een conventioneel through-hole exemplaar, terwijl de andere vier SMD-exemplaren zijn (resp. 1206, 0805, 0603 en 0402). Door de schakeling loopt een stroom van 10 mA. De temperatuurverschillen tussen de afzonderlijke weerstanden zijn vrij klein maar zijn in het warmtebeeld duidelijk te zien. Het geringe temperatuurbereik in dit beeld is te zien in **figuur 4b**.

Het behoeft geen betoog dat een warmtebeeldcamera niet alleen een goed hulpmiddel is voor elektronica-ingenieurs; het is ook een handige manier om snel gebieden met warmteverlies en isolatiefouten in een woonomgeving te identificeren. Om dat uit te testen ging ik naar buiten en nam wat foto's. Hier in Europa zijn vensters aan de buitenzijde vaak voorzien van rolluiken (metaal of kunststof). Ik heb een paar foto's gemaakt om te demonstreren in welke mate deze het warmteverlies reduceren. **Figuur 5a** toont het venster met opgehaalde rolluik; in **figuur 5b** is het volledig gesloten. U ziet een maximaal temperatuurverschil van 2 °C.

Het spreekt voor zich dat wanneer we een camera in de hand hebben, we een selfie (in dit geval een thermo-selfie) moeten maken. De resolutie en nauwkeurigheid van dit specifieke cameramodel zijn niet genoeg om koorts te meten via de huidtemperatuur. (Seek Thermal Inc produceert echter gespecialiseerde apparatuur voor dergelijke toepassingen.) Volgens **figuur 6** zou ik onmiddellijk naar bed moeten met een kool drankje. De temperatuurnauwkeurigheid wordt opgegeven als ±2 °C en is sterk afhankelijk van de oppervlakte-emissiviteit die kan worden ingesteld in het menu.

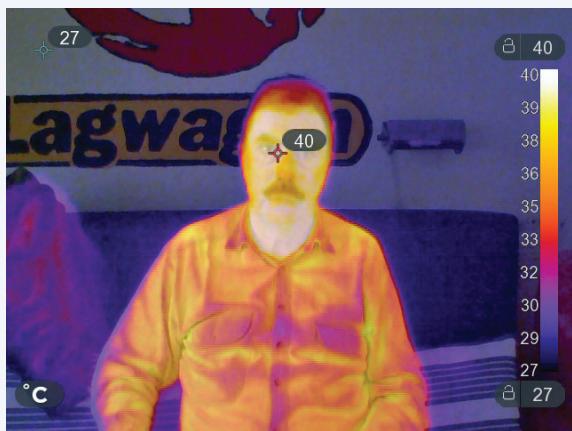
Ik heb al deze foto's maar één keer gemaakt en ze vervolgens in de galerij bewerkt en als kopie opgeslagen. Open hiervoor de galerij door op het landschapspictogram te tikken, selecteer de juiste afbeelding met de pijltjestoetsen en activeer de optie bewerken. Als u klaar bent, slaat u de afbeelding op als kopie. Op deze manier blijft het origineel bewaard voor andere toepassingen. Dit alles (inclusief live streaming) kan ook op een smartphone of tablet worden gedaan via WiFi en de bijbehorende app, hoewel ik dat niet heb gecontroleerd. Een app voor de PC, zoals die voor andere warmtebeeldcamera's beschikbaar is, zou wenselijk zijn.

Bewerkingen uitvoeren met 'dikke' vingers op het kleine display kan frustrerend zijn en vereist veel concentratie en geduld. Ik gebruik deze infraroodcamera vooral om snel elektronische schakelingen te checken en vond hem opmerkelijk handig.

Conclusie

Een warmtebeeldcamera is een uitstekend hulpmiddel in het lab om problemen in prototypes op te sporen en overbelasting te detecteren. Het is een veel praktischer benadering vergeleken met bijvoorbeeld het gebruik van een thermometer-sonde of een natte vinger. In sommige gevallen krijgt u een snelle indicatie dat een onderdeel in uw nieuwste project zijn dissipatielimit nadert zodat u misschien nog genoeg hebt om de stekker uit het stopcontact te trekken en er geen rookeffecten optreden. Het is ook buitengewoon handig om warmtebeelden te kunnen maken en opslaan voor latere analyse en evaluatie, indien nodig.

De Seek Fusion-weergave die normale en thermische beelden combineert, komt goed van pas om snel een defect onderdeel als warmtebron te identificeren of om een gebied te tonen van waaruit warmte ontsnapt uit een gebouw of afgesloten ruimte. Voor algemene laboratorium-toepassingen hoeft de resolutie van het warmtebeeld niet bijzonder hoog te zijn. Met dit in gedachten zou ik ook denken (hoewel ik niet de mogelijkheid had om die te testen) dat de geringere resolutie van de



Figuur 6. Een thermo-selfie. Ik zit er warmpjes bij!

budgetversie van de Seek Shot meer dan voldoende zou moeten zijn voor het meeste laboratoriumwerk. In deze besprekking heb ik gekozen voor de betere en duurdere Pro-versie. Voor iets meer dan € 800 doet de Seek Shot Pro het goed in de vergelijking met een vergelijkbare professionele camera van Flir, die ongeveer 10 keer duurder is (en inmiddels meer dan 10 jaar oud). Al met al vond ik hem best indrukwekkend en vooral handig voor algemeen laboratoriumgebruik! 

200654-03

Vragen of opmerkingen?

Hebt u vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur of naar de redactie van Elektor via redactie@elektor.com.

Een bijdrage van

Recensie en tekst:

Alfred Rosenkränzer

Redactie: **Dr. Thomas Scherer**

Vertaling: **Eric Bogers**

Layout: **Giel Dols**



GERELATEERDE PRODUCTEN

► Seek Shot Pro

www.elektror.nl/seek-shot-pro-thermal-imaging-camera-320x240

WEBLINK

[1] Seek Shot webpagina:

www.thermal.com/seekshot-series.html

Objectgeoriënteerd programmeren

een korte inleiding met C++

Roland Stiglmayr (Duitsland)

Wilt u zelf krachtige objectgeoriënteerde programma's schrijven? Dan moet u zich eerst vertrouwd maken met de voordelen van OOP ten opzichte van procedureel programmeren. Laten we eerst wat theorie behandelen, om dan met enkele praktische voorbeelden verder te gaan.

Objectgeoriënteerd programmeren (OOP) bestaat al meer dan 40 jaar. Enkele van de vele voordelen ervan zijn de superieure kwaliteitsborgingsprocedures, het eenvoudige software-onderhoud en de uitstekende structurering die teamwerk bij programma-ontwikkeling vereenvoudigt. De afgelopen jaren heeft OOP ook zijn weg gevonden naar embedded programmeren. De Arduino IDE ondersteunt ook het gebruik ervan; het lijkt daarom een goed moment om het onderwerp wat nader te bekijken. Dit artikel kan de belangrijkste kenmerken van OOP slechts kort aanstippen, maar zal hopelijk een goed kennis-

fundament creëren waarop u uw eigen objectgeoriënteerde programma's kunt opbouwen.

Als het gaat om software-ontwikkeling, kan men in principe twee benaderingen gebruiken. De traditionele methode maakt gebruik van *procedureel programmeren*, in tegenstelling tot *objectgeoriënteerd programmeren*. Procedureel programmeren is wat we altijd hebben gedaan (bijvoorbeeld bij programmeren in assembler of in standaard C). Hierbij wordt het programma opgesplitst in kleine modules (dat wil zeggen procedures of functies) en sequentieel uitgevoerd. De modules communiceren met behulp van gemeen-

Listing 1. Klasse-declaratie van Virt_M (les_1).

```
class Virt_M
{
    private:
        float voltage;           //the following members are only
        float current;          //accessible within the class
        float p_result;         //virtual voltage/ V
        float r_result;         //virtual current/ A
                                //calculated power/ W
                                //calculated resistance/ Ohm.

    public:
        String s= ("public: Demo attribute s, type String"); //for demo only
        // declaration of the constructor, its method is called when an object
        // of this class is created

        Virt_M (float v, float c);           //declaration of the constructor
        // declaration of the methods/ functions of the class Virt_M

        float get_P ();                    //method reads power
        float get_R ();                    //method reads resistance
        float get_Voltage ();             //method reads voltage
        float get_Current ();             //method reads current
        void prep_Meas (float v, float c); //method simulates the measurement

    private:
        void set_Voltage (float v);       //only accessible within the class
        void set_Current (float c);      //interface method for writing voltage
                                         //interface method for writing current
};

//end of the class declaration
```

schappelijke, vaak globaal gedeclareerde data en gebruiken deze om resultaten van hun acties uit te wisselen. OOP daarentegen kapselt de gegevens en de functies die deze gebruiken in objecten in. Toegang tot gegevens en functies vindt plaats via hiervoor bestemde functies, ook wel interfaces genoemd. Inkapseling beschermt de data tegen ongeoorloofde toegang.

Wat zijn nu de kenmerken van OOP? Bij OOP worden functionele eenheden gecreëerd met logisch verwante leden zoals data en functies. De functies worden hier methoden genoemd. In het geval van een datalogger bijvoorbeeld zouden de data-acquisitie-eenheid met zijn vele meetkanalen, de verwerkingseenheid en de gebruikersinterface elk als een functionele eenheid kunnen worden beschouwd. Een functionele eenheid kan de eigenschappen van andere functionele eenheden erven, zodat uit deze basiseenheden een modulaire structuur kan worden opgebouwd. Bescherming van de data en methoden wordt bereikt door het principe van inkapseling. We kunnen zo'n functionele eenheid beschouwen als een apart gegevenstype: een *klasse*. De klasse vertegenwoordigt een soort blauwdruk waaruit de variabelen worden gecreëerd. Die variabelen worden objecten of instanties van de betreffende klasse genoemd. Om misverstanden te voorkomen, moet u zich terdege bewust zijn van het verschil tussen een klasse en een object! Vanuit een klasse kan een willekeurig aantal objecten worden gemaakt. Van elke persoon die in een bedrijf werkt, zou zouden we kunnen zeggen dat hij een object voorstelt, waarbij elk van deze objecten is gemaakt vanuit een en dezelfde klasse.

Genoeg theorie, tijd voor de praktijk! Zoals altijd vormen kant-en-klare voorbeelden waar u aan kunt sleutelen en die u kunt aanpassen om uw eigen ideeën uit te proberen, de beste manier om te beginnen. En wanneer dit binnen een vertrouwde geïntegreerde ontwikkelomgeving (IDE) gebeurt, spaart dat tijd en is de leercurve minder steil. Om deze reden hebben we de populaire gratis Arduino IDE gebruikt om de hier gebruikte voorbeelden uit te voeren. De geïntegreerde compiler is een volwaardige C++ compiler en daardoor bij uitstek geschikt voor objectgeoriënteerd programmeren. Om de sketches in een Arduino-board te laden, wordt dit board aangesloten op de USB-poort van de computer. Onze voorbeeldprogramma's (in de Arduino-wereld worden die 'sketches' genoemd) zijn georganiseerd als zeven 'lessen' die kunnen worden gedownload van de projectpagina [1] bij dit artikel. Nadat ze zijn gedownload, kunnen ze worden uitgepakt en naar de map Arduino IDE Sketch worden gekopieerd. Als dat eenmaal is gebeurd, zijn de voorbereidingen voltooid. De voorbeelden zijn allemaal gebaseerd op het concept van een virtueel meetapparaat; er is geen hardware bij betrokken. We illustreren hiermee alleen de processen die betrokken zijn bij het overdragen van informatie. De voorbeeld-sketches zijn niet geschikt voor een 'echt' hardwarematig meetapparaat, maar ze demonstreren op aanschouwelijke wijze de basisfuncties van OOP aan de hand van een taak die gemakkelijk te begrijpen is.

De eerste klasse

Om te beginnen zullen we kennismaken met het creëren van een klasse, de betekenis van de toegangsspecificatie, de basisprincipes van inkapseling en het creëren van instanties. Hiertoe moeten we de sketch van *les_1* in de IDE laden, compileren en vervolgens naar het board overdragen. Voordat we de seriële monitor van de Arduino IDE gebruiken om de uitvoer te bekijken die door het programma wordt gegenereerd, werpen we eerst een blik op de broncode van het programma in **listing 1**.

Op de eerste regel zien we de descriptor-klasse met de naam **Virt_M**. Deze vertelt de compiler dat een klasse met de naam **Virt_M** moet worden gegenereerd. In de volgende regels kunnen we declaraties zien van de interne data (of attributen) die in de klasse worden gebruikt, en de methoden van de klasse. Voor toegangscontrole worden de leden van de klasse, de attributen en de methoden van de klasse voorzien van **private** en **public** toegangsspecificaties. Leden met **private** toegang zijn alleen toegankelijk vanuit de klasse. Alle leden met **public** toegang zijn toegankelijk van buiten de klasse. We zullen later zien wat dit inhoudt.

De eerste methode, die dezelfde naam heeft als de klasse en die geen waarde retourneert, is de constructor. De constructor wordt altijd aangeroepen wanneer een object van het type van deze klasse wordt gecreëerd. Aan de constructor worden meestal waarden doorgegeven om de objectattributen te initialiseren.

De andere methoden in het kort: de methoden die beginnen met **get_** staan toe dat de **private** attributen van de klasse worden gelezen. De methoden die beginnen met **prep_** worden gebruikt om deze attributen in te stellen. Deze methoden zijn **public** en brengen daardoor interfaces tot stand met de private attributen. De **set_** methoden die zijn opgegeven als **private** kunnen alleen binnen de klasse worden gebruikt. **String s** wordt gebruikt om de specificatie voor **public** toegang te demonstreren. De sluit-accolade gevolgd door een puntkomma geeft het einde van de declaratie van de klasse aan.

Hierna volgen de definities van de methoden die hierboven zijn gedeclareerd. Dit is identiek aan de definities van functies in standaard C, afgezien van de gebruikte syntaxis. Die luidt hier:

```
class name::method name(parameterst)
```

De verwijzing naar de klasse wordt tot stand gebracht met de scope reference operator, een dubbele dubbele punt.

Onze eerste klasse is nu klaar; van hieruit kunnen we objecten maken. Net als bij de functiedeclaratie bestaat de naam van de klasse uit het type gevolgd door de naam van het object en (indien overeengekomen) een lijst met waarden die moeten worden overgedragen (**listing 2**). Bij het maken van een object, ook wel instantiëren genoemd, wordt de constructormethode van de klasse aangeroepen en worden de attributen van het betreffende object gemaakt. Het is erg belangrijk om te beseffen dat elk object zijn eigen, individuele datagedeelte heeft, namelijk de attributen.

We hebben nog niet gezegd wat een object van deze klasse doet. In dit voorbeeld simuleert het de acquisitie van spanning- en stroom-meetwaarden; deze verzonneen waarden worden samen met de resulterende waarden voor weerstand en vermogen beschikbaar gesteld voor de corresponderende methoden. Een **public** methode wordt als volgt aangeroepen:

```
Object_name.method_name () ;
```

Attributen die met **private** worden beschermd, zijn alleen toegankelijk met de methoden die daarvoor bestemd zijn. Als u probeert ze rechtstreeks te benaderen, retourneert de compiler een foutmelding. Dit kan worden gedemonstreerd door de betreffende commentaar-slashes (**//**) in de setup-routine weg te halen (**listing 3**). Als ander voorbeeld van toegangscontrole hebben we **String s** opzettelijk als **public** gedeclareerd. Attributen die op deze manier zijn gedeclareerd, zijn rechtstreeks toegankelijk met **Objectname**.

Listing 2. Creëren van de instanties (les_1).

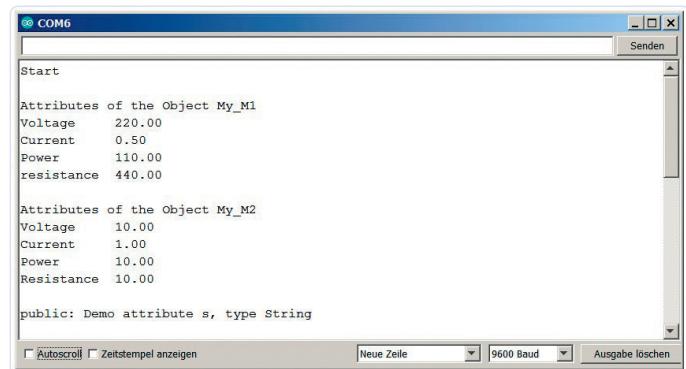
```
// creating instances/ objects of type
// Virt_M. preset the attributes start values
//-----
Virt_M My_M1 (220.0,0.5);      //creates the object/ instance My_M1 by
                                //calling the constructor of Virt_M

Virt_M My_M2 (10.0,1.0);       //creates the object/ instance My_M2 by
                                //calling the constructor of Virt_M
```

Listing 3. Test van de toegangscontrole (les_1).

```
// for testing the access control delete the comment instruction "//"
// -----
// My_M1.voltage= 0;           //error > attribute is private
// My_M1.set_Voltage(0);       //error > method is private
String str= (My_M1.s);         //permitted > attribute s is public
Serial.println (My_M1.s);
```

Attributename. Maar let op! Het object beschermt **public** leden niet tegen onjuist gebruik door het gebruikersprogramma. Het hoofdprogramma verandert de gelezen waarden met behulp van de methode **prep_Meas**, die de interne **set** methoden aanroeft. De programmavoorbeelden bevatten nuttig commentaar en veel informatie, die moeten bijdragen aan een beter begrip. Na het starten van de applicatie voert de setup-routine de attributen van de objecten uit zodat we die kunnen bekijken met behulp van de seriële monitor (**figuur 1**).

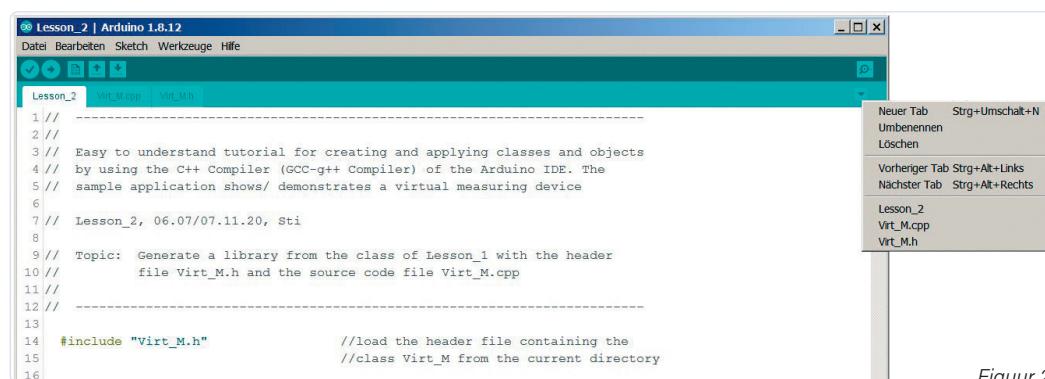


Figuur 1. Uitvoer van de setup-routine van les_1.

Een klasse als programmabibliotheek

Het concept van bibliotheken is niet nieuw in de wereld van software-ontwikkeling. U hebt ze al gebruikt, als u enige ervaring hebt met coderen in standaard C. Ze worden nog intensiever gebruikt in OOP. De reden hiervoor is dat de code-ontwikkelaar de details van de broncode van een klasse moet kennen om er gebruik van te kunnen maken. Het gebruik van bibliotheken ondersteunt de herbruikbaarheid van programma-onderdelen en bevordert daarmee het concept van modulariteit in het programma-ontwerp. Een voorbeeld hiervan wordt gegeven in *les_2*, waar de klasse uit de eerste sketch wordt gebruikt als programmabibliotheek. Een bibliotheekitem bestaat uit twee bestanden. Het headerbestand (.h) bevat het declaratiegedeelte van de klasse en het bronstekst-bestand (.cpp) de methodedefinitie van de klasse. Hoewel het niet strikt noodzakelijk is, is het handig om voor beide bestanden dezelfde naam te gebruiken. Om onze **Virt_M** klasse als een bibliotheek weer te geven, hoeft alleen het declaratiedeel naar het headerbestand te worden overgebracht en het definitiedeel naar het bronstekstbestand. Tijdens de ontwikkeling wordt aanbevolen om de bibliotheekbestanden op te slaan in dezelfde directory als het gebruikersprogramma. Het gebruik van de tab-functie in de Arduino IDE helpt ons hierbij, maar daarover later meer.

Het is gebruikelijk om aan het begin van het header-bestand te controleren of het al door de compiler is opgenomen. Die situatie kan zich voordoen als een ander header-bestand al onze header heeft ingesloten. Als dat het geval is, worden leden met dezelfde naam meerdere keren gedeclareerd. De syntax voor de preprocessor om dit te testen is:



Figuur 2. De Arduino tab-functie (les_2).

```
#ifndef Virt_M_h // the symbol Virt_M_h is used to
    test if the header file was already integrated
#define Virt_M_h // if the symbol does not yet exist,
    then define the symbol
class .... // now declare the class
{ .... }; // end of declaration
#endif // finish conditional compiling
```

De `#include "Virt_M.h"` instructie in het bronbestand wordt gebruikt om het headerbestand op te nemen. In ons voorbeeld is `Arduino.h` ook opgenomen. Dit is nodig omdat de compiler op dit moment de standaardbibliotheeken nog niet kent. Daarmee is onze bibliotheek klaar. Om nu de `Virt_M` klasse in een gebruikersprogramma te gebruiken, kunnen we `#include "Virt_M.h"` gebruiken om het header-bestand te laden.

Arduino-bibliotheeken bieden vaak al een instantie van hun klasse-bibliotheek. In de Wire-bibliotheek is `Wire` bijvoorbeeld een object van de klasse `TwoWire`. Het object wordt opgenomen met `#include <Wire.h>`.

Nadat de ontwikkeling is voltooid, kunnen we onze bibliotheekbestanden in hun eigen directory opslaan. Hiervoor maakt Arduino de bibliotheekdirectory aan in de sketchbook-directory. We kunnen daar een andere map maken waarin we onze bestanden zullen opslaan. Dit werkt ook via de IDE met *Sketch -> Include library -> Add .ZIP library*. Zoals al opgemerkt, is de tab-functie van de Arduino IDE erg handig bij het schrijven van meerdere bestanden die bij een project horen. Deze functie wordt geactiveerd via de kleine driehoek in de rechterbovenhoek van het venster (**figuur 2**).

Overerving – een kernconcept van OOP

Als u zich eenmaal in OOP begint te verdiepen, duurt het niet lang voordat u het idee van overerving tegenkomt. Laten we eens kijken naar de voordelen van dit concept. Overerving betekent het beschikbaar stellen van de attributen en methoden van een basisklasse aan een overervende klasse. De overervende klasse kan een onderliggende klasse, afgeleide klasse of subklasse worden genoemd. De basisklasse wordt ook wel de bovenliggende klasse of superklasse genoemd. De afgeleide klasse voegt extra eigenschappen toe aan die welke worden geërfd en die ook verder kunnen worden overerfd. Dit betekent dat veel verschillende afgeleide klassen kunnen voorkomen uit een overervende klasse, zodat de resulterende structuur niet noodzakelijk lineair is, maar eventueel vertakt. Een relatie die bestaat uit vele takken!

Les_3 laat zien hoe we overerving kunnen gebruiken voor onze doeleinden. Om het voorbeeld wat zinvoller te maken, gebruiken we multiplex-adressen om de verschillende meetkanalen te selecteren en deze te koppelen aan de aangemaakte objecten. `Virt_M` is de basisklasse. Hier kunt u zien dat er drie constructors zijn met een verschillend aantal parameters. C++ laat in het algemeen dit zogenaamde ‘overladen’ van functiedefinities toe (dat wil zeggen functies die dezelfde naam gebruiken met verschillende nummers en/of verschillende parametertypes). In OOP staat dit bekend als *polymorfisme*. Merk op dat de sleutelwoord-toegangsspecificatie is ingesteld op `protected`. Dit stelt de toegankelijkheid van methoden, klassen en andere leden in voor de overervende klasse. In ons voorbeeld geeft het aan dat alle overervende klassen toegang hebben tot de `set_Mux` methode.

Om te tonen dat dit is toegestaan, is de methode gedefinieerd in het declaratiegedeelte (dit heeft alleen werkelijk zin voor zeer korte

functies). De eerste afgeleide klasse is `Volt_M`, die de klasse `Virt_M` erft. De syntax hiervoor is:

```
class Volt_M : public Virt_M // class Volt_M
    // inherits the class Virt_M
```

Het keyword `public` zorgt ervoor dat de toegangscontrole tot de leden van `Virt_M` behouden blijft. `Volt_M` levert het `sV` attribuut, de constructor en de `get_Unit` methode aan zijn overgeërfd leden. De declaratie van de constructor introduceert niets nieuws, maar de definitie wel. Hier wordt de constructor 2 van `Virt_M` aangeroepen. Tijdens instantiatie worden de parameters die aan `Volt_M` worden doorgegeven, doorgegeven aan de `Virt_M` klasse, de startwaarde `ini_m` rechtstreeks via de constructor `Virt_M`, het multiplex-adres via de `set_Mux` methode van `Volt_M`. De definitie van de constructors is:

```
Volt_M::Volt_M (byte v_mux, float ini_m):Virt_M
    (ini_m) //set ini_m via constructor of of class
    // Virt_M
    { set_Mux (v_mux); }
//v_mux via set method
```

De klasse `Amp_M` is ook een erfenis van `Virt_M`. Deze heeft een vergelijkbare structuur als `Volt_M` met het verschil dat de constructor de constructor 3 aanroeft vanuit `Virt_M`. De volledige overdracht van waarden naar `Virt_M` wordt uitgevoerd zonder enige aanvullende methode. Het is vermeldenswaard dat een overerving een overgeërfd methode kan opheffen.

De klassen zijn nu volledig geformuleerd, zodat we er exemplaren van kunnen maken. De objecten `M1`, `M2` en `M3` van klassetype `Virt_M` tonen de aanroep van de overladen constructor. `My_Volt_M` en `My_Amp_M` zijn objecten van de overgeërfd klasse `Volt_M` en `Amp_M`. Na het starten van het programma kan het gedrag van onze objecten worden gevolgd en begrepen met behulp van de seriële monitor. Ongeldige toegangen zijn opzettelijk in de sketch ingebouwd, maar worden in het voorbeeld in eerste instantie weggecommenteerd. Als we de commentaartekens `//` verwijderen en het geheel nogmaals compileren, kunnen we de foutmeldingen die door de compiler worden gegenereerd bestuderen.

We moeten ons ervan bewust zijn dat de overervende objecten de overgeërfd objecten gebruiken alsof ze een deel van zichzelf zijn. Elk object heeft echter zijn eigen individuele dataset. De relatie is bijna alsof de code van `Virt_M` is gekopieerd en geplakt in `Volt_M` en `Amp_M`. Het grootste nadeel van dat soort relatie (afgezien van het extra werk en de omvangrijkere code) zou zijn wanneer er wijzigingen worden aangebracht in `Virt_M`. Alle andere klassen die `Virt_M` bevatten, moeten dan ook worden gewijzigd.

Een klasse kan ook een vriend zijn

In het laatste voorbeeld zagen we dat het vanwege inkapseling niet mogelijk is om het multiplex-adres van een object te wijzigen. Vooral bij het programmeren (vrijwel) op hardwareniveau is het echter vaak wenselijk om over veel toegangsopties te beschikken. Als we bijvoorbeeld alle multiplex-adressen opeenvolgend willen adresseren tijdens het debuggen, moeten we voor elk adres een object maken. Een andere mogelijkheid zou zijn om het object na de oproep te vernietigen en het vervolgens opnieuw te maken. We zouden de basisklasse natuurlijk vanaf het begin zo kunnen formuleren dat de toegang openbaar is. Dit zou echter betekenen dat

Listing 4. Klasse-declaratie van Virt_M met Debug_M als 'Friend' klasse (les_4)

```
class Virt_M
{
    friend class Debug_M;                                //the class Debug_M gets
                                                        //access to the private members
                                                        //of Virt_M

private:                                              //internal members of the class
    float value=0;                                     //virtual measurement value, set to 0
    byte mux_adr= MUX_INVALID;                         //mux address for simulation

public:                                               //members also accessible from outside
    Virt_M (byte mux, float ini_m);                   //constructor with initial values
    void set_Value (float val);                       //method writes the virtual measurement
    float get_Value ();                               //method reads the virtual measurement
    byte get_Mux ();                                 //method reads mux address
    String s="public: DemoString";                  //for demo only

};                                                       //end of the class declaration
```

Listing 5. Declaratie van het structuur-datatype t_result (les_5).

```
struct t_result      //type for returning data
{
    byte mux;          //mux adress
    String s;           //a string
    float val;          //measurement value of type float
};
```

Listing 6. Attribuut-declaratie van de Dev_M klasse (les_5).

```
class Dev_M
{
private:
    Volt_M *ptr_v;                                //pointer to an object of type Volt_M
    Amp_M *ptr_a;                                 //pointer to an object of type Amp_M
    t_result result;                             //for returning data
    const String SP=( "Power/ W : ");
    const String SR= ("Resist/ Ohm: ");
```

een van de voordelen van OOP wordt opgegeven, namelijk dat de gegevens tot op zekere hoogte beschermd zijn. *Les_4* toont een nette oplossing voor dit probleem. We overerven de basisklasse **Virt_M** naar een nieuwe klasse met de naam **Debug_M** en promoveren deze tot friend in de basisklasse. De functie van **Debug_M** is identiek aan die van **Volt_M** met de uitzondering dat deze ook toegang heeft tot de private leden van **Virt_M**. Om het multiplex-adres te wijzigen, kunnen we de **set** methode in **Debug_M** gebruiken die als public is gedeclareerd. **Listing 4** toont de declaratie van een **friend** klasse. De klasse **Virt_M** bevat geen methode om het multiplexadres in te stellen. In plaats daarvan krijgt de constructor twee waarden voor het initialiseren van de klassekenmerken. Een van deze waarden is het multiplexadres. De parameters worden doorgegeven wanneer de basisklasse en de subklassen worden geïnstantieerd.

Opnieuw toont de uitvoer van de **setup** routine de resultaten van onze inspanningen. **Friend** klassen worden zelden gebruikt, hoewel ze het voordeel bieden dat ze gebruik maken van bestaande methoden en de gewenste functionaliteit bieden zonder het beveiligingsconcept van inkapseling bij correct gebruik in gevaar te brengen.

Een klasse die toegang heeft tot een methode van een andere klasse

In de laatste voorbeelden hebben we vermogens- en weerstands berekeningen uitgevoerd in het hoofdprogramma. We lezen de attributen van **Volt_M** en **Amp_M** via hun methoden. Nu gaan we dit doen in een aparte klasse. *Les_5* definieert de nieuwe klasse **Dev_M**, die toegang heeft tot methoden van de objecten **Volt_M** en **Amp_M**. Hiervoor worden, wanneer een instantie van **Dev_M** wordt

Listing 7. De constructormethode van Dev_M slaat de pointer op (les_5).

```
// the constructor passes the pointers to the objects of Volt_M, Amp_M
// -----
Dev_M::Dev_M (Volt_M *ptrv, Amp_M *ptr)
{
    ptr_v= ptrv;           //pointer to Volt_M
    ptr_a= ptr;            //pointer to Amp_M
}
```

Listing 8. Definitie van de get_Powr-methode van Dev_M (les_5).

```
t_result Dev_M::get_Powr ()      //calculate power
{
    result.s= sP;             //name, unit
    result.val= ptr_v -> get_Value () * ptr_a -> get_Value ();
    return result;            //return via type t_result
}
```

Listing 9. Instantiëren van de objecten, pointers naar de objecten overdragen (les_5).

```
Volt_M My_Volt_1 (V_MUX0, 1.00);          //object of type Volt_M
Volt_M My_Volt_2 (V_MUX1, 2.00);          //object of type Volt_M
Amp_M My_Amp_1 (A_MUX0, 1.10);            //object of type Amp_M
Amp_M My_Amp_2 (A_MUX1, 2.10);            //object of type Amp_M
Dev_M My_Dev_1 (&My_Volt_1, &My_Amp_1);    //object of type Dev_M, passing pointers
Dev_M My_Dev_2 (&My_Volt_2, &My_Amp_2);    //object of type Dev_M, passing pointers
```

gemaakt, verwijzingen naar de aan te roepen objecten door de constructor doorgegeven aan Dev_M.

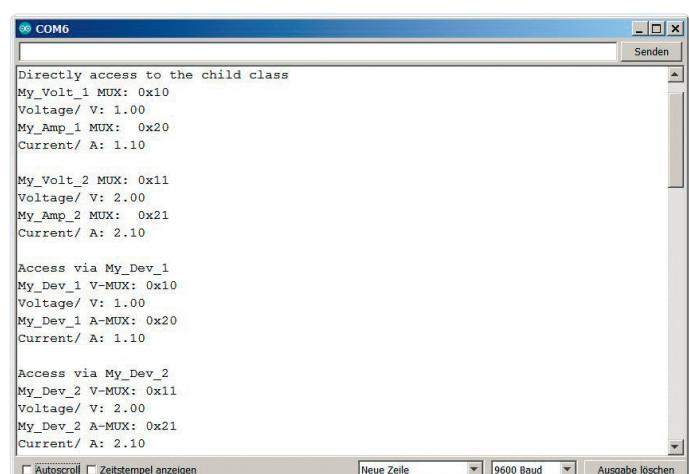
Eerst maken we echter een gestructureerd datatype met behulp van **struct** dat uit verschillende elementen bestaat die de resultaten bevatten. Het datatype heeft de naam **t_result** (**listing 5**). Daarop volgt de declaratie van de **Dev_M** klasse. Hier worden de pointers **ptr_v** en **ptr_a** toegewezen die worden gebruikt in de klassen **Volt_M** en **Amp_M**. De ***** operator geeft aan dat de erop volgende naam moet worden geïnterpreteerd als een pointer (**listing 6**). De constructormethode slaat de doorgegeven pointers op bij de attributen (**listing 7**).

De **Dev_M** methode kan nu de pointers gebruiken om toegang te krijgen tot de externe objecten. Dat wordt gedaan met behulp van de instructie **pointername -> methodname**. De **->** operator geeft toegang tot een methode met behulp van een pointer in plaats van de **.** (punt) operator (**listing 8**).

Om een instantie van **Dev_M** te kunnen maken, is het nodig om vooraf nieuwe instanties van de indexklassen te initialiseren. Deze moeten van het type **Volt_M** en **Amp_M** zijn. Bij het instantiëren van **Dev_M** voeren we de verwijzingen naar deze objecten in de constructor-parameterlijst in. Hiervoor gebruiken we de operator **&** (**listing 9**). De uitvoer van de **Setup** routine in **figuur 3** in onze applicatie laat zien dat de methode-aanroep van de **My_Dev_1** en **My_Dev_2** objecten alle relevante resultaten opleveren.

Meervoudige overerving

In het vorige voorbeeld moesten we eerst twee afzonderlijke instanties maken voordat we de methoden van **Dev_M** konden gebruiken. Er zijn toepassingen waar dit zeker zinvol is, maar voor ons is het vooral een introductie in het gebruik van pointers naar objecten. Sommigen vragen zich misschien af of basis- en afgeleide klassen simpelweg kunnen worden geërfd door een andere klasse. Dit is



Figuur 3. Uitvoer van de setup-routine van les_5.

Listing 10. Expliciete verwijzing naar een lid in geval van dubbelzinnigheid (les_6).

```
t_result Dev_M::get_Volt ()           //read the current voltage
{
    result.s= Volt_M::get_Unit ();      //call the Volt_M method
    result.val= Volt_M::get_Value ();   //call the Virt_M method
    return result;
}
```

Listing 11. Ambiguïteit van de get_Mux-methode (les_6).

```
Serial.print ("My_Dev_1, Volt MUX: 0x"),
Serial. println (My_Dev_1.Volt_M::get_Mux(), HEX); //method of base class
Serial.print (My_Dev_1.get_Volt().s);
Serial.println (My_Dev_1.get_Volt().val);           //initial value of instantiation of My_Dev_1
Serial.println (');
```

Listing 12. Een methode gebruiken om de dubbelzinnigheid op te lossen (les_6).

```
Serial.print ("My_Dev_2, Volt MUX: 0x");
Serial.println (My_Dev_2.get_VMux(),HEX);           //method of the inherited class
Serial.print (My_Dev_2.get_Volt().s);
Serial.println (My_Dev_2.get_Volt().val);
Serial.println (');
```

precies wat is geïmplementeerd in *les_6*: meervoudige overerving. De twee afgeleide klassen *Volt_M* en *Amp_M* worden overgeërfd door de basisklasse *Dev_M*. Omdat de afgeleide klassen *Virt_M* bevatten, heeft *Dev_M* toegang tot leden van alle klassen.

Om eenvoudig de complexe resultaten van de *Dev_M* methoden te retourneren, declareren we nogmaals het structurgegevenstype *t_result*. Wanneer de klasse is gemaakt, worden de overgeërfde klassen weergegeven (gescheiden door komma's) na de *:* operator:

```
class Dev_M: public Volt_M, public Amp_M
```

Met de volgende constructor-declaratie worden de parameters die moeten worden doorgegeven aan de overervende klassen, toegewezen in een parameterlijst:

```
Dev_M (byte v_x, float i_v, byte a_x, float i_a);
// the constructor of class Dev_M
```

De definitie van de constructormethode wijst vervolgens de parameters toe aan de overdrachswaarden van de overervende klassen. Bij het instantiëren van *Dev_M* worden de constructormethoden *Volt_M* en *Amp_M* aangeroepen en worden de parameterattributen van deze klassen toegewezen.

```
Dev_M::Dev_M (byte v_x, float i_v, byte a_x, float i_a):
V Volt_M (v_x, i_v), Amp_M (a_x, i_a) {}
```

Zoals eerder vermeld, hebben we toegang tot alle attributen en methoden van de overervende klassen via *Dev_M*, op voorwaarde dat de toegangsspecificatie dit toestaat. *Dev_M* biedt ons geschikte methoden om dit te doen. Er doet zich hier echter een probleem voor: aangezien de twee overgeërfde klassen zelf zijn afgeleid van de *Virt_M* klasse, hebben ze methoden en attributen met dezelfde

naam. Dat leidt tot onduidelijkheden. Om dit op te lossen gebruiken we de scope resolution operator *::* om expliciet aan te geven welk lid van welke klasse we bedoelen (**listing 10**).

Onze klassen zijn nu gedefinieerd en we kunnen er objecten van instantiëren. In principe zijn de methoden van *Dev_M* volledig voldoende om een applicatie te schrijven. Voor een beter begrip maken we ook meer instanties van de basisklasse en de afgeleide klassen. Omdat bij instantiatie aan elk object een ander multiplex-adres wordt toegewezen, kunnen we zien welk object momenteel wordt geadresseerd.

Nadat we de applicatie hebben gestart, toont de seriële monitor de resultaten van de methode-aanroepen. Het uitvoeren van de *Setup* routine is interessant. De attributen van de basisklasse worden op verschillende manieren gelezen. Via het multiplexadres van het *My_Dev_1* object zal de methode van de basisklasse worden gebruikt. Zoals we bij de methodedefinities van *Dev_M* hebben opgemerkt, doet zich hier ook het probleem van ambiguïteit voor. Om het multiplex-adres te lezen, moeten we expliciet specificeren of we het adres van de spanning of van de stroom bedoelen. De methode wordt in beide gevallen *get_Mux* genoemd (**listing 11**). We lossen het probleem op door de klasse-referentie te specificeren.

Het is natuurlijk veel duidelijker en uiteindelijk eenvoudiger om te voorzien in een eigen methode in de afgeleide klassen *Volt_M* en *Amp_M* (**listing 12**). In het voorbeeld heten deze methoden respectievelijk *get_VMux* en *get_AMux*. Deze worden gebruikt bij het lezen van de attributen van *My_Dev_2*.

De *sizeof* operator wordt gebruikt om het aantal bytes in de datavelden van ons object te bepalen. De basisklasse reserveert 11 bytes, terwijl klassen die hiervan afgeleid zijn 17 bytes gebruiken. Omdat de basisklasse in elke afgeleide klasse is opgenomen, nemen de afgeleide klassen zelf elk 6 bytes in beslag. *Dev_M* erft twee keer 17 bytes en heeft zelf 22 bytes nodig; in totaal zijn dat 56 databytes (**figuur 4**).

Iets lekkers voor Arduino-fans

Ik weet zeker dat er Arduino-fans zijn die regelmatig een van de Arduino-methoden in hun programma willen integreren. Misschien heeft u een eigen display ontwikkeld en wilt u daar naar toe kunnen schrijven met de werkelijk krachtige `print` methode. Zoals reeds gesuggereerd, kan dit worden bereikt door een klasse te maken die erft van de klasse `Print`. `Les_7` is een sketch die laat zien hoe dat gedaan kan worden. Onze klasse `My_Print_C` erft alle methoden van `Print` en overschrijft de oorspronkelijke `write` methode, die normaal gesproken weer te geven data naar specifieke hardware overdraagt. De `write` methode ontvangt een enkel teken van de `print(ln)` methode als argument en stuurt die vervolgens naar het display via een hardware-interface zoals de I²C-bus. Ons voorbeeld gebruikt geen hardware, dus de overgedragen tekens worden naar een string geschreven, waarvan de inhoud kan worden bekijken op de seriële monitor.

Omdat de `print(ln)` methoden overladen zijn, maakt het nauwelijks uit wat voor soort waarden we kunnen 'printen'. Dat is alles. Nu kunnen we `print` gebruiken om karakters naar ons eigen display te schrijven.

Klaar

Ik hoop dat dit artikel u een goede introductie heeft gegeven in de principes van objectgeoriënteerd programmeren. We hebben gekeken naar klasse-bibliotheken en hoe we deze in onze applicaties kunnen integreren. In het laatste voorbeeld hebben we getoond dat het overerven en wijzigen van externe klassen geen probleem is. Hoewel er nog veel te leren valt, zouden de basisprincipes voor het schrijven van uw eigen krachtige objectgeoriënteerde programma's nu allemaal aanwezig moeten zijn.

```
COM9
Setup
My_Volt_0, MUX: 0x10
Voltage/ V: 0.10

My_Amp_0, MUX: 0x20
Current/ A: 0.20

My_Dev_1, Volt MUX: 0x11
Voltage/ V: 1.00

My_Dev_1, Amp MUX: 0x21
Current/ A: 1.10

My_Dev_2, Volt MUX: 0x12
Voltage/ V: 2.00

My_Dev_2, Amp MUX: 0x22
Current/ A: 2.20

Number of data bytes of My_Virt: 11
Number of data bytes of My_Volt_0: 17
Number of data bytes of My_Amp_0: 17
Number of data bytes of My_Dev_1: 56

Autoscroll  Zeitstempel anzeigen  Neue Zeile  9600 Baud  Ausgabe löschen
```

Figuur 4. Uitvoer van de setup-routine van `les_6`.

Ik hoop dat u overtuigd bent van de voordelen van OOP ten opzichte van procedureel programmeren. De concepten zijn in het begin misschien moeilijk te begrijpen, vooral als u vertrouwd bent met procedureel programmeren. Maar geen angst – het gezegde luidt niet voor niets 'oefening baart kunst'! ◀

200563-03

GERELATEERDE PRODUCTEN

- ▶ **Arduino Uno R3**
www.elektor.nl/arduino-uno-r3 E
- ▶ **Elektor Arduino Electronic Bundle**
www.elektor.nl/elektor-arduino-electronics-bundle

Een bijdrage van

Idee en tekst:
Roland Stiglmayr

Redactie: **Rolf Gerstendorf**
Layout: **Giel Dols**

Vragen of opmerkingen?

Hebt u vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur of naar de redactie van Elektor, via redactie@elektor.com.

WEBLINKS

- [1] **Projectpagina bij dit artikel:** www.elektormagazine.nl/200563-03
- [2] **C++ Programming:** https://en.wikibooks.org/wiki/Subject:C%2B%2B_programming_language

Oost West Lab Best

...waar Kurt Diedrich synthesizers bouwt

Kurt Diedrich en Eric Bogers

Omstreeks 1964 verscheen de eerste Moog-synthesizer [1] – een (geheel analoog) modulair opgebouwd elektronisch instrument dat de muziekwereld met zijn nieuwe klankmogelijkheden revolutioneerde. In diverse uitvoeringen is de Moog tot 1980 geproduceerd. Deze synthesizer vormde de inspiratie voor veel muziekprojecten – waaronder de Elektuur Formant uit 1977, die nog altijd veel populariteit geniet [2]. In het thuislaboratorium van Kurt Diedrich, waar we in deze aflevering een bezoekje aan brengen, is ook een Moog-achtige synthesizer geboren – een project dat voortdurend verder wordt ontwikkeld en dat uitermate geschikt is voor zelfbouw, niet in de laatste plaats omdat er uitsluitend goed verkrijgbare en betaalbare componenten voor worden gebruikt.



Figuur 1. Het elektronicalaboratorium van Kurt Diedrich...



Figuur 2. ...die hier aan de laatste versie van zijn synthesizer werkt.

Bij de wat oudere lezers zal de naam Kurt Diedrich mogelijk een belletje doen rinkelen – en terecht. Laten we Kurt zelf aan het woord:

“Al tijdens mijn studie geologie (van 1972 tot 1980) merkte ik dat ik elektronica eigenlijk interessanter vond dan geologie. In 1973 bouwde ik mijn eerste synthesizer (er zouden er nog meer volgen). Uit deze tijd stammen ook mijn eerste contacten met (toen nog) Elektuur. In 1981 verscheen mijn eerste boek (over synthesizers) bij Frech Verlag [3].

Na mijn studie heb ik nog enige tijd als assistent aan de universiteit gewerkt, tot ik in 1981 bij Elektuur in Beek een baan als redacteur en ontwerper kreeg. Dat was het begin van een heel mooie tijd met veel fijne collega’s. Ik heb daar onder andere een CEM-Curtis synthesizer ontwikkeld, naast een groot aantal andere schakelingen op het gebied van meettechniek en muziekelektronica. Vanaf 1985 werkte ik ook als ontwerper en schrijver voor Elex – het tijdschrift voor de beginnende elektronicus m/v; tot ik in 1987 in het kader van bezuinigingen helaas het veld moest ruimen.

Daarna werkte ik als schrijver van technische handboeken voor diverse bedrijven, de laatste 17 jaar bij HEAD Acoustics in Herzogenrath bij Aken.

Na het bereiken van de pensioengerechtigde leeftijd vroeg ik me af hoe ik al die vrije tijd nuttig en aangenaam zou kunnen besteden. En toen bedacht ik dat ik vroeger synthesizers heb gebouwd – in 2014 heb ik de draad weer opgepakt. Mijn doel is goed functionerende (lees: klinkende) instrumenten te ontwikkelen met gewone, goed verkrijgbare en goedkope onderdelen. En ik geloof dat me dit heel behoorlijk lukt.”

Kurt verkeert in de benijdenswaardige omstandigheden dat hij over verschillende werkplekken kan beschikken. **Figuur 1** toont zijn goed ingerichte en ruime elektronicalaboratorium – en in **figuur 2** is Kurt bezig met de jongste incarnatie van zijn synthesizerproject. In **figuur 3** is Kurts computerkamer annex bibliotheek te zien, en **figuur 4** toont de werkplaats voor het betere mechanische werk. Zeg eens eerlijk – dat is toch om jaloers op te worden...

Het zou ver buiten het kader van deze artikelreeks vallen om diep op het synthesizerproject van Kurt in te gaan – we volstaan hier met een beknopte (en noodgedwongen onvolledige) beschrijving van het blokschema van de momentele versie (**figuur 5**), met de opmerking dat dit een project is dat voortdurend verder ontwikkeld wordt.

“Het blokschema toont de basisstructuur van de synthesizer. Elke groene rechthoek staat voor een een print. Elke print op zijn beurt bevat één enkele module, met uitzondering van de drievoudige VCA-print, waarop ook de schakelingen van de ruisgenerator en de mixer (die weinig ruimte in beslag nemen) een plaatsje hebben gevonden.

Er wordt een externe schakelende voeding met een uitgangsspanning van minimaal 20 V_{DC} gebruikt. Op die manier blijft de gevaarlijke netspanning buiten het instrument. De benodigde 2x12 V wordt vervolgens op een interne voedingsprint gegenereerd met behulp van een DC/DC-converter (2x500 mA). Als schakelende voeding kan bijvoorbeeld de voedingseenheid van een afgedankte laptop worden gebruikt, mits die de vereiste uitgangsspanning levert.

De MIDI-converter wordt tot leven gebracht via een Arduino Nano. Deze schakeling is geen eigen ontwikkeling; wel heb ik de print ervoor ontworpen.

De retrigger-eenheid is van groot belang voor het legato-spelen (dat komt er op neer dat een toets al wordt ingedrukt voordat de vorige toets is losgelaten – iets wat bij snel spel vrijwel onvermijdelijk is. Zonder legato-mogelijkheid kan een synthesizer als deze eigenlijk slechts met één vinger bespeeld worden – en dat klinkt niet bijzonder goed...)

De delayed vibrato-eenheid zorgt voor een langzaam oplopend frequentievibrato wanneer een toets langer dan 1 à 2 seconden wordt ingedrukt. Een interessant effect dat sommige muziekstukken een bijzondere charme geeft. De LFO heeft onder andere een uitgang voor een symmetrische blokgolf, wat tot interessante geluiden leidt.

Voor virtuoos spel op de synthesizer is natuurlijk een keyboard nodig. Het is het enige deel van het hier gepresenteerde instrument dat kant-en-klaar moet worden gekocht. De synthesizer is ontworpen voor gebruik met een standaard MIDI-toetsenbord. De handel biedt een scala aan goedkope MIDI-toetsenborden. Ik gebruik het ‘Easy Key 49’-model van SwissSonic.

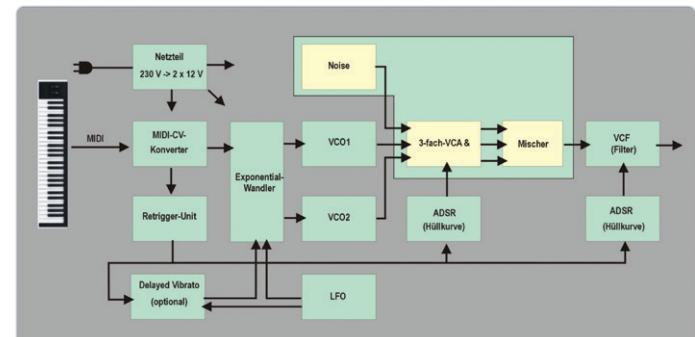
De meeste elektronische toetsinstrumenten (in hun totaliteit ook wel keyboards genoemd) hebben een extra MIDI-uitgang die ook kan worden gebruikt om de hier beschreven synthesizer aan te sturen.”



Figuur 3. De computerkamer.



Figuur 4. Een werkbank voor het betere mechanische werk.



Figuur 5. Blokschema van de synthesizer.

Tot zover een heel beknopte beschrijving; geïnteresseerden kunnen voor de complete beschrijving inclusief schema's en dergelijke rechtstreeks contact opnemen met Kurt Diedrich (zie **kader**). ▶

200681-02

Vragen of opmerkingen?

Hebt u vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur via subroutine-sy@gmx.de of naar de redactie van Elektor via redactie@elektor.com.

Een bijdrage van

Tekst, foto's en blokschema:
Kurt Diedrich

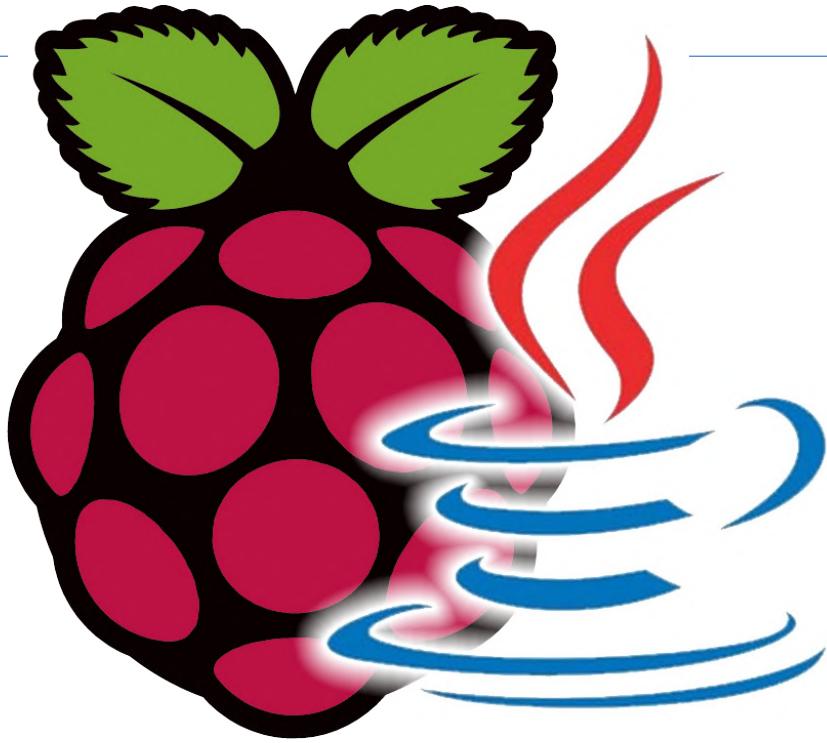
Redactie: **Eric Bogers**
Layout: **Giel Dols**

WEBLINKS

- [1] Moog synthesizer: https://en.wikipedia.org/wiki/Moog_synthesizer
- [2] Retrotronica: Formant Synthesizer: www.elektrormagazine.nl/magazine/elektor-200804/15464
- [3] Boeken van Kurt Diedrich: www.subroutine.info/bücher/

Java op de Raspberry Pi

deel 1: GPIO's



Frank Delporte (België)

De programmeertaal Java bestaat al lang, maar hij is nog steeds heel geschikt voor het ontwikkelen van code voor moderne rekenplatforms zoals de Raspberry Pi. In deze serie willen we de mogelijkheden tonen. In het eerste deel geven we enige achtergrondinformatie over de taal en gaan we onderzoeken hoe Java GPIO-pinnen kan inlezen en aansturen.

Java: een korte inleiding

Java is één van die programmeertalen die al heel lang meedoen. De eerste versie kwam in 1995 uit [1], hetzelfde jaar waarin JavaScript, PHP en Qt het licht zagen. Python is iets ouder, dat deed zijn intrede in 1991.

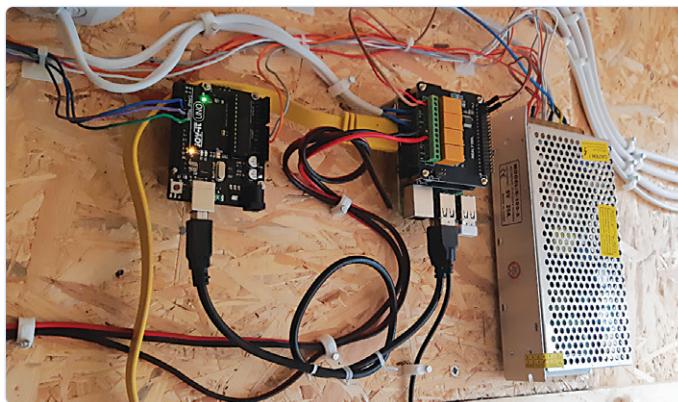
Hoewel de handelsnaam 'Java' van Sun Microsystems naar Oracle is gegaan, vindt de ontwikkeling sinds 2020 via GitHub als open source-project plaats [2]. Sinds 2018 worden jaarlijks twee nieuwe releases van Java uitgebracht, zodat correcties en nieuwe features regelmatig onze machines bereiken. Dat geldt ook voor verbeteringen voor embedded platforms zoals de Raspberry Pi.

Bij sommigen lijkt het gevoel te bestaan dat 'Java dood is', maar de vele conferenties (ook al zijn die in deze Corona-tijden virtueel) en Java-gerelateerde projecten bewijzen dat de taal nog altijd

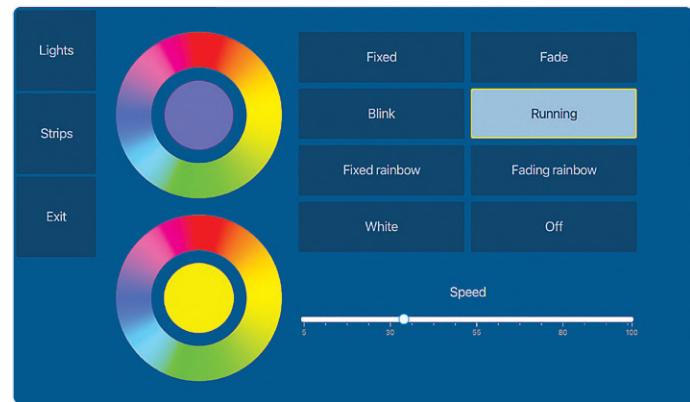
springlevend is. De belangrijkste bijdragen aan de ontwikkeling van Java komen van een lange lijst van bekende bedrijven zoals Oracle, Microsoft, SAP, Google en anderen [3]. Er circuleert ook een gerucht dat de helft van de Microsoft Azure-cloud onder Java draait! Java-distributies zijn tegenwoordig beschikbaar bij verschillende open source (jdk.java.net, adoptopenjdk.net) en commerciële leveranciers (Azul, BellSoft, Oracle en anderen).

Java op de Raspberry Pi?!

Maar de Raspberry Pi was toch ontworpen om er Python op te draaien? Misschien, maar dat betekent niet dat hij geen andere talen aan kan. En, hoewel dit artikel over Java gaat, willen we zeker niet beweren dat Java beter is dan Python, C of een andere taal! Ieder project stelt zijn eigen eisen, waar een specifieke taal



Figuur 1. Het hart van het aanraakscherm van de drumcabine.



Figuur 2. De aanraakscherm-interface bestuurt de relais en de LED-strips.

de perfecte oplossing voor kan zijn of waar het ontwikkelteam de meeste ervaring mee heeft.

Zelf ben ik Java-ontwikkelaar en ik was nieuwsgierig of ik mijn kennis zou kunnen toepassen op de Raspberry Pi. Ik kwam op dat idee toen mijn eerste poging mislukte om in Python een pong-spel te bouwen. Ik vond de gebruikersinterface erg tegenvallen. Gelukkig kun je gebruikersinterfaces genereren met JavaFX, een onafhankelijk project [4, 5] dat Java uitbreidt met een framework om GUI's (grafische gebruikersinterfaces) te bouwen. Het biedt alle bekende basiscomponenten (drukknoppen, labels, tabellen, grafieken) en er is een lange lijst van gratis open source-bibliotheken die nog meer mogelijkheden bieden.

Ik vond de perfecte aansluiting tussen Java en de Raspberry Pi, toen ik een touchscreen-interface wilde bouwen voor de drumcabine van mijn zoon (**figuren 1 en 2**). In combinatie met een relaiskaart, een Arduino en enkele LED-strips werd dit de basis voor mijn eerste stappen in een nieuwe wereld van embedded programmeren in Java.

De voorbereiding

Voor de experimenten die ik in dit artikel met u wil delen, hebt u een recente Raspberry Pi met een ARMv7- of ARMv8-processor nodig. Oudere boards met een ARMv6 hebben een andere interne layout waar standaard-Java niet op werkt. Als u Java wilt gebruiken op zo'n ARMv6-kaart, is er wel een oplossing die met de Azul Zulu JDK [6] werkt. En als u geen zin hebt om veel te typen, kunt u gebruik maken van een GitHub-repository [7] waar alle codevoorbeelden uit deze serie beschikbaar zijn.

Bereid als eerste de SD-kaart voor met het 'Imager'-tool [8] en kies "Raspberry Pi OS Full (32-bit)" (**figuur 3**). Als u de Raspberry Pi hebt gestart, opent u een terminal-venster en typt u `java -version` om te controleren of u versie '11' hebt. U zult zien dat OpenJDK is voorgeïnstalleerd in deze volledige versie van het OS:

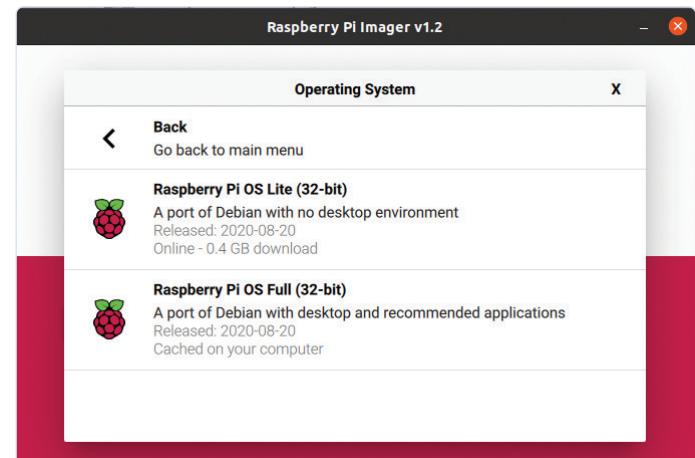
```
$ java -version
openjdk version "11.0.9" 2020-10-20
OpenJDK Runtime Environment (build
 11.0.9+11-post-Raspbian-1deb10u1)
OpenJDK Server VM (build 11.0.9+11-post-Raspbian-
 1deb10u1, mixed mode)
```

Visual Studio Code

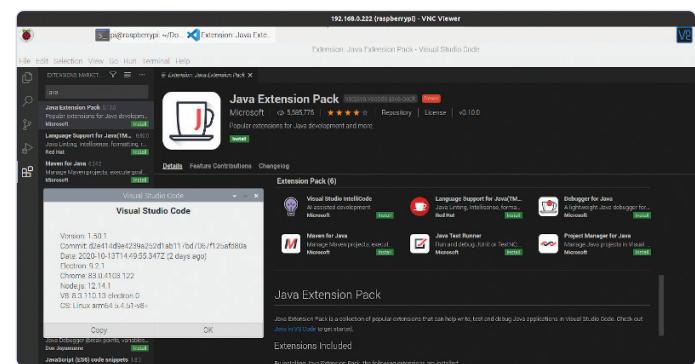
U kunt uw Java-code ontwikkelen, testen en uitvoeren op een PC voordat u die overbrengt naar de Raspberry Pi. Maar er is ook een andere manier: Visual Studio Code [9]. Er is voor deze gratis IDE (Integrated Development Environment) van Microsoft een groot aantal uitbreidingen beschikbaar, wat het tot een perfect gereedschap maakt voor elk programmeerproject. Er is een versie beschikbaar voor 32-bits ARM-systemen (zoals Raspberry Pi OS), voor 64-bits ARM-systemen (het nieuwe Raspberry Pi OS, dat nog in ontwikkeling is) en zelfs voor Ubuntu Desktop [10]. Er bestaat ook een 'Java Extension Pack' dat verschillende Java-uitbreidingen toevoegt aan de IDE zodat het een volledige Java-IDE (**figuur 4**) wordt!

Experimenteren met Hello World!

Laten we ons eerste Java-programma op de Raspberry Pi proberen. Met Visual Studio Code, een teksteditor of een terminalvenster maken we een nieuw tekstbestand aan met de naam "HelloWorld.java" en met de volgende inhoud:



Figuur 3. Raspberry Pi Imager tool: de beste manier om een SD-kaart voor te bereiden.



Figuur 4. Visual Studio Code is te gebruiken op een Raspberry Pi en heeft een Java Extension Pack in de aanbieding.

```
public class HelloWorld {
    public static void main(String args[]) {
        System.out.println("Hello World!");
    }
}
```

Al onze code maakt deel uit van de klasse `HelloWorld`. De conventie is dat die klasse dezelfde naam heeft als het bestand. Een Java-programma start met de methode `public static void main(String args[])`. Het enige dat we hier doen, is "Hello World!" afdrukken als uitvoer van het programma. Omdat we met Java 11 werken, kunnen we zo'n eenvoudig programma draaien zonder dat we het hoeven te compileren. Geef in de terminal, in de map waar uw Java-bestand staat, het commando `java HelloWorld.java`. U krijgt dan het volgende resultaat:

```
pi@raspberrypi:~/elektor $ java HelloWorld.java
Hello World!
```

Natuurlijk is `print()` in Python korter dan `System.out.println()` in Java; laten we dat echter beschouwen als een 'jeugdzone'.

Listing 1. De code voor CPUtemp.java om de processortemperatuur van de Raspberry Pi in te lezen [7].

```

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.util.List;
import java.util.ArrayList;

public class CPUtemp {
    private static final String FILE = "/sys/class/thermal/thermal_zone0/temp";
    private static final List<Integer> values = new ArrayList<>();

    public static void main(String[] args) throws InterruptedException {
        while(true) {
            checkTemp();
            Thread.sleep(1000);
        }
    }

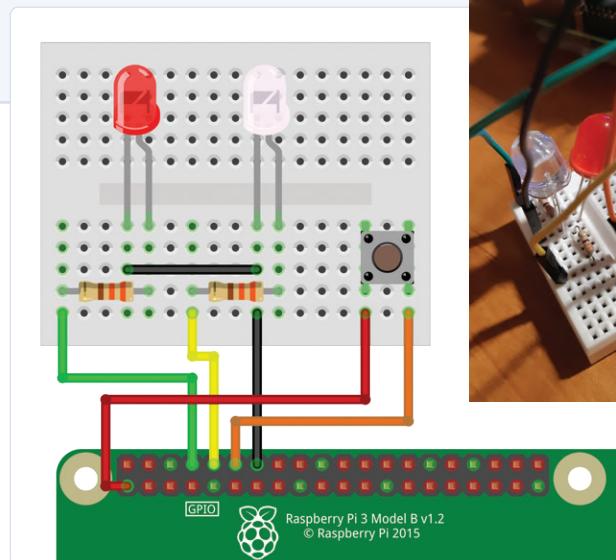
    private static void checkTemp() {
        try (BufferedReader br = new BufferedReader(new FileReader(FILE))) {
            int value = Integer.valueOf(br.readLine());
            values.add(value);
            int total = values.stream().mapToInt(Integer::valueOf).sum();
            System.out.println("Now: " + value
                + " - Average: " + (total / values.size())
                + " - Number of measurements: " + values.size());
        } catch (Exception ex) {
            System.err.println("Error during temperature check:
                + ex.getMessage());
        }
    }
}

```

De temperatuur van de CPU uitlezen

Veel systeemparameters, inputs en outputs, zijn toegankelijk via de map `/sys/` van het Linux-bestandssysteem. De opgeslagen waarden kunnen als gewoon tekstbestand worden gelezen. Laten we eens kijken of we de temperatuur van de Raspberry Pi-processor kunnen vinden in één van de bestanden in de map `sys`. De naamgeving in het bestandssysteem is niet altijd even duidelijk, en het is ook niet altijd eenvoudig om de uitgelezen waarden te interpreteren, maar het is een goede manier om iets te weten te komen over het Linux-systeem. Voer het volgende commando in:

```
$ ls -l /sys/
total 0
drwxr-xr-x  2 root root 0 Dec  2 15:44 block
drwxr-xr-x 29 root root 0 Feb 14 2019 bus
drwxr-xr-x 64 root root 0 Feb 14 2019 class
drwxr-xr-x  4 root root 0 Feb 14 2019 dev
drwxr-xr-x 10 root root 0 Feb 14 2019 devices
drwxr-xr-x  3 root root 0 Feb 14 2019 firmware
drwxr-xr-x  8 root root 0 Jan  1 1970 fs
drwxr-xr-x 12 root root 0 Jan  1 1970 kernel
drwxr-xr-x 122 root root 0 Feb 14 2019 module
drwxr-xr-x  2 root root 0 Dec 15 11:39 power
```



Figuur 5. Fritzing-layout (links) voor de LED's en de drukknop. Rechts een foto van de opbouw.

Vragen of opmerkingen?

Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur of naar de redactie van Elektor, via redactie@elektor.com.

Zo te zien is daar een hoop informatie te vinden! Laten we dat eens nader onderzoeken.

```
$ cat /sys/firmware/devicetree/base/cpus/cpu@0/  
    compatible  
arm,cortex-a72  
$ cat /sys/class/net/wlan0/address  
dc:a6:32:c5:b7:9d  
$ cat /sys/class/bluetooth/hci0/uevent  
DEVTYPE=host  
$ cat /sys/class/thermal/thermal_zone0/temp  
30667
```

Dat laatste zou best eens een temperatuur in graden Celsius kunnen zijn als we het door 1000 delen! Met een eenvoudig programma kunnen we die waarde elke seconde uitlezen en de gemiddelde temperatuur berekenen.

De code in **listing 1** maakt gebruik van wat meer Java-methoden dus moeten we beginnen met meerdere imports. De `io`-methoden worden gebruikt om een `sys`-file uit te lezen als een tekstfile. `List` en `ArrayList` gebruiken we om een lijst van alle gemeten waarden bij te houden. We roepen vanuit de methode `main()` een aparte methode `checkTemp()` aan om de temperatuur op te halen en die op te slaan in de lijst. Zo houden we de code netjes. Het is een goede gewoonte om elke functionaliteit te isoleren in een eigen methode. Zoals u ziet gebruiken we `Thread.sleep(1000)` om één seconde te wachten tussen de temperatuurcontroles. In onze methode wordt het gemiddelde berekend door de som van de lijst van waarden te berekenen met behulp van een stream. Streams werden geïntroduceerd in Java 8 en vormen een uiterst krachtige manier om met reeksen items te werken. Net als bij het eerste programma is de naam van de klasse gelijk aan de bestandsnaam: "CPUTemp.java". Ook dit is een eenvoudige Java-klasse zonder extra afhankelijkheden, dus we kunnen hem uitvoeren zonder te compileren. U kunt de uitvoer stoppen met 'Ctrl+c':

```
pi@raspberrypi:~/elektor $ java CPUTemp.java  
Now: 36998 - Average: 36998 - Number of measurements: 1  
Now: 34563 - Average: 35780 - Number of measurements: 2  
Now: 35537 - Average: 35699 - Number of measurements: 3  
Now: 36024 - Average: 35780 - Number of measurements: 4  
Now: 35537 - Average: 35731 - Number of measurements: 5
```

Een LED aansturen en een drukknop inlezen

We maken nog een Java-toepassing die uit één bestand bestaat. Deze kan twee LED's aan- en uitschakelen en de toestand van een drukknop inlezen. De bedrading is heel eenvoudig: we sluiten gewoon twee LED's en één drukknop aan, elk op een eigen GPIO. De drukknop werkt op 3,3 V, niet op 5,0 V, dus let op dat u de juiste voedingspin gebruikt! De LED's zijn verbonden met pinnen BCM 14 en 15 en de drukknop zit op BCM 18 (**figuur 5**).

Testen met de terminal

We kunnen de GPIO's aansturen vanuit de terminal met de ingebouwde commando's van Raspberry Pi OS. Daarmee testen we onze bedrading. Voer de volgende commando's in om pin 14 te configureren als uitgang (op) en maak hem hoog (dh) of laag (dl). Doe hetzelfde voor de andere LED (BCM 15):

```
$ raspi-gpio set 14 op  
$ raspi-gpio set 14 dh  
$ raspi-gpio set 14 dl
```

Op soortgelijke wijze kunnen we de drukknop testen: we configureren de pin als een ingang en vragen diens toestand op:

```
$ raspi-gpio set 18 ip  
$ raspi-gpio get 18  
GPIO 18: level=0 fsel=0 func=INPUT pull=DOWN  
$ raspi-gpio get 18  
GPIO 18: level=1 fsel=0 func=INPUT pull=DOWN
```

Als de drukknop is ingedrukt, krijgt 'level' de waarde 1.

En dan nu in Java

Op dezelfde manier als in "CPUTemp.java" gebruiken we de terminal-commando's in een Java-programma met behulp van `Runtime.getRuntime().exec(cmd)`. Met de gebruikers-inputklasse `Scanner` kunnen we het resultaat van het commando inlezen en dat gebruiken om de toestand van de drukknop te controleren. Dit is te zien in **listing 2**.

De code maakt gebruik van een enum `Action` om het verwerken van de commando's te vergemakkelijken. In zijn eenvoudigste vorm is een enum niet meer dan een lijst van voorgedefinieerde namen, maar in dit voorbeeld voegen we aan elke naam een waarde toe. Dat is een groot voordeel van het gebruik van enums, want ze maken de code veel beter leesbaar en vaak ook nog eenvoudiger. In de methode `doAction` wordt de enum `Action` gebruikt als een parameter en wordt een commando gegenereerd afhankelijk van de waarde van die parameter.

Het commentaar in de code zou voldoende moeten zijn om alles duidelijk te maken. Dit is gewoon eenvoudige Java-code om te illustreren hoe we de ingangs- en uitgangs-GPIO's kunnen gebruiken. Het ingewikkeldste deel is de methode `runCommand` die gebruik maakt van een combinatie van methoden om het commando uit te voeren, het resultaat te lezen en eventuele fouten af te handelen. Als het niet meteen helder is bij het lezen, dan komt dat wel goed als u de code gaat draaien!

Ook in dit voorbeeld wordt geen gebruik gemaakt van externe afhankelijkheden, dus we kunnen het uitvoeren zonder te compileren. De uitvoer geeft duidelijk aan wat er gebeurt, als het goed is ziet u LED's knipperen met verschillende tussenpozen. Als de LED's ophouden met knipperen, kunt u op de knop drukken, die wordt tien keer ingelezen met een interval van 1 seconde. De verwachte uitvoer is als volgt:



GERELATEERDE PRODUCTEN

- **F. Delporte, Getting Started with Java on the Raspberry Pi**
www.elektor.nl/19292
- **Raspberry Pi 4 Starter Kit**
www.elektor.nl/19427

Listing 2. Code voor RaspiGpio.java voor het besturen van de LED's en het inlezen van de drukknop [7].

```
import java.io.IOException;
import java.util.Scanner;

public class RaspiGpio {

    private static final String LED_1 = "14";
    private static final String LED_2 = "15";
    private static final String BUTTON = "18";

    private enum Action {
        OUTPUT_PIN("op"),
        INPUT_PIN("ip"),
        DIGITAL_HIGH("dh"),
        DIGITAL_LOW("dl");
        private final String action;
        Action(String action) {
            this.action = action;
        }
        String getAction() {
            return action;
        }
    }

    public static void main(String[] args) throws InterruptedException {
        // Configure the two LEDs as output pins
        doAction(LED_1, Action.OUTPUT_PIN);
        doAction(LED_2, Action.OUTPUT_PIN);

        // Configure the button as input pin
        doAction(BUTTON, Action.INPUT_PIN);

        // Blink the LEDs with different interval
        for (int i = 0; i < 20; i++) {
            System.out.println("Blink loop: " + i);
            if (i % 2 == 0) {
                doAction(LED_1, Action.DIGITAL_HIGH);
            } else {
                doAction(LED_1, Action.DIGITAL_LOW);
            }
        }
    }
}
```

```
$ java RaspiGpio.java
Executing: raspi-gpio set 14 op
Executing: raspi-gpio set 15 op
Executing: raspi-gpio set 18 ip
Blink loop: 0
Executing: raspi-gpio set 14 dh
Executing: raspi-gpio set 15 dh
Blink loop: 1
Executing: raspi-gpio set 14 dl
Executing: raspi-gpio set 15 dl
...
Executing: raspi-gpio get 18
Button check 0: GPIO 18: level=0 fsel=0 func=INPUT pull=DOWN
- PUSHED: false
...
Button check 3: GPIO 18: level=1 fsel=0 func=INPUT pull=DOWN
- PUSHED: true
...
```

Voorbereiding voor de volgende aflevering

We hebben nu wat basiskennis opgedaan als introductie tot Java op de Raspberry Pi. Ga vooral lekker experimenteren met de voorbeeldcode en lees alvast wat achtergrondinformatie door de links te volgen. Hopelijk maakt dat de volgende stappen in het tweede artikel beter te begrijpen. We gaan dan een volledige Java-toepassing maken en onze GPIO's koppelen met een internetpagina. ↗

200617-03

Een bijdrage van
Idee, tekst en afbeeldingen:
Frank Delporte

Redactie: **Stuart Cording**
Vertaling: **Evelien Snel**
Layout: **Giel Dols**

```

        if (i % 3 == 0) {
            doAction(LED_2, Action.DIGITAL_HIGH);
        } else {
            doAction(LED_2, Action.DIGITAL_LOW);
        }
        Thread.sleep(500);
    }
    doAction(LED_1, Action.DIGITAL_LOW);
    doAction(LED_2, Action.DIGITAL_LOW);

    // Read the button state 10 times
    for (int j = 0; j < 10; j++) {
        String result = runCommand("raspi-gpio get " + BUTTON);
        System.out.println("Button check " + j
            + ":" + result
            + " - PUSHED: " + (result.contains("level=1")));
        Thread.sleep(1000);
    }
}

private static void doAction(String pin, Action action) {
    runCommand("raspi-gpio set " + pin + " " +
        action.getAction().toLowerCase());
}

private static String runCommand(String cmd) {
    System.out.println("Executing: " + cmd);
    Scanner s = null;
    try {
        s = new Scanner(Runtime.getRuntime().exec(cmd).getInputStream())
            .useDelimiter("\A");
        return s.hasNext() ? s.next() : "";
    } catch (Exception ex) {
        System.err.println("Error during command: " + ex.getMessage());
        return "";
    } finally {
        if (s != null) {
            s.close();
        }
    }
}
}

```

WEBLINKS

- [1] Andrew Binstock, 'Java's 20 Years Of Innovation', **Forbes**: <http://bit.ly/3qelpVu>
- [2] OpenJDK project: <https://github.com/openjdk>
- [3] Liam Tung, Oracle: 'Programming language Java 14 is out with these 16 major feature improvements', **ZDNet**: <http://zd.net/35wVW2O>
- [4] OpenJFX project: <https://github.com/openjfx>
- [5] OpenJFX: <https://openjfx.io/>
- [6] Frank Delporte, 'How to install and use Java 11 and JavaFX 11 on Raspberry Pi boards with ARMv6 processor': <http://bit.ly/35yRrFc>
- [7] Bijbehorende codevoorbeelden op GitHub: <http://bit.ly/3i9bP4v>
- [8] Raspberry Pi Imager tool: <http://bit.ly/3i58seU>
- [9] Frank Delporte, 'Visual Studio Code on the Raspberry Pi (with 32 and 64-bit OS)': <http://bit.ly/3qeh9GN>
- [10] Visual Studio Code downloads voor ARM: <http://bit.ly/2LqUng3>

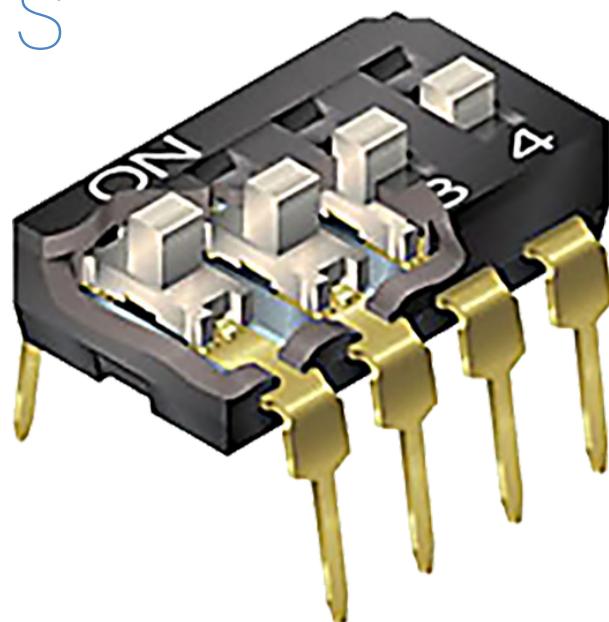
DIP-schakelaars

vreemde onderdelen

David Ashton (Australië)

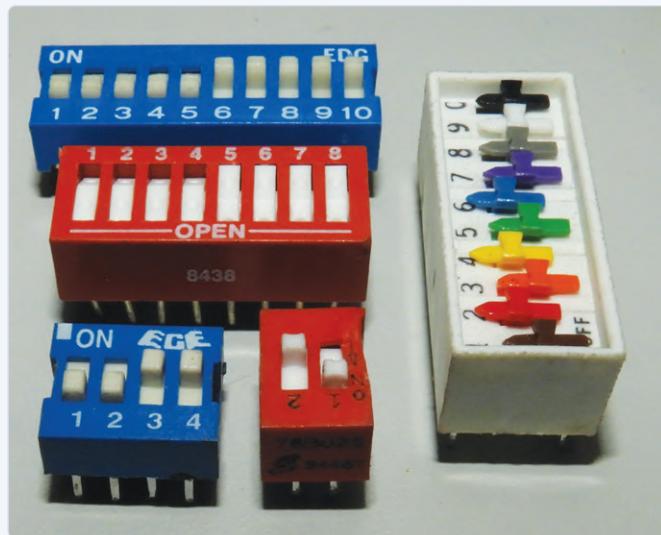
Wacht eens even – DIP-schakelaars zijn toch niet ‘vreemd’? Het zijn toch gewoon schakelaars? Welnu, het blijkt dat er ook vreemde (zelden gebruikte) DIP-schakelaars bestaan. Hier stellen we u een paar uit mijn verzameling voor, in volgorde van toenemende vreemdheid.

Voor het geval u het niet wist: DIP staat voor *Dual Inline Package*, en dat betekent dat dat de pinnen in twee parallelle rijen staan, zoals bij IC's. Bij de meeste bedraagt de afstand tussen de rijen 0,3 inch (zoals IC's), maar bij sommige is dat 0,2 inch. DIP-schakelaars worden gebruikt voor het instellen van opties, functies, adressen of wat dan ook waarvoor een robuuste, zelden gebruikte configuratiemogelijkheid nodig is. De meeste DIP-schakelaars zijn

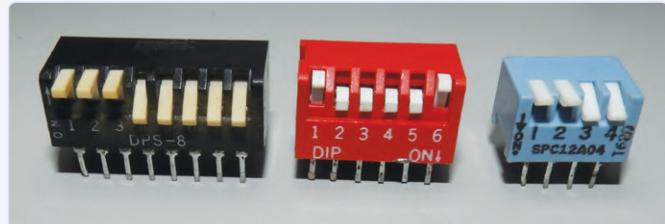


8-voudig maar zoals we nog zullen zien, zijn er veel verschillende typen in omloop.

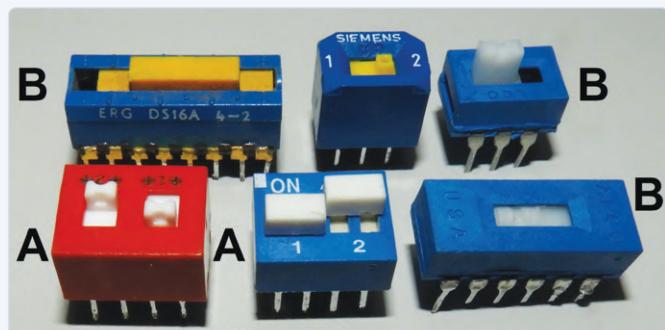
Laten we eerst als uitgangspunt een blik werpen op de standaard DIP-schakelaars die we allemaal kennen. Een groepsportret van een paar uit mijn verzameling heb, is te zien in **figuur 1**. Ze zijn in veel kleuren verkrijgbaar en vrolijken anderszins saaie printen aanzienlijk op.



Figuur 1. Een verzameling typische DIP-schakelaars.



Figuur 2. Piano-type DIP-schakelaars zijn gemakkelijker in te stellen.



Figuur 3. Dubbele en omschakeltypen.

Sommige hebben verzonken hendels van het schuiftype, en dat betekent dat u een paperclip of (erger) een pen moet zoeken (of een potlood, maar dat is een absolute afrader). Anderen hebben hendels die trots een stukje uitsteken zodat u die in principe met een vingernagel kunt omzetten. Sommige DIP-schakelaars, zoals die rechts in figuur 1, hebben kleurgedeerde hendels. Als u de kleurcode van weerstanden kent, kunnen deze gemakkelijk worden gelezen en ingesteld.

Sommige staan verticaal met de schakelaars aan de zijkant (**figuur 2**). Dat komt van pas om ze toegankelijk te maken via het achterpaneel van een apparaat. Bij zulke 'piano'-typen is het iets gemakkelijker een instelling met een vingernagel te wijzigen.

Nu betreden we onbekend(er) terrein. Er zijn dubbele typen (**figuur 3a**) waar bij bediening twee onafhankelijke schakelaars tegelijk veranderen. Er zijn ook DIP-wisselschakelaars (**figuur 3b**). De kleinere hebben twee wisselschakelaars (dus DPDT – dual-pole dual-throw) en de grotere hebben vier of meer wisselschakelaars. Die zijn handig om bijvoorbeeld verzwakkers in of uit te schakelen. Mijn verzameling is ook een paar keuzeschakelaars rijk (**figuur 4**). U selecteert/sluit daarmee één van acht of tien paar contacten. De blauwe (links) hebben een deksel (bij de onderste verwijderd); daar hebt u een kleine schroevendraaier nodig om de actuator naar een van de 10 posities te schuiven. Dus als één rij contacten gemeenschappelijk wordt gebruikt (bij sommige van deze schakelaars is dat af-fabriek al voor u gedaan), kunt u ze bijvoorbeeld gebruiken om één uitgang van een CD4017 CMOS-teller te selecteren voor een synthesizer of alarmsysteem.

Er bestaan ook decimale en hexadecimale DIP-schakelaars (**figuur 5**). Deze komen van pas als u een binaire of BCD-code moet instellen (vaak een adres op een print). Dat gaat daarmee veel gemakkelijker dan wanneer u eerst in uw hoofd een getal naar BCD

of hex moet omrekenen. Ze zijn erg compact en hebben meestal een gemeenschappelijke aansluiting en 4 binaire of BCD-gecodeerde aansluitingen. Ze zijn verkrijgbaar voor liggende, staande of schuine montage.

In **figuur 6a** ziet u een enkelpolige aan/uit-schakelaar (single-pole single-throw, SPST). Ik twijfelde of ik die moest noemen, omdat hij vrij groot is en niet de standaard rastermaat heeft, maar deze schakelaars zijn beter dan jumpers (die u nooit kunt vinden als u ze nodig hebt!). Dan zijn er nog de typen van **figuur 6b**. Deze hebben verende draadverbindingen in plaats van 'echte' schakelaars, maar ze doen hetzelfde en zijn gemakkelijk in het gebruik. En ze kunnen waarschijnlijk flink meer stroom voeren dan de kleine schakelaars die gewoonlijk worden gebruikt.

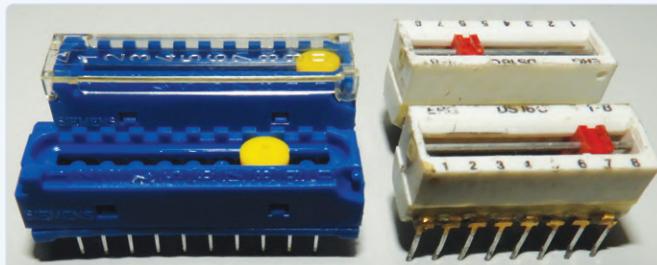
En dan hebben we nog het monster van **figuur 7**, 16-voudig, waarbij elke schakelaar 3 standen heeft (gemeenschappelijk naar contact 1, uit, gemeenschappelijk naar contact 2). U kunt deze gebruiken om een bit te definiëren als 1, 0 of *don't care*. Komt misschien van pas in een logic analyzer?

Dit type schakelaar wordt vaak in afstandsbedieningen gebruikt. Met drie standen voor elke schakelaar neemt het aantal mogelijke codes aanzienlijk toe – 3^8 (6561) in plaats van 2^8 (256) bij een 8-voudige schakelaar. ►

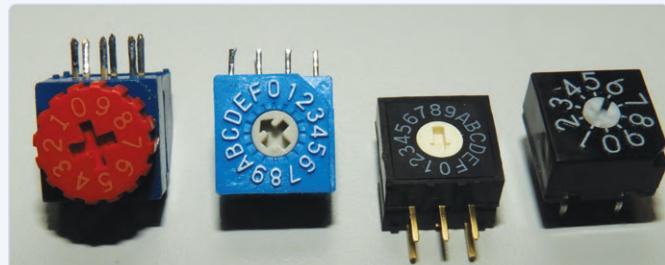
200716-03

Vragen of opmerkingen?

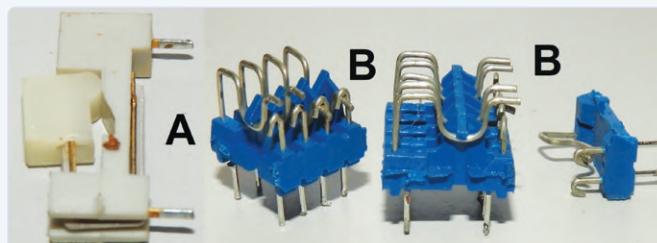
Hebt u vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur of naar de redactie van Elektor, via redactie@elektor.com.



Figuur 4. DIP-schakelaars van het type 'één-van-'.



Figuur 5. Hex- en BCD-schakelaars.



Figuur 6. Enkvoudige schakelaar en exemplaren met robuuste, verende draadverbindingen.



Figuur 7. Een driestanden 0/1/don't care DIP-schakelaar.

Interactief

correcties & updates || brieven van lezers

Ralf Schmiedel (Elektor) en Jens Nickel (Elektor)



Simpele functiegenerator

Elektor juli/augustus 2020, p. 20 (160548)

Er zit een foutje in het schema: instelpot P3 moet worden aangesloten tussen uitgang pin 6 en ingang pin 2 van IC4, en niet tussen pinnen 2 en 7. IC4 fungeert als buffer voor het open-collector uitgangssignaal van de LM311; P3 regelt de amplitude van het uitgangssignaal.



Elektor Uno R4

Elektor juli/augustus 2016, p. 58 (150790)

De ATmega328PB wordt nu volledig ondersteund door de standaard AVR-toolchain van de Arduino IDE. Dit maakt installatie van de Elektor Uno R4 veel sneller omdat er geen speciale toolchain meer gedownload hoeft te worden. Het maakt het board ook toekomstbestendig omdat nieuwe toolchain-functies automatisch worden gebruikt. Om de Elektor Uno R4 te installeren, plakt u deze link https://github.com/ElektronLabs/Arduino/releases/download/v1.0.1/package_elektor_arduino_r4_1_8_x_index.json in het vak *Additional Boards Manager URLs* van het *Preferences*-venster van de IDE. Filter in de Boards Manager (Arduino IDE-menu Tools -> Boards -> Boards Manager) op 'R4' en zorg ervoor dat de versie is ingesteld op '2.0.0'. We hebben dit getest met Arduino IDE versie 1.8.13.



Analoge elektronica ontwerpen – passieve filters

Elektor januari/februari 2021, p. 74 (200522)

Met veel belangstelling heb ik het informatieve artikel van Alfred Rosenkranzer gelezen. Het viel me op dat er een bepaalde filterconfiguratie is die niet in het artikel werd behandeld. Dit filter heeft uitstekende eigenschappen, vooral met betrekking tot de brede selectiviteit die haalbaar is.

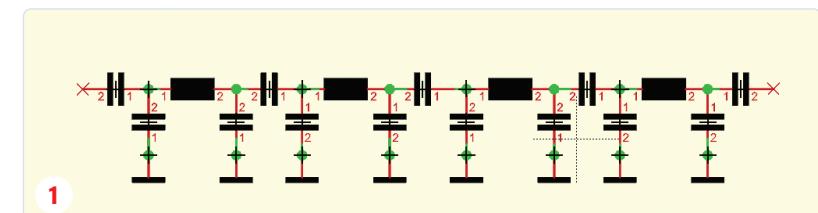
Ik werk aan het ontwerp van een SDR voor ons *Charly 25*-project, dat een STEMlab 16 Red Pitaya-unit gebruikt als SDR-kern in een TRX-ontwerp. We gebruiken min of meer acceptabele banddoorlaatfilters als preselector voor het front-end. Standaard banddoorlaatfilters bleken relatief slecht te zijn met betrekking tot de *out of band*-onderdrukking.

We wilden > 80 dB tot 100 MHz bereiken; dat is nodig omdat de A/D-omzetter veel ruis produceert en een gevoelige ontvanger doorgaans een omschakelbare breedband-voorversterker nodig heeft die in ons geval een versterking van +36 dB biedt.

In onze zoektocht naar een betere filterstructuur kwamen we de 'tubular filter'-configuratie tegen die vooral voor microgolftoepassingen wordt gebruikt. Dit type filter kan worden gezien als een reeks capacitief gekoppelde pi-netwerken (1).

Helaas laten de gebruikelijke gratis simulatieprogramma's dit filtertype grotendeels links liggen – de dure commerciële programma's voor het simuleren van microgolfschakelingen waren natuurlijk niet voor ons beschikbaar. De eerste filterstructuren werden daarom gecreëerd door middel van simulatie en stapsgewijze aanpassing met behulp van de Elsie filtersimulatie-software. Een van onze teamleden kreeg later medelijden met ons en schreef een klein programma om dit type filter te helpen berekenen:

<https://charly25-sdr.github.io/hardware/rx-filters-v2/calculator>



Bovendien heb ik uiteindelijk een gratis LC Filter Design Tool kunnen vinden dat ook dit type filter ondersteunt (zie *Tubular Filter*):

<https://rf-tools.com/lc-filter/>

Het resultaat van al onze inspanningen is een preselector-board voor gebruik met de 11 amateurbanden van 160 tot 6 m (plus ongefilterde doorvoer voor de ontvangst van radio-uitzendingen) dat onze eisen qua selectiviteit ruimschoots overtreft (2).

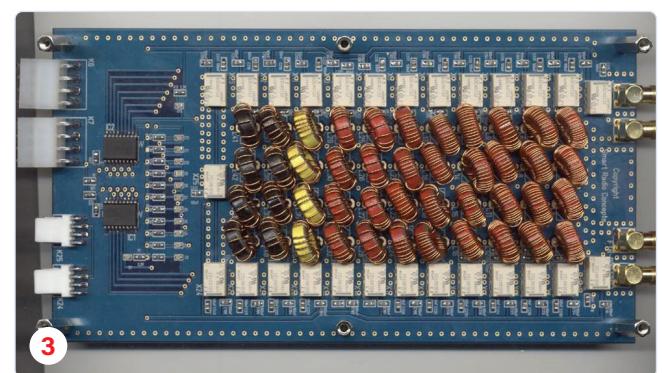
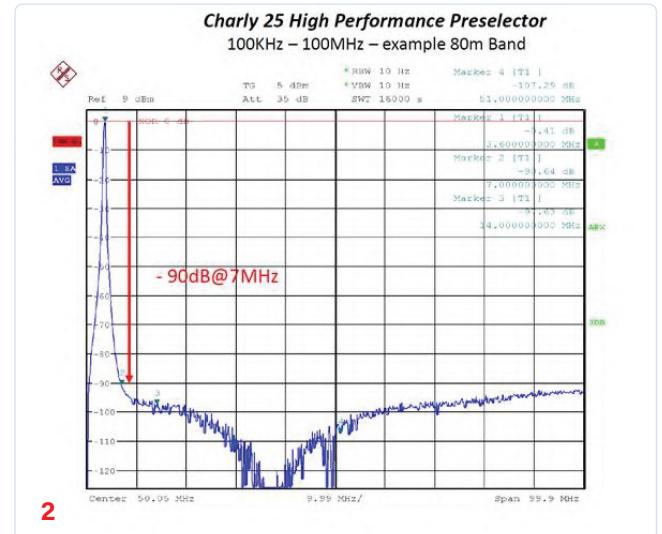
Het is de moeite waard erop te wijzen dat de prestaties van dit board werden bereikt door een zorgvuldige plaatsing van de componenten op een 4-laags print; de configuratie van het massavlak is uiterst kritisch. Een eenvoudiger layout met een 2-laags PCB kan een out of band-onderdrukking van ongeveer 70 dB bereiken. De complete preselectorprint is te zien in 3. Hij wordt bestuurd via een I²C-bus, waardoor hij compatibel is met de meeste controllers die worden gebruikt voor ontvangerprojecten. Ga voor meer informatie over het board naar:

www.smartradioconcepts.com

Wat de spoelen betreft, was het duidelijk dat poederijzer-kernen het beste geschikt zijn voor deze toepassing en een lage intermodulatievervorming garanderen.

De filterprint gebruikt geen PIN-diodes voor HF-omschakeling; ook daar is het gevaar van signaalintermodulatie-effecten te groot; vandaar het grote aantal relais.

Edwin Richter (DC9OE)



'Stekend' slechte satellietontvangst

Ik geniet momenteel van enige (welverdiende) ontspannende *downtime* in de warmte van de Braziliaanse staat Bahia. Deze geniet enige roem als de geboorteplaats van de samba, maar het leven is er niet zo eenvoudig, en soms moet ik mijn denkhoed opzetten als er zich technische problemen voordoen. Onlangs merkten we iets vreemds bij de ontvangst van onze (nog steeds analoge) satelliet-TV. Elke avond rond zonsondergang kwam er steeds meer ruis op het signaal en konden we sommige kanalen niet meer ontvangen. 's Morgens was alles dan weer normaal.

Nu weet ik van terrestrische uitzendingen dat bij zonsondergang atmosferische storingen kunnen optreden, maar bij golflengtes in de orde van grootte van een decimeter en met een min of meer vrij zicht leek me dat onwaarschijnlijk. Dus nam ik contact op met de plaatselijke satelliet-installateur die onze schotel een paar jaar geleden had geïnstalleerd.

Veel tijd had hij niet nodig: hij klom op het dak en vervang de gecorrodeerde LNB. De plastic dop die op de ingang van de LNB wordt geklikt, was broos geworden en gedeeltelijk wegebrokkeld (waarschijnlijk ten gevolge van UV- en IR-zonnestraling). Hij vond ook sporen van ongedierte... het lijkt erop enkele wespen (die hier groter zijn dan de Europese variëteiten) een comfortabel nest in de LNB hadden ingericht. De installateur had dit eerder gezien en zei dat, omdat het de laatste nachten behoorlijk koel waren (20 °C) en de LNB's overdag opwarmen, die nare beestjes 's ochtends wegvliegen en bij zonsondergang terugkomen naar waar ze zich lekker voelen – en dan de signaalweg blokkeren. Probleem opgelost – voor omgerekend ongeveer € 15!

Wolfgang Meyer

Vragen of opmerkingen?

Hebt u vragen of opmerkingen naar aanleiding van dit artikel?

Stuur een e-mail naar de redactie van Elektor, via redactie@elektor.com.

200715-03

Wiens product is het eigenlijk?

recht-op-reparatie beweging laat van zich horen

Tessel Renzenbrink (Nederland)

De recht-op-reparatie beweging staat voor het recht van mensen om hun eigen producten te repareren en te veranderen. Het repareren van dingen (in plaats van ze te vervangen) heeft veel voordelen. Het bespaart energie, grondstoffen en geld en vermindert de hoeveelheid afval. Bovendien zijn er weinig klussen die meer voldoening geven dan kapotte dingen in hun oorspronkelijke staat terug te brengen. Overal in Europa werken wetgevende instanties aan regelgeving om de repareerbaarheid te verbeteren. De krachtigste daarvan is een door het Europees Parlement aangenomen resolutie voor een alomvattende aanpak om het recht op reparatie te waarborgen. Daarin wordt een lange lijst van maatregelen voorgesteld, van het verlengen van garanties tot de verplichting reserve-onderdelen beschikbaar te stellen.

Geplande veroudering

Volgens een onderzoek uit 2018 van de Europese Commissie [1] zouden de meeste EU-burgers producten liever repareren dan ze vervangen. Op basis van gedragsexperimenten ontdekten onderzoekers dat 62% tot 83% van de mensen de reparatie-optie verkoos boven nieuwkoop (afhankelijk van het producttype). Burgers zijn best bereid om de wegwerpeconomie in te ruilen voor duurzamere en herstelbare producten. Maar diverse obstakels maken het moeilijk om dat doel te realiseren. Misschien wel de meest schaamteloze hiervan is geplande veroudering (*planned obsolescence*): de praktijk waarbij een ontwerp bewust van zwakke plekken wordt voorzien zodat het op een vooraf bepaald tijdstip defect zal raken. Bij digitale apparaten kan deze praktijk de vorm aannemen van het beëindigen van software- of beveiligings-updates na slechts één of twee jaar. Het apparaat wordt onbruikbaar, ook al is de hardware nog volledig functioneel.

Een ander obstakel is het onmogelijk maken van reparatie – bijvoorbeeld door een product dusdanig te ontwerpen dat het bijna onmogelijk is om onderdelen te vervangen. De beroemde *iFixit Smartphone Repairability Scores* somt kwalijke voorbeelden op die variëren van vastgelijmd covers tot nauwelijks bereikbare batterijen [2]. Ook reserve-onderdelen die niet verkrijgbaar of onbetaalbaar zijn vallen onder deze categorie. Een andere strategie is om de handleiding tot intellectueel eigendom te verklaren waardoor consumenten en onafhankelijke reparatiewerkplaatsen deze niet kunnen krijgen.

Repareerbaarheidsindex

Gebrek aan informatie is nog een andere hindernis die het voor mensen moeilijker maakt om voor repareerbare producten te kiezen. Uit de bovengenoemde Europese studie bleek dat, wanneer informatie over repareerbaarheid wordt verstrekt op het moment van aankoop, consumenten "meer dan twee keer zo sterk geneigd zijn om producten te kiezen met de hoogste repareerbaarheidsclassificaties". Het Franse *Ministère de la Transition écologique* doet hier wat aan met de introductie van een repareerbaarheidsindex [3]. Vergelijkbaar met de iFixit Score worden producten gelabeld met een repareerbaarheidsindex



Illustratie: Jose Conejo Saenz, via Pixabay.com. Pixabay-licentie.

(*Repairability Index*) die van 1 tot 10 loopt. Deze index is gebaseerd op vijf criteria, waaronder gemakkelijke demontage en beschikbaarheid van reserveonderdelen. Etikettering is sinds januari van dit jaar verplicht voor TV's, wasmachines, laptops, telefoons en grasmaaiers. Na verloop van tijd zullen er meer productcategorieën worden toegevoegd. Het doel van de minister is dat binnen vijf jaar 60% van de elektronische producten repareerbaar is.

Recht op reparatie voor Europese burgers

De Repairability Index kan te zijner tijd in alle EU-landen worden ingevoerd. In november 2020 stemde het Europees Parlement voor een resolutie waarin wordt opgeroepen tot een duurzamere interne markt [4]. Deze resolutie roept expliciet op tot een recht op reparatie voor Europese burgers en stelt een lange lijst van maatregelen voor. Deze omvatten het garanderen van gratis beschikbaarheid van handleidingen, het aanmoedigen van standaardisering van reserve-onderdelen ten behoeve van interoperabiliteit, het vaststellen van een

verplichte minimumperiode voor de levering van reserveonderdelen en het leveren van die onderdelen tegen een redelijke prijs. Om de informatie-asymmetrie aan te pakken, moeten leveranciers op het moment van aankoop informatie verstrekken over de repareerbaarheid en duurzaamheid van producten. Dit omvat zowel de beschikbaarheid van reserveonderdelen als software-updates. De Repairability Index wordt voorgesteld als een middel om dat doel te bereiken.

Om geplande veroudering een halt toe te roepen, stelt het Parlement een brede strategie voor. Het opzettelijk toevoegen van kwetsbaarheden aan een ontwerp moet worden verboden, en software-updates moeten gedurende de geschatte levensduur van een product worden verstrekt. Om het vertrouwen van de consument in tweedehands en gerepareerde goederen te vergroten, moeten garanties overdraagbaar worden. In plaats van een garantie aan een persoon te koppelen, moet deze aan het product zelf worden gekoppeld. Om een gelijk speelveld te creëren voor Europese bedrijven die aan deze strenge eisen moeten



Illustratie: Wilfried Pohnke, via Pixabay.com. Pixabay-licentie.

Burgers zijn best bereid om de wegwerpeconomie in te ruilen voor duurzamere en herstelbare producten.

voldoen, dringt het Parlement er bij de Commissie en de lidstaten op aan meer inspanningen te leveren om geïmporteerde niet-conforme goederen uit de schappen te verbannen.

Transformatie van de wegwerpeconomie

Het Parlement is niet het enige EU-orgaan dat oproept tot een recht op reparatie. In maart 2020 keurde de Europese Commissie het *Actieplan Circulaire Economie* goed, waarin ook expliciet het recht op reparatie wordt genoemd [5]. Zowel het Parlement als de Commissie roepen op om reparatierechten deel uit te laten maken van het Europese *Green Deal*-initiatief, het actieplan om Europa in 2050 klimaatneutraal te maken. De ambitie is om de huidige *take-make-use-discard* economie om te vormen tot een duurzame, circulaire economie. Het faciliteren van reparaties om de levenscyclus van producten te verlengen, past in die ambitie. Een duurzame economie is niet alleen ecologisch evenwichtiger, maar ook veerkrachtiger. In zijn resolutie wijst het Parlement erop dat de Covid-19-crisis duidelijk heeft gemaakt dat het huidige economische systeem, met zijn wereldwijde toeleverings-

ketens, kwetsbaar is. Het vraagt daarom om nieuwe bedrijfsmodellen en ondersteuning voor kleine en middelgrote Europese bedrijven. Een focus op lokaal repareren in plaats van nieuwe goederen vanuit het buitenland te importeren, zou bijdragen aan de ontwikkeling van kennis, vaardigheden en lokale economieën.

De initiatieven moeten nog het langzame wetgevingsproces van de EU doorlopen voordat ze als wet worden aangenomen. Gedurende dat proces kunnen ze kunnen nog worden afgezwakt. Maar het feit dat het recht op reparatie is verankerd in de Green Deal van de Europese Unie, een van de topprioriteiten op de EU-agenda, is veelbelovend. ■

200713-03

Vragen of opmerkingen?

Hebt u vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur of naar de redactie van Elektor, via redactie@elektor.com.

WEBLINKS

- [1] Dr. Annette Cerulli-Harms et al., 'Behavioral Study on Consumers' Engagement in the Circular Economy', pagina 11, oktober 2018: <https://bit.ly/39PvWkA>
- [2] Smartphone Repairability Scores, Ifixit: <http://bit.ly/2M8yZfj>
- [3] Ministère de la Transition écologique, 'The anti-waste law for a circular economy: key measures', januari 2021: <http://bit.ly/3sHioAi>
- [4] Europees Parlement, 'European Parliament resolution of 25 November 2020 Towards a more sustainable single market for business and consumers': <http://bit.ly/2NeJcaL>
- [5] Europese Commissie, 'Changing how we produce and consume: New Circular Economy Action Plan shows the way to a climate-neutral, competitive economy of empowered consumers', maart 2020: <http://bit.ly/395SdeE>

Uit het leven gegrepen

het grote blunderboek

Ilse Joostens (België)

Het leven van een elektronicus gaat niet altijd over rozen, ongeacht of u nou een beginnende hobbyist bent of een door de wol geverfde oude rot in het vak. Of u het leuk vindt of niet, de technologie evolueert in een sneltreintempo, waardoor een leven lang leren het credo is als u met de nieuwste ontwikkelingen wilt blijven. Leren is een proces van vallen en opstaan en de allerbelangrijkste dingen leert u gewoonlijk op de 'harde' manier, waarna gevoelens van teleurstelling, mislukking en vooral diepe schaamte onvermijdelijk uw deel zijn.

Tenzij u tot de 'pampersgeneratie' – soms ook 'pretparkgeneratie' genoemd – behoort, wordt u daar uiteindelijk alleen maar beter van, want van uw fouten kunt u het meeste leren. En als het écht te gortig wordt is het natuurlijk ook nog een optie om uw beste management-talent te voorschijn te halen en te proberen een ander de schuld te geven.

Vallende messen en pyrotechniek

Toen ik nog jong was, ergens in de jaren tachtig van de vorige eeuw, was er, in vergelijking met heden ten dage, relatief weinig informatie rond elektronica beschikbaar. Computers waren geen gemeengoed en ook het internet moest nog geboren worden. Ik moest het dus vooral doen met een oud boek over elektriciteitsleer en wat obscure werkjes die ik links en rechts had gevonden. De hoeveelheid boeken over elektronica in de plaatselijke bieb was ook tamelijk beperkt en databoeken waren destijds alleen beschikbaar voor bedrijven en hun streng kijkende grijsbearde elektronici in stofjas.

Op een gegeven ogenblik was ik vrolijk aan het experimenteren met wat diodes, lampjes en een treintrafo. Door de polariteit te wisselen of door wisselspanning te gebruiken kon ik via twee draadjes twee groepen lampjes onafhankelijk van elkaar sturen, tot ik (natuurlijk) het onzalige idee kreeg om dat ook met netspanning te proberen. Ik had als eerste test een bruggelijkrichtertje gebouwd met daarachter een gloeilampje. Ik herinner mij nog goed dat ik het lampje een fractie van een seconde heb zien oplichten alvorens de bruggelijkrichter met een luide knal en felle flits als een vuurwerkbom met een vonkenregen in mijn gezicht ontplofte. Wist ik veel dat diodes een bepaalde maximale spersspanning hebben en dat ik die in dit geval grof te boven was gegaan. De hoofdschakelaar lag eruit om nog te zwijgen over de paniek in de ogen van mijn huisgenoten. Een jaar of zo later kocht ik mijn eerste soldeerbout, zo'n bakelieten kachelpook uit de plaatselijke bouwmarkt. Mijn eerste soldeerverbindingen zagen er eerder 'klonterig' uit maar ik was blij dat ik niet langer hoefde te improviseren om verbindingen te maken. In die tijd waren platenspelers met ingebouwde versterker nog populair en op een gegeven moment had ik mij geëngageerd om de platenspeler van een buurmeeuw te herstellen. Er was intern ergens een draadje losgeraakt. Ik had niet veel plaats en toen ik bijna klaar

was gleed de soldeerbout van de tafel waarna ik die in een reflex heb geprobeerd te vangen, natuurlijk net aan de hete kant... Naar analogie met de bekende beurswijsheid "Probeer nooit een vallend mes te vangen" heb ik toen tot mijn schade en schande ervaren dat dit ook geldt voor hete dingen zoals soldeerbouten. Het gevolg was de geur van verschroeid vlees en een forse brandwond aan mijn rechterhand. Ik heb, de pijn verbijtend, de klus afgewerkt waarbij ik alles wat ook maar enigszins koel aanvoelde vastpakte om de pijn wat te verzachten, waaronder een setje alkaline D batterijen dat toevallig rondslingerde en die ik om beurten in mijn hand hield. Enige weken later waren er opnieuw problemen met de platenspeler en had iemand anders er naar gekeken, waarna ik de vraag kreeg waarom ik nou in godesnaam een batterij in de platenspeler had gestopt...

Wij hadden nog zo'n ouderwetse boormachine liggen zonder toerenregeling en de 'Boormachineregelaar' uit het boek "302 Schakelingen" van (toen nog) Elektuur leek me wel wat. Ik was toen ook al bezig met zelf printjes etsen met zo'n stift met etsbestendige inkt en het geheel was vrij snel opgebouwd én met succes getest. Helaas gebruikte ik toen bij gebrek aan financiële middelen nogal eens 'gerecyclede' componenten van eerder dubieuze herkomst. Toen ik het ding naderhand in gebruik nam begaf een van de condensatoren het waarbij heel de werkruimte op een mum van tijd gevuld werd met een vreselijk ruikende walm. Nadat de mist opgetrokken was en ik de stoute condensator met rookbom-aspiraties onschadelijk had gemaakt heeft het ding uiteindelijk nog jaren trouwe dienst bewezen.

Bedrieglijke eenvoud

Ook professioneel heb ik er een uitgebreid brokkenparcours op zitten, maar de ruimte voor dit artikel is helaas te beperkt om alles aan bod te laten komen. Ik zal dus niet te veel uitweiden over die



keer dat ik het ene telefoontje na het andere in het Frans kreeg van klanten en dat ik sufgekletst het netsnoer van een apparaat doorgeknipt had zonder de stekker eerst uit het stopcontact te halen, waarna een groot deel van het bedrijfsgebouw in het donker zat. Of die keer dat ik met een kolomboormachine in mijn hand heb geboord... Of die keer dat ik – alweer sufgekletst – in contact kwam met fase én nulleider van de netspanning waarna ik door de hevige elektrische schok, maar vooral ook door de schrikreactie, met bureaustoel en al, twee meter verder onzacht tegen een archiefkast landde.

Ook nu nog zijn het de op het eerste zicht eenvoudige dingen die me uiteindelijk de das omdoen. Tijdens het printontwerp een IC met meer dan honderd pootjes op een print pleuren is een fluitje van een cent, maar de 5 V en massa van bijvoorbeeld een USB-connector de eerste keer juist aansluiten, ho maar! Tja, de polariteit der dingen, dat zal nu eenmaal voor eeuwig en altijd mijn nemesis blijven...

Vooraleer ik afsluit wil ik u oproepen om uw persoonlijke blunders en gênantste momenten met de Elektor Community te delen via uw commentaar op dit artikel. Het is niets om u voor te schamen... Nou ja, misschien een klein beetje toch, maar leedvermaak is ook vermaak... 

200682-02

Vragen of opmerkingen?

Hebt u vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur of naar de redactie van Elektor, via redactie@elektor.com.

Een bijdrage van

Tekst en keuze van illustraties:
Ilse Joostens (België)

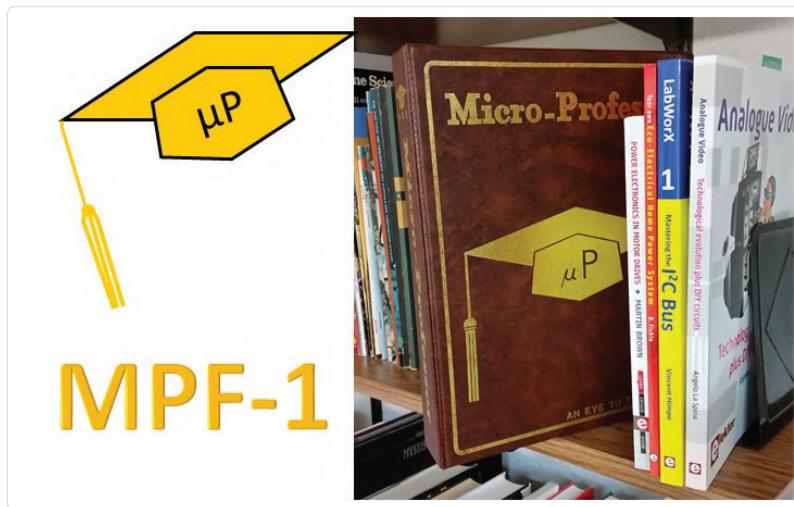
Redactie: **Eric Bogers**
Layout: **Giel Dols**

WEBLINKS

- [1] **De pampergeneratie:** <https://bit.ly/2LWJQZW>
- [2] **ElectroBOOM Funny Compilation:** www.youtube.com/watch?v=gAnBc4iFCvk
- [3] **ElectroBOOM YouTube Channel:** www.youtube.com/c/Electroboom/featured

Micro-Professor

assembleertaal leren op een Z80



aan het licht: in plaats van de verwachte onverteerbare microprocessor-documentatie, zag je de microprocessor zelf, gemonteerd op een board en klaar voor gebruik. Geïnteresseerd? Ga met me mee op een ontdekkingsreis naar geavanceerde training anno 1981, met de Multitech Micro-Professor MPF-1.

In een industriesector die zo snel evolueert als de elektronica, is permanente opleiding de enige manier om de aansluiting niet te verliezen. Stan Shih, zijn echtgenote Carolyn Yeh en vijf andere zakenpartners zagen dit en richtten in 1976 het bedrijf Multitech op in Taipei (Taiwan) – nu beter bekend onder de naam 'Acer'. Naast de distributie van reserve-onderdelen voor elektronische apparaten en verstrekken van advies met betrekking tot microprocessoren, vervaardigde Multitech ook microprocessor-leersystemen in hun eigen fabrieken. Ze hadden daarom al ervaring met deze systemen toen de Micro-Professor MPF-1 (**figuur 1**) in 1981 werd gelanceerd.

De hardware

De MPF-1 was een single-board computer met een zescijferig zeven-segment-display en een toetsenbord met 36 toetsen. Hij was opgebouwd rond een Zilog Z80-microprocessor die geklokt werd op 1,79 MHz. **Figuur 2** toont de layout van het board. De Z80-microprocessor werd vergezeld van een Z80 PIO-IC, een CTC-IC, een 8255, 2 KB RAM en 4 KB EPROM. De

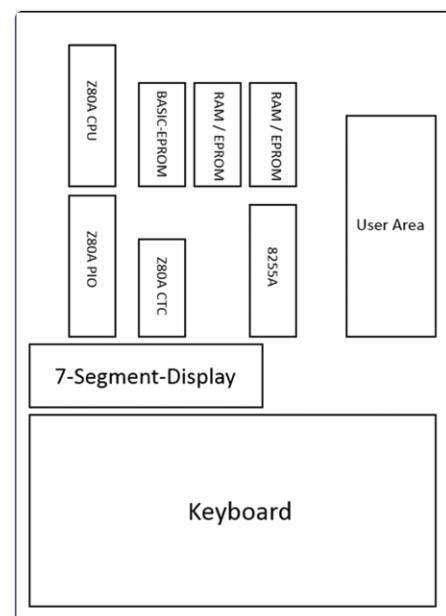
8255, een programmeerbare parallele I/O-component, verbond het toetsenbord en het zeven-segment-display met de processor. Hij vormde ook een interface naar de ingebouwde luidspreker en

de connector voor de cassettereader, die werd gebruikt om gegevens en programma's op audiocassette op te slaan – toen de goedkoopste optie voor gegevensopslag. De Z80-CTC is een counter/timer met vier



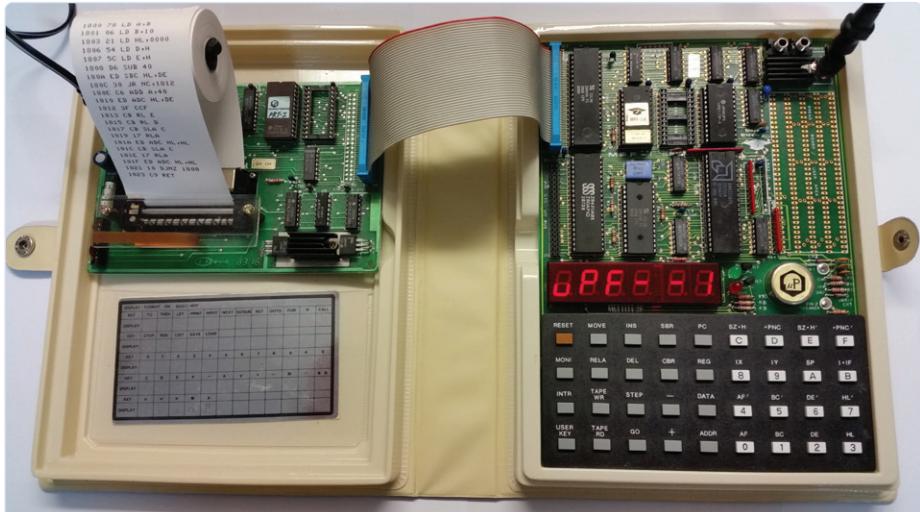
Figuur 1. De Micro-Professor MPF-1B als geopend boek, samen met de bijbehorende documentatie.

Figuur 2. De layout van het Micro-Professor board.



Karl-Ludwig Butte (Duitsland)

In 1981 waren er veel boeken over de Zilog Z80-microprocessor, die in 1976 was geïntroduceerd, maar dit specifieke boek was anders. Met zijn rugdikte van 4 cm zou je ongeveer 500 pagina's aan samengebalde informatie verwachten, uitmonddend in eindeloze uren saaie studie van moeilijke en abstracte leerstof. Maar openslaan van het boek bracht een geweldige verrassing



Figuur 3. De MPF-1B met aangesloten thermoprinter.

onafhankelijk programmeerbare counter/timers. Ten slotte is de Z80-PIO een parallele I/O-component die wordt gebruikt om periferie aan te sluiten. Multitech bood onder meer een thermoprinter (**figuur 3**) en een EPROM-programmer als randapparatuur aan, evenals een board voor spraakuitvoer. Er waren ook andere bedrijven die extensies maakten of verkochten voor de Micro-Professor. Zo heeft het instituut voor afstandsonderwijs Christiani in Konstanz (Duitsland) zijn Micro-Professor curriculum uitgebreid met een eigen I/O-kaart die speciaal is ontworpen voor het afstandsonderwijs.

Documentatie

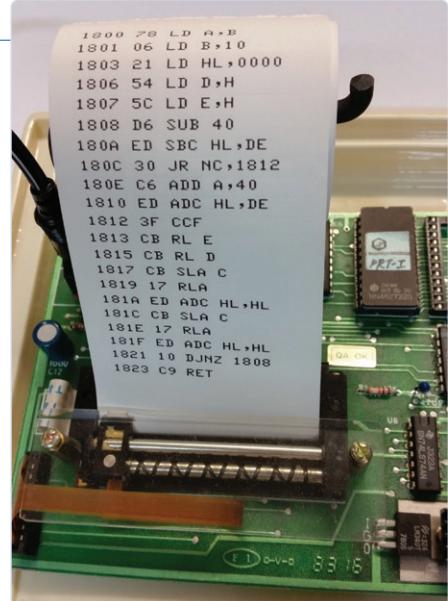
De documentatie voor de Micro-Professor (**figuur 1**) was net zo minimaal als de hardware-configuratie en zeker niet vergelijkbaar met de liefdevol samengestelde gebruikershandleiding voor de CP1-leercomputer van Kosmos uit Stuttgart, beschreven in Elektor juli/augustus 2018 [1]. Het is geen toeval dat een gerenommeerd instituut voor afstandsonderwijs als Christiani de Micro-Professor adopteerde en een eigen educatieve cursus publiceerde. Multitech leverde een gebruikershandleiding van slechts 95 pagina's bij de MPF-1. De handleiding bij mijn Micro-Professor was tenminste in het Duits. De *BASIC MPF Operation Manual* was slechts 39 pagina's dik, en net als de *MPF-1 Experiment Manual* die ook bij de Micro-Professor werd geleverd, was deze alleen beschikbaar in de originele Engelse versie.

De auteur(s) gingen ervan uit dat de lezer

over een goede basiskennis van microprocessor-technologie beschikte. De gebruikershandleiding van de MPF-1 begint bijvoorbeeld met de volgende zinnen (na een opsomming van de technische gegevens en een korte beschrijving van de verschillende keys): “*Het monitorprogramma bevat alle serviceroutines die nodig zijn om het de gebruiker gemakkelijk te maken: (1) programma's in RAM laden, gevolgd door testen en/of aanpassen.*” Hoewel alle essentiële informatie, zoals de Z80-instructieset, de programmeerarchitectuur van de Z80, de geheugenstructuur en de I/O-pintoewijzingen verderop in de handleiding beschikbaar was, was deze meestal in tabelvorm zonder enige nadere uitleg. Het was daarom raadzaam om voor geschikt referentiemateriaal te zorgen, zoals *How to Program the Z80* van Rodnay Zaks. Hij had veel meer te vertellen over de Z80 – zijn boek was meer dan 600 pagina's dik.

Programmeren in assembleertaal

Om uit eerste hand te leren hoe het is om met de Micro-Professor te werken, heb ik het voorbeeldprogramma ‘Square Root’ uit de handleiding gebruikt. Gelukkig was het niet alleen in de verkorte steno-notatie van de Z80-assembleertaal afgedrukt, maar ook in hex, omdat alleen hexadecimale code met het toetsenbord kan worden ingevoerd. Eerst moet u de ADDR-toets indrukken en het startadres invoeren, daarna kunt u met de DATA-toets omschakelen naar gegevensinvoer en het eerste databyte



Figuur 4. De assembleertaal-listing van het vierkantswortel-programma.

invoeren. Dan drukt u op de ‘+’-toets om naar het volgende byte te gaan; dan kunt u meteen het volgende byte van het programma invoeren. Op deze manier kan het programma vrij snel worden ingetoetst. Om het te controleren, heb ik het uitgeprint met behulp van de Disassembler Listing Utility, die wordt geleverd door de aangesloten thermoprinter, beginnend op hex-adres 6000 (**figuur 4**).

Nadat ik had gecontroleerd of de printout overeenkwam met de listing in de handleiding, probeerde ik het programma uit. De Micro-Professor moest de vierkantswortel van 81 berekenen. Het programma verwachtte deze waarde (in hexadecimaal formaat natuurlijk) aan te treffen in processorregister BC. Met behulp van de REG- en BC-toetsen kon ik de huidige waarde van dit register zien, en met de DATA-toets kon ik het eerste byte invoeren (BC is een 16bit-register). Op dit punt stond ik voor de vraag welk byte als eerste moest komen: 51h (equivalent met 81 decimaal) of ooh. Conform het principe dat praktische ervaring meer waard is dan boekenwijsheid, probeerde ik eerst ooh gevolgd door 51h, wat een volkomen zinloos resultaat opleverde. Dat betekende dat ik opnieuw moest beginnen en achtereenvolgens REG, BC, DATA, 5, 1, +, 0, o moet intoetsen (**figuur 5a**). Om de proefrun te starten, moest ik als adres vervolgens 1800h invoeren en op de GO-toets drukken (**figuur 5b**). En toen werd het spannend: ik controleerde het resultaat in register DE, en ziedaar: op het display lichtte de waarde ‘9’ op (**figuur 5c**).

VRAAGGESPREK MET MAX D. SOFFE, MANAGING DIRECTOR VAN FLITE ELECTRONICS INTERNATIONAL LIMITED

Max D. Soffe is lid van de Industrial Advisory Board van de afdeling Electronic Engineering aan Royal Holloway, University of London, en van het Institute of Engineering Technology (www.theiet.org). Van 2000 tot 2004 was hij voorzitter van de Educational Technology Engineering Manufacturers Association (ETEMA) in Groot-Brittannië. Hij was ook de directeur van de British Educational Suppliers Association (BESA).

Karl-Ludwig Butte: Flite Electronics International kocht de intellectuele eigendomsrechten van de Micro-Professor MPF-1B van Acer. Waarom eigenlijk?

Max D. Soffe: Acer heette vroeger Multitech. Ik ontmoette ze tijdens mijn eerste reis naar Japan in 1981. Ze presenteerden de MPF-1B Z80 Microprocessor Trainer in Tokio. Ik had het geluk om de oprichters van Multitech te ontmoeten en een vriendschappelijke relatie te ontwikkelen. Het bedrijf had toen slecht 20 medewerkers; het was gevestigd in Hsinchu Science Park, Taipei. Ik maakte een advertentie van twee pagina's in een tijdschrift met de naam Wireless World in het VK en verkocht in drie weken tijd honderd stuks MPF-1B. Daarom gaf Multitech mij de distributierechten in het VK, hoewel we wereldwijd begonnen te verkopen.

Later moest Acer zijn omzet vergroten en wilde zich concentreren op de sterk groeiende PC-markt, en had daar zoals bekend veel succes. De kleine Z80-trainers waren eigenlijk verliesgevend. In 11 jaar had Flite Electronics duizenden stuks MPF-1B verkocht. Omdat het product nog steeds duur was voor een particuliere gebruiker, waren onze afnemers voornamelijk tertiair onderwijs, universiteiten en hogescholen. We verkochten nog steeds veel en

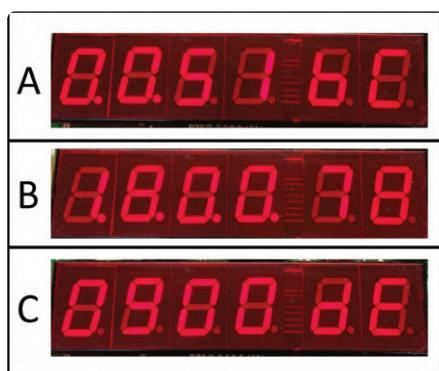
dus onderhandelde ik met Acer over de koop van de intellectuele eigendomsrechten. In februari 1993 was het dan zover.

Het was niet zo succesvol als ik had gewild, omdat Acer zojuist zijn handen van het product had afgetrokken en het aan ons overliet om de 'piraterij' van het product te stoppen. We konden het ons niet veroorloven om de strijd aan te gaan met drie of vier bedrijven, die brutaalweg de MPF-1B met handleiding, print en firmware kopieerden. We hebben echter doorgedreven en hebben de PCB ietwat aangepast, waarbij we alle firmware hebben behouden. De verkrijgbaarheid van de onderdelen begon een probleem te worden en er waren problemen met de (aangegeven) snelheid van de Z80-processor, EEPROM en RAM.

Karl-Ludwig Butte: Er zijn websites zoals https://en.wikipedia.org/wiki/Micro-Professor_MPF-I waar te lezen is dat de Micro-Professor nog steeds wordt vervaardigd en verkocht door Flite Electronics International. Is dat correct? En zo ja, waar kunnen lezers een exemplaar bestellen en tegen welke prijs?

Max D. Soffe: Ja, natuurlijk maken we nog steeds de MPF-1B, zij het in heel kleine series. De laatste batch die we deden was ongeveer zes maanden geleden: 25 stuks voor een universiteit in het Midden-Oosten. De docenten hadden bij hun opleiding in het VK de MPF-1B 20 jaar geleden gebruikt en onderwijsden er nog steeds machinecode/hexadecimale codering mee op de universiteit.

Wanneer Elektor-lezers een e-mail sturen naar sales@flite.co.uk en een beetje geduld hebben, krijgen ze antwoord met een prijsopgave. We hebben er momenteel zo'n acht op voorraad. Houd



Figuur 5. A: Invoer van de waarde 81_{dec} in register BC. B: Startadres. C: Uitkomst 9 in register DE.



Figuur 6. De Micro-Professor in BASIC-modus met de toetsenbord-overlay.

De BASIC-interpreter

Kort na de introductie van de MPF-1 kwam de MPF-1B op de markt en daarna werd de MPF-1 herernoemd tot MPF-1A om hem te onderscheiden van de B-versie. Wat kon de MPF-1B dat zijn voorganger niet kon? Het antwoord is dat Multitech een nieuwe EPROM had toegevoegd met daarin een kleine BASIC-interpreter. Ik wilde die natuurlijk uitproberen, aangezien BASIC met een zeven-segment-display en een hex-toetsenbord totaal nieuw voor mij was. De MPF-1B werd voor dit doel geleverd met een toetsenbord-sjabloon, die in het geval van mijn apparaat inmiddels al lang is verdwenen. Tegenwoordig is het internet het antwoord op dit soort problemen. Ik vond het sjabloon al snel en printte het op vrij dik papier. Het uitsnijden van de gaatjes met een hobbymes duurde even, maar het fraaie resultaat is te zien in **figuur 6**.

er rekening mee dat wanneer deze verkocht zijn, het twee tot drie maanden kan duren voordat we een nieuwe batch opstarten. Ik heb gemerkt dat er een paar tweedehands exemplaren worden verkocht op eBay.

Zoals ik al heb gezegd, zijn de componenten nu vrij duur in relatie tot hun prestaties, en aan de datumcode op het IC kan men zien wanneer ze geproduceerd zijn (9243 bijvoorbeeld staat voor de 43ste week van 1992). We besteden relatief veel tijd aan het testen van de componenten en het eindproduct op snelheid en timingproblemen. Om deze reden monteren we alle IC's in voetjes, zodat we IC's snel kunnen omwisselen. Wanneer we een compleet nieuw exemplaar leveren, werkt deze natuurlijk perfect, met een garantie van 1 jaar.

We realiseren ons dat we een dergelijk product nooit een succesvol businessmodel zal worden. Het was de springplank voor Acer om een groot internationaal merk te worden, maar net als Acer zal de Micro-Professor ons nooit een grote winst opleveren. Helaas kunnen we het ons niet veroorloven hem voor minder dan € 250 excl. btw en verzendkosten te verkopen.

Karl-Ludwig Butte: Wat voor documentatie levert u bij de Micro-Professor?

Max D. Soffe: De documentatie is erg belangrijk. We voorzien elke Micro-Professor van een Student Workbook. Deze handleiding begeleidt de student vanaf het uitpakken en opstarten van het systeem tot het begrijpen van microprocessorsystemen, architectuur, hardware, software en interfacing. Het was aanvankelijk bedoeld als een zelfstudie-handleiding om studenten te trainen in hexadecimaal programmeren. Zoals bekend werken computers binair; de daarop volgende stap is hexadecimaal. Hexadecimale

codering was een uitkomst in de tijd dat geheugen klein en duur was. De Micro-Professor leerde de student om zeer compacte code te schrijven, en niet de slordige code zoals veel programmeurs tegenwoordig doen. Het een en weer schuiven van tekstballonnen is eigenlijk geen echt programmeren.

Karl-Ludwig Butte: Wat is momenteel de belangrijkste activiteit van Flite Electronics International?

Max D. Soffe: Als gevolg van de piraterij van de Micro-Professor in de jaren negentig werkten Flite Electronics samen met de firma K & H in Taiwan, die apparatuur voor onderwijs technologie vervaardigt. Dit kwam doordat onze klantenkring voornamelijk onderwijsinstellingen waren, geen particulieren. Wij zijn de distributeur van de educatieve technologieproducten van K & H in het VK. We hebben onlangs een laboratorium voor elektrische machines ter waarde van £ 240.000 geïnstalleerd aan de Coventry University. We hebben het systeem niet ontworpen, maar samengesteld uit bestaande K & H-modules. Vorige week hebben we voor £ 50.000 een uitbreiding aan dit laboratorium toegevoegd.

We leveren ook nog steeds algemene microprocessor-trainers voor de Motorola 68000, de Intel 8086 en de 8032; ook hier verkopen we van elk type zo'n 10 stuks per jaar, voornamelijk ter ondersteuning van bestaande klanten, maar we doen voor deze producten ook graag een prijsopgave. Een ander aanvullend product dat we in de jaren negentig hebben ontworpen als toevoeging voor al onze microprocessor-trainingssystemen, is een alles-in-één laboratorium dat de APB (applications board) wordt genoemd. Ook deze wordt geleverd met een training-manual.

Karl-Ludwig Butte: Hartelijk dank voor dit gesprek.

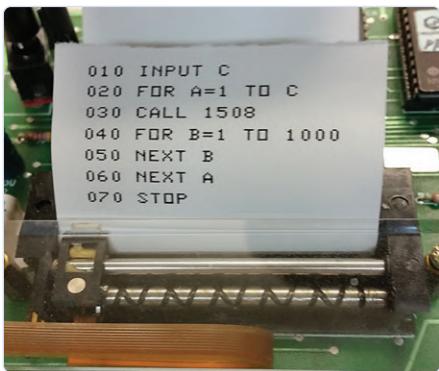
Als voorbeeldprogramma heb ik een routine uit de BASIC-MPF Operation Manual gekozen die meerdere signaaltoontjes kan genereren. De BASIC-interpreter bevindt zich in het geheugen vanaf het start-adres 0800h en wordt gestart met het GO-commando. Hij kondigt zijn aanwezigheid aan met 'bASIC' op het display, zoals in figuur 6. Nu kan het programma worden ingevoerd. Eerst het regelnummer, dan de BASIC-instructie, gevolgd door eventuele variabelen, getallen of parameters. Elke BASIC-instructie kan worden ingevoerd met een eigen toetsaanslag. Als laatste wordt op ENTER gedrukt, net als vroeger bij de Apple II, PET 2001 of TRS-80. Alles werkte tot op zekere hoogte prima; ik ondervond problemen met het invoeren van het statement **FOR A = 1 TO C**. In plaats van het isgelijkteken bleef er een 5 op het display verschijnen, hoewel ik toch

eerst op de SHIFT-toets had gedrukt. Dit probleem bleek te wijten aan een oude gewoonte: omdat het toetsenbord van de Micro-Professor zo sterk lijkt op het toetsenbord van een zakrekenmachine, drukte ik kort op de SHIFT-toets in plaats van deze ingedrukt te houden, zoals bij een PC. Het duurde even voordat ik me realiseerde dat de Micro-Professor zich als een soort PC beschouwt, dus ik moest de SHIFT-toets ingedrukt houden terwijl ik op de isgelijktoets drukte.

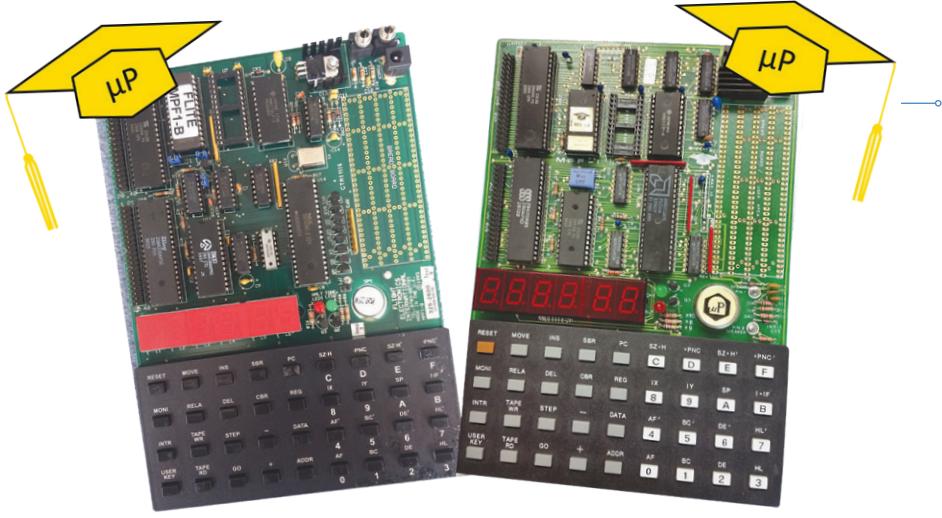
Na het oplossen van dit probleempje was het kinderspel om de rest van het programma in te voeren. **Figuur 7** toont een aantal voorbeelden van de manier waarop BASIC-instructies worden weergegeven op het zeven-segment-display. **Figuur 7a** toont **10 Input C** in normale notatie, terwijl in **figuur 7b** de regel **50 Next B** te zien is. De handleiding bevat



Figuur 7. A: 10 Input C. B: 50 Next B. C: Invoerprompt.



Figuur 8. Voorbeeld-printout van een BASIC-programma op de thermoprinter.



Figuur 9. De Micro-Professor van Flite Electronics International (links) en zijn Multitech-tegenhanger (rechts).

een tabel met de BASIC-instructies en de weergave daarvan op het display. Dat is ook echt nodig, want sommige gevallen vergen de nodige fantasie. Bewerken is ook mogelijk, maar in vergelijking de is vi-editor uit de Unix-wereld bijzonder gebruiksvriendelijk.

Nu kunnen we het BASIC-programma uitproberen dat we zojuist hebben ingevoerd. Nadat u het programma met de RUN-toets hebt gestart, wordt eerst gevraagd naar het aantal te genereren signaaltoontjes (**figuur 7c**). Ik voerde 3 in en drukte op ENTER, en toen hoorde ik drie relatief hoge toontjes, net als verwacht. Bij een relatief lang programma is de kans op tikfouten tamelijk groot. Naar mijn mening is het opsporen van tikfouten op een cryptisch zeven-segment-display – voorzichtig uitgedrukt – geen pretje. Gelukkig kunnen BASIC-programma's als normale tekst worden afgedrukt als er een printer op het systeem is aange-

sloten. **Figuur 8** toont de listing van het signaaltoon-programma.

Een geslaagde onderneming

Eén ding is zeker: als u eenmaal serieus met de Micro-Professor hebt gewerkt, weet u alles van de Z80-microprocessor en zijn periferie wat er te weten valt. Dat gaat is niet altijd gemakkelijk en doorzettingsvermogen is zeker wenselijk, maar naar mijn mening is het de moeite waard. Het is ook verbazingwekkend hoe sterk het concept lijkt op de "Snelcursus assembler" van Miroslav Cina, waarvan de eerste aflevering in Elektor juli/augustus 2015 is verschenen [2].

Tijdens het werk aan dit artikel vond ik aanwijzingen op internet dat de Micro-Professor MPF-1B nog steeds wordt geproduceerd en verkocht. Ik heb dat gecontroleerd en contact opgenomen met Flite Electronics International Ltd in Waltham Chase, Hampshire, Groot-Brittannië

[3]. Ze bevestigden dat ze de rechten op de Micro-Professor hebben verworven en het apparaat nog steeds produceren. **Figuur 9** toont de Micro-Professor van Flite Electronics zij aan zij met de Taiwanese tegenhanger van Multitech. Flite Electronics was zo vriendelijk me een exemplaar te sturen, en ik kan bevestigen dat deze Micro-Professor van uitstekende kwaliteit is. Natuurlijk wilde ik alle details weten, en Max D. Soffe, managing director van Flite Electronics, stemde in met een interview (zie kader).

200679-03

Vragen of opmerkingen?

Hebt u vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur via LKL_Butte@web.de of naar de redactie van Elektor via redactie@elektor.com.



GERELATEERDE PRODUCTEN

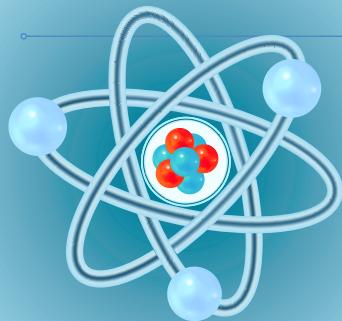
- **E-boek: Assembly Language Essentials**
www.elektor.nl/assembly-language-essentials-e-book

Een bijdrage van

Tekst en illustraties: **Karl-Ludwig Butte**
Redacteur: **Jens Nickel**
Vertaling: **Eric Bogers**
Layout: **Giel Dols**

WEBLINKS

- [1] **Butte, Karl-Ludwig:** "Vroege microprocessor-techniek – toen alle begin makkelijk was", Elektor juli/augustus 2018: www.elektormagazine.nl/magazine/elektor-201807/41722
- [2] **Cina, Miroslav:** "Snelcursus assembler", Elektor juli/augustus 2015: www.elektormagazine.nl/magazine/elektor-201507/27925
- [3] **Flite Electronics International Limited:** <http://www.flite.co.uk>



Alle begin...

...hoeft niet zo moeilijk te zijn – zelfs als het om condensatoren gaat

Eric Bogers

Na de weerstanden die we de vorige keer hebben besproken [1] zijn nu condensatoren aan de beurt. Condensatoren kunnen ook als weerstanden worden beschouwd, maar dan voor wisselstroom. Gelijkstroom laten ze namelijk niet door, wisselstroom wel. Maar condensatoren worden niet alleen als wisselstroom-weerstand gebruikt, maar ook voor de opslag van elektrische energie.



Een condensator kan worden beschouwd als twee geleiders (bijvoorbeeld metalen platen, of meer algemeen elektroden) op korte afstand van elkaar, die geen elektrisch contact met elkaar maken. Dit komt ook tot uitdrukking in het schemasymbool (**figuur 1**): twee strepen tegenover elkaar, die geen contact met elkaar maken.

Wacht even! Hoe is het mogelijk dat een



Figuur 1. Schemasymbolen van condensatoren. Van links naar rechts: 'gewone' condensator, instelbare condensator ('trimmer'), elektrolytische condensator.

component geen gelijkstroom doorlaat maar wel wisselstroom? Dat kunnen we aanschouwelijk maken wanneer we bedenken wat 'stroom' eigenlijk is: het transport van lading. Wanneer er lading wordt getransporteerd, loopt er een stroom. Als we een condensator op een gelijkspanning aansluiten, zal er een initiële stroom lopen – van de ene plaat van de condensator bewegen zich elektronen naar de pluspool, en van de minpool bewegen elektronen naar de andere plaat. Maar zodra de condensator op deze manier geladen is, kunnen de elektronen geen kant meer op – ze kunnen zich niet op magische wijze van de ene plaat direct naar de andere bewegen. Er loopt dan geen stroom meer, en de condensator gedraagt zich als onderbreking (of isolator of weerstand met oneindig hoge waarde). Dat is anders wanneer we een wisselspanning over de condensator aanleggen: dan wordt

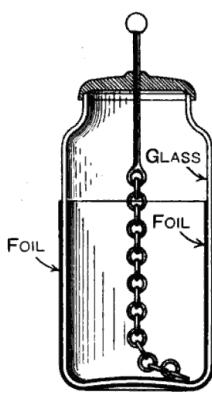
de condensator voortdurend geladen, ontladen, omgeladen enzovoort – en dan loopt er wel degelijk een stroom heen en weer: een wisselstroom dus. Even voor de goede orde: er bewegen zich geen elektronen *door* de condensator van de ene plaat of elektrode naar de andere!

Capaciteit

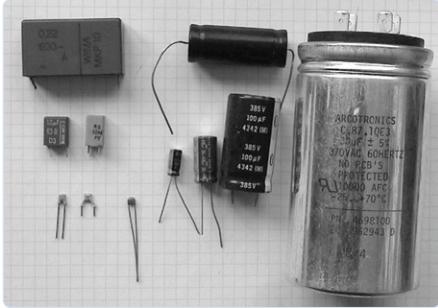
We hebben het hiervoor al even aangestuip: een condensator kan lading (en dus elektrische energie) opslaan – en wel des te meer naarmate de *capaciteit* van de condensator groter is.

Voor de capaciteit C van een condensator geldt:

$$C = \frac{\epsilon \cdot A}{d}$$



Figuur 2. Opbouw van een Leidse fles
(bron: Wikimedia Commons, https://commons.wikimedia.org/wiki/File:Leyden_jar_showing_construction.png).



Figuur 3. Enkele veelvoorkomende condensatortypen.

Hierin is ϵ de *diëlektrische constante*, A het oppervlak van de elektroden ('platen') en d de afstand tussen de platen. De formule mag u wat ons betreft meteen weer vergeten, we geven deze alleen om u een indruk te geven van de grootheden die bepalend zijn voor de capaciteit van een condensator.

Het ligt voor de hand dat de capaciteit toeneemt met het oppervlak van de elektroden ('des te groter, des te meer lading past er in'). Wanneer de elektroden dichter bij elkaar worden gemonteerd, neemt de capaciteit eveneens toe. De diëlektrische constante, die een grote invloed op de capaciteit heeft, verdient enige nadere uitleg.

Tussen de elektroden van een condensator bevindt zich een niet-geleidend medium ('materiaal'). Voor de diëlektrische constante ϵ geldt de volgende betrekking:

$$\epsilon = \epsilon_0 \cdot \epsilon_r$$

Hierin is ϵ_0 de permittiviteit van het vacuüm (een natuurconstante met de bijzonder kleine waarde $8,85 \cdot 10^{-12} \text{ C}^2/\text{N}\cdot\text{m}^2$) en ϵ_r de relatieve diëlektrische constante van het medium, een materiaaleigenschap. Voor papier heeft ϵ_r de waarde 2, voor vacuüm en lucht de waarde 1 en voor keramiek en mica de waarde 7. Het moge duidelijk zijn dat door een geschikte keuze van het diëlektricum de capaciteit van een condensator fors vergroot kan worden. De capaciteit van een condensator wordt uitgedrukt in *farad* (afgekort F). Dit blijkt in de elektronica-praktijk een onhandig grote eenheid; normaal gesproken hebben we te maken met waarden in de orde van grootte van picofarad (10^{-12} F), nanofarad (10^{-9} F) en microfarad (10^{-6} F). Er bestaan echter condensatoren met waarden van $1 \dots 10 \text{ F}$ of meer: dat zijn zogenaamde *supercaps* die onder verschillende merknamen verkrijgbaar zijn en die als een soort reservebatterij worden gebruikt om bijvoorbeeld een klokschakeling aan de praat te houden als de netspanning wegvalt.

Verschijningsvormen

De condensator deed zijn intrede in het jaar 1746 in het laboratorium van Pieter van Musschenbroek aan de Universiteit van Leiden. Deze condensator bestond uit een (glazen) fles, aan binnenvandaan en buitenzijde bekled met tinfoolie (**figuur 2**). In combinatie met een elektriseermachine die voor het laden zorgde, was deze *Leidse fles* goed voor spectaculaire (en zeker niet ongevaarlijke) experimenten.

Tegenwoordig kunnen condensatoren (gelukkig) veel kleiner worden geconstrueerd. De bekendste verschijningsvormen zijn te zien in **figuur 3**. Helemaal rechts zien we een letterlijk 'dikke' metaal/papier-condensator; deze condensatoren worden vaak voor de ontstoring van motoren gebruikt. Het voordeel is dat deze condensatoren zeer robuust zijn; het nadeel is echter dat ze niet bepaald klein genoemd kunnen worden.

Elektrolytische condensatoren (elco's) zijn beduidend compacter (**figuur 3** middelste rij). Bij elco's bestaat het diëlektricum uit een *elektrolyt*; een groot nadeel van dit type condensatoren is echter dat ze gepolariseerd zijn, dat wil zeggen een plus- en een minus-aansluiting hebben. Deze condensatoren moeten beslist in de juiste stand worden gemonteerd; vergissingen kunnen (vooral bij zogenaamde tantaalcondensatoren) spectaculaire en bijzonder onwelriekende gevolgen hebben. Er bestaan overigens ook bipolaire elco's, en daarbij maakt het weer niet uit in

welke stand (met welke polariteit) ze worden gebruikt.

De linker rij van **figuur 3** toont boven enkele polypropyleencondensatoren en daaronder een paar keramische condensatoren. Die laatste zijn lekker klein en goedkoop, maar zijn alleen in relatief kleine capaciteitswaarden verkrijgbaar.

Over de verschillende soorten condensatoren die verkrijgbaar zijn en hun specifieke eigenschappen valt zoveel te vertellen dat we daar gemakkelijk een heel nummer van Elektor mee zouden kunnen vullen. Wanneer u echter uw eerste schreden op het pad der elektronica zet, hebt u aan de bovengenoemde typen doorgaans genoeg. Al doende zult u andere typen leren kennen – maakt u zich daar geen zorgen over!

De condensator als energiereservoir

Een condensator kan elektrische energie opslaan, en wel des te meer naarmate zijn capaciteit groter is en de spanning hoger:

$$Q = C \cdot U$$

Hieruit volgt meteen de officiële definitie van de eenheid van capaciteit, de farad: de capaciteit van een condensator is de lading die bij een spanning van één volt kan worden opgeslagen. De in een condensator opgeslagen energie wordt als volgt berekend:

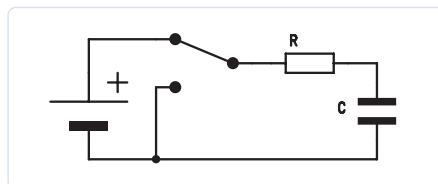
$$W = \frac{1}{2} \cdot C \cdot U^2$$

Condensatoren als energiereservoir komen we vooral tegen in netvoedingen: de (omlaag getransformeerde) net-wisselspanning wordt gelijkgericht met als resultaat een pulserende gelijkspanning; deze wordt opgeslagen in een (behoorlijk dikke) condensator die als reservoir of buffer fungeert – dat wordt het bufferen of afvlakken van de spanning genoemd, en de betreffende condensator (meestal een elco) wordt buffer- of afvlakcondensator genoemd. Zodra dioden en gelijkrichtschakelingen aan de orde komen, gaan we nader op dit onderwerp in.

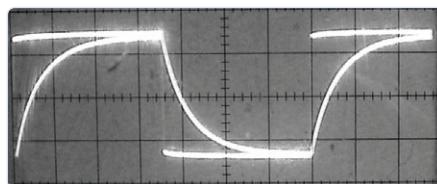
Laad- en ontladkarakteristiek

Wanneer een condensator wordt geladen of ontladen, is de spanning over die condensator afhankelijk van de al of nog aanwezige lading:

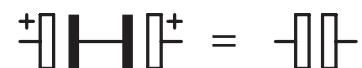
$$U = \frac{Q}{C}$$



Figuur 4. Periodiek laden en ontladen van een condensator via een weerstand.



Figuur 5. Zo zien de laad- en onlaadcycli er op het scherm van een oscilloscoop uit.



Figuur 6. Twee elco's in antiserie vormen een bipolaire elco.

Als we een condensator met een constante stroom laden, dan neemt de spanning over de condensator lineair toe – en bij ontladen met een constante stroom neemt de spanning lineair af. In de praktijk worden condensatoren echter zelden met een constante stroom geladen of ontladen; dat gebeurt veel vaker met een constante spanning (**figuur 4**). Omdat de spanning over de condensator bij het laden voortdurend toeneemt, zal de spanning over de weerstand voortdurend afnemen (de spanning over de weerstand is immers gelijk aan de constante laadspanning minus de spanning over de condensator) en dat betekent uiteraard dat de laadstroom ook voortdurend afneemt. En dat wil dan weer zeggen dat het laden van de condensator steeds trager verloopt – de spanning over de condensator nadert asymptotisch tot de laadspanning (**figuur 5**).

Bij het ontladen van een condensator via een weerstand gebeurt precies het omgekeerde: ten gevolge van de hoge condensatorspanning loopt er in eerste instantie een grote ontladestroom, maar omdat de spanning over de condensator daarbij voortdurend afneemt, zal de ontladestroom ook steeds kleiner worden.

Bij het laden van een condensator via een weerstand gelden de onderstaande formules voor de spanning en de stroom (met de afleiding ervan zullen we u niet lastigvallen; u kunt ze eigenlijk ook weer vergeten, we geven ze hier alleen omwille van de volledigheid):

$$U_C = U_0 \cdot (1 - e^{-t/RC})$$

$$I_C = \frac{U_0}{R} \cdot (e^{-t/RC})$$

En voor het ontladen gelden deze formules:

$$U_C = U_0 \cdot (e^{-t/RC})$$

$$I_C = \frac{-U_0}{R} \cdot (e^{-t/RC})$$

Serie- en parallelschakeling van condensatoren

Ter afsluiting van deze aflevering kijken we nog even naar de serie- en parallelschakeling van condensatoren. Parallelenschakeling is eigenlijk eenvoudig: wanneer we condensatoren parallel zetten, kunnen we ons voorstellen dat de 'platen' van de resulterende samengestelde condensator groter zijn geworden en dus dat de resulterende capaciteit ook groter zal zijn. Er geldt:

$$C_{total} = C_1 + C_2 + \dots + C_n$$

Maar dat is leuk! Voor het parallelenschakelen van condensatoren geldt eigenlijk dezelfde betrekking als voor het in serie schakelen van weerstanden! Zou voor de serieschakeling van condensatoren iets dergelijks gelden? Inderdaad: voor de serieschakeling van condensatoren geldt een vergelijkbare betrekking als voor de parallelenschakeling van weerstanden:

$$C_{total} = \frac{1}{\frac{1}{C_1} + \frac{1}{C_2} + \dots + \frac{1}{C_n}}$$

Nog een opmerking over elektrolytische condensatoren: wanneer twee elektrolytische

condensatoren in 'antiserie' worden geschaakeld, is het resultaat een bipolaire elco – zie **figuur 6**.

Hier laten we het voor deze keer bij; in de volgende aflevering komen condensatoren als wisselstroomweerstanden aan de orde. Daar valt best wel het en en ander over te vertellen, omdat er bij condensatoren een faseverschuiving optreedt tussen spanning en stroom – dit in tegenstelling tot weerstanden. █

200680-02

De artikelreeks "Alle begin..." is gebaseerd op het boek "Basiscursus elektronica" van Michael Ebner, dat bij Elektor is verschenen.

Een bijdrage van

Idee en illustraties: **Michael Ebner**

Tekst en redactie: **Eric Bogers**

Layout: **Giel Dols**

Vragen of opmerkingen?

Hebt u vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur of naar de redactie van Elektor, via redactie@elektor.com.



GERELATEERDE PRODUCTEN

➤ Boek: Basiscursus elektronica

www.elektor.nl/13950

➤ E-boek: Basiscursus elektronica

www.elektor.nl/18232

WEBLINK

[1] Elektor januari/februari 2021, p. 38: www.elektrormagazine.nl/200551-01

De Elektor Store

Nooit duur, altijd verrassend!

De Elektor-store heeft zich ontwikkeld van de community-shop voor de eigen producten van Elektor (boeken, tijdschriften, kits en modules) tot een volwassen webshop die veel waardevolle elektronica-aanbiedingen heeft. We bieden hier

producten aan waar we zelf enthousiast over zijn of die we gewoon willen uitproberen. Suggesties zijn altijd welkom (sale@elektor.com).

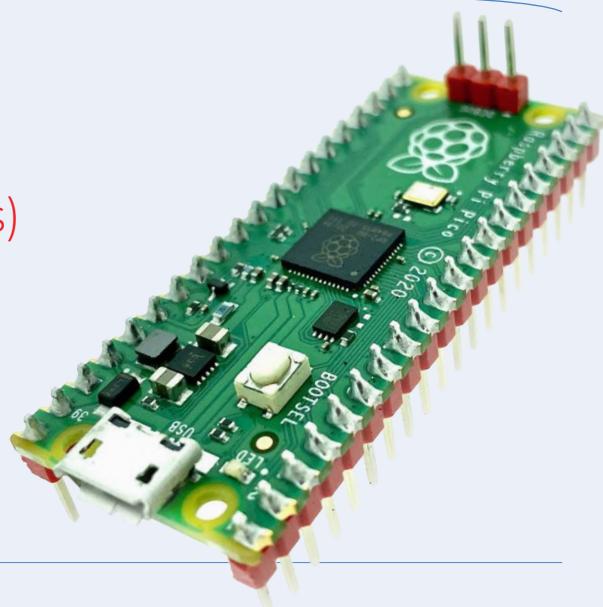
Ons motto: **nooit duur, altijd verrassend!**

Raspberry Pi Pico (met gemonteerde headers)

Prijs: € 9,95



www.elektor.nl/19568



OY-IT JT-RD6006 labvoeding (360 W)

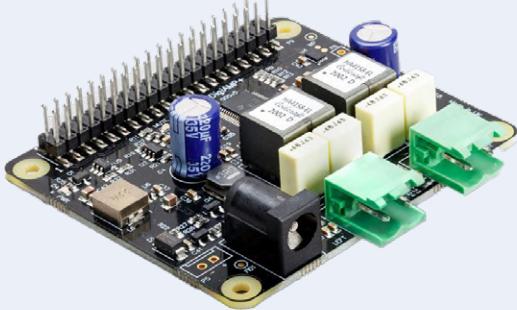


Prijs: € 89,95

Ledenprijs: € 80,96



www.elektor.nl/19564



IQaudio DigiAMP+ DAC & Class-D versterker voor Raspberry Pi

Prijs: € 29,95

Ledenprijs: € 26,96

www.elektor.nl/19538

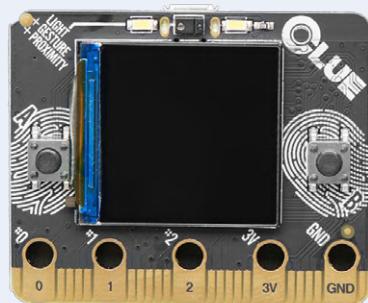


IoTize TapNLink Primer NFC-BLE-WiFi Evaluation Kit

Prijs: € 59,95

Ledenprijs: € 53,96

www.elektor.nl/19494

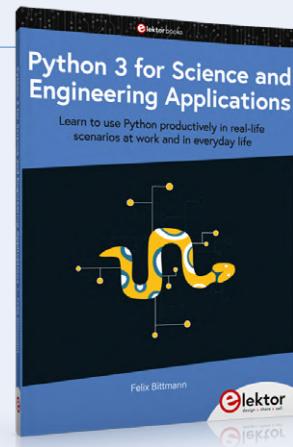


Adafruit CLUE – nRF52840 Express met Bluetooth LE

Prijs: € 49,95

Ledenprijs: € 44,96

www.elektor.nl/19512



Python 3 for Science and Engineering Applications

Prijs: € 29,95

Ledenprijs: € 26,96

www.elektor.nl/19441

Hexadoku

puzzelen voor elektronici

PC, oscilloscoop en soldeerbout kunnen weer even op adem komen terwijl u uw hersenen pijnigt met onze Hexadoku. De instructies voor deze puzzel zijn heel eenvoudig. De Hexadoku werkt met de hexadecimale cijfers 0 t/m F, helemaal in de stijl van elektronici en programmeurs.

Vul het diagram van 16 x 16 hokjes zodanig in dat **alle** hexadecimale cijfers van 0 t/m F (dus 0...9 en A...F) precies éénmaal voorkomen in elke rij, in elke kolom en in elk vak van 4 x 4 hokjes (gemarkeerd door de dikkere zwarte lijnen). Een aantal cijfers is al aangegeven

en deze bepalen de uitgangssituatie voor de puzzel. Onder de inzenders met de goede oplossing verloten we vijf waardebonnen. Om mee te dingen naar een van deze prijzen dient u **de cijfers in de grijze hokjes** naar ons op te sturen.



Doe mee en win!

Onder de internationale inzenders met het juiste antwoord verloten we vijf Elektor-waardebonnen, elk ter waarde van **50 Euro**.

Het is dus zeker de moeite waard om mee te doen!

Stuur uw antwoord (de getallen in de grijze hokjes) vóór **14 juni 2021** naar: hexadoku@elektor.nl

De prijswinnaars

De juiste oplossing van de Hexadoku uit het maart/april-nummer 2021 is: **29BF4**.

Oplossingen die ons vóór 10 april 2021 hebben bereikt, deden mee aan de trekking van vijf SparkFun MicroMod-bundels.

De winnaars zijn bekendgemaakt op www.elektormagazine.com/hexadoku.

Allemaal van harte gefeliciteerd!

	7		1	F	6		2		0	8			5		
A	4	0						5	C	2	D	7			
	9		0		5				1	7		A			
D		5		B				4	8	9			0	1	
3		A		B	E			D	C		F	0	1		
9			2			8			5	A	C	E	B		
		C		9					0	5		3			
8		E	5		F	1		3				D			
	D			2	B	C			E		6		A		
C		3	A				0		B						
F	E	8	B	3		0			2			4			
A	1	7		F	E		4		B		3		C		
B	F		C	5	0			E		1		2			
	5		E	1		2			B		C				
		A	D	4	2	6				7	B	E			
1		E	8			3	D	C	4		9				

9	A	C	F	3	8	0	1	7	E	D	2	4	B	5	6
B	0	1	D	E	A	2	5	8	3	4	6	9	C	7	F
2	8	3	E	4	F	6	7	5	9	B	C	A	D	1	0
4	5	6	7	9	B	C	D	A	F	0	1	E	2	8	3
D	9	F	A	1	E	3	B	0	4	5	8	2	7	6	C
0	B	E	5	6	2	7	4	9	1	C	A	F	3	D	8
8	4	2	6	C	9	5	F	B	D	3	7	1	E	0	A
C	1	7	3	8	D	A	0	2	6	E	F	5	4	9	B
5	C	8	B	7	3	1	A	4	2	6	D	0	9	F	E
E	6	4	9	B	0	F	8	C	A	7	3	D	1	2	5
3	F	D	1	2	C	4	6	E	0	9	5	8	A	B	7
A	7	0	2	D	5	9	E	F	8	1	B	3	6	C	4
6	D	9	8	A	4	E	C	1	B	F	0	7	5	3	2
7	E	A	0	5	6	8	3	D	C	2	9	B	F	4	1
F	2	5	C	0	1	B	9	3	7	A	4	6	8	E	D
1	3	B	4	F	7	D	2	6	5	8	E	C	0	A	9

Medewerkers van Elektor International Media en hun familieleden zijn van deelname uitgesloten.

Word lid van de Elektor Community

Neem nu een lidmaatschap!



- Een compleet web-archief t/m 1980!
- 6x Elektor Magazine (Print)
- 9x digitaal (PDF) inclusief Elektor Industry (EN)
- 10% korting in onze webshop, en exclusieve aanbiedingen
- Elektor's jaarlijkse DVD-ROM

- Samen ontwikkelen met duizenden leden van het online LAB, met toegang tot meer dan 1000 Gerberfiles, en een directe lijn naar onze experts!
- Breng je eigen project tot publicatie of zelfs verkoop in onze shop

Ook verkrijgbaar

Het digitale lidmaatschap!



- Toegang tot ons web-archief
- 10% korting in onze webshop
- 6x Elektor Magazine (PDF)
- Exclusieve aanbiedingen
- Toegang tot meer dan 1000 Gerberfiles



www.elektor.nl/member



CONTEST

Plaats een foto van je eerste
of favoriete Elektor-cover
op sociale media.

- 1 Tag @ElektorLabs
- 2 Voeg #Elektor60 toe

Doe mee en win een van
onze leukle prijzen!

More info:
www.elektor.nl/contest-Elektor60

