

ESP32-gebaseerde

wekker

maakt verbinding met
een tijdserver,
bestuurt radio en TV



FOCUS ON

Embedded & AI

Hoe verder met AI en
embedded systems?
tools, platforms en
obsolete schrijvers



RP2040 PIO in de praktijk
experimenten met
de programmeerbare I/O



Ontmoet ons op embeddedworld

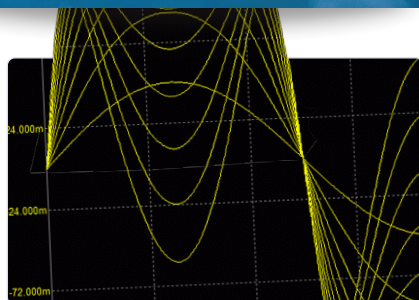
Exhibition & Conference

... it's a smarter world



Video-output met
microcontrollers
VGA en DVI

p. 92



Simuleren met Micro-Cap
eerste stappen in een
complexe wereld

p. 48



Armeluis chiptweaker
doe-het-zelf hardware-hacks

p. 22



Prototypeplaten die samenwerken met uitbreidingsmodules



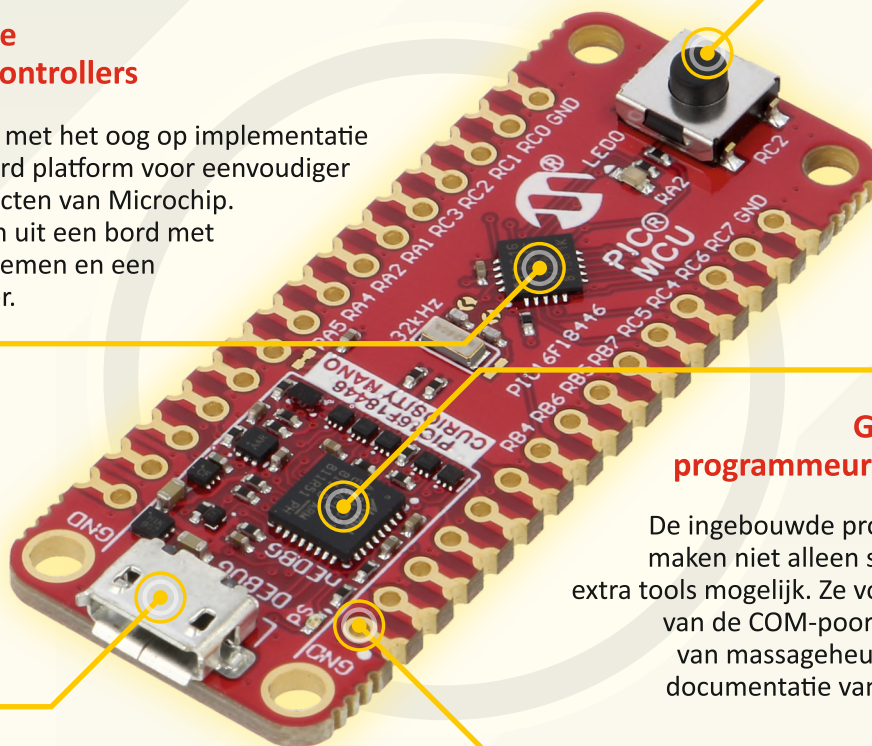
Basiscomponenten

Curiosity Nano ontwikkelingsborden hebben een knop, led, kwartsresonator en programmeerbare spanningsregelaar. Zo zijn voor basaal prototypen en werken met het bord geen aanvullende accessoires nodig.



Brede keuze aan microcontrollers

De serie is ontwikkeld met het oog op implementatie van een genormaliseerd platform voor eenvoudiger migratie tussen producten van Microchip. Klanten kunnen kiezen uit een bord met 8-bits PIC- of AVR-systemen en een 32-bits microcontroller.



Geïntegreerde programmeur en debugger



De ingebouwde programmeur en debugger maken niet alleen software-updates zonder extra tools mogelijk. Ze vormen ook een emulator van de COM-poort en vervullen de functie van massacheugen waarop de volledige documentatie van het bord is opgeslagen.



USB-interface

Voor het programmeren en uitvoeren van je code op het platform MC Nano hoef je het bord alleen maar aan te sluiten op een USB-poort van je computer en vervolgens het .hex-bestand naar het apparaat te kopiëren.

Genormaliseerde aansluitingen



Ongeacht het gekozen model microprocessor en zelfs de architectuur zitten de basisaansluitingen op de PCB-plaat altijd op dezelfde plek. De hoge mate van standaardisering bespaart tijd kosten tijdens prototypingsproces

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 (abonnements) 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: Senefelder Misset, Doetinchem
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 kosteloos verkrijgbaar.

Auteursrecht

Niets uit deze uitgave mag vervaelvoudigd 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 de in Elektor opgenomen schakelingen slechts voor particuliere of wetenschappelijke doeleinden vervaardigd worden en niet in of voor een bedrijf. Het toepassen van de schakelingen geschiedt 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

Hoofredacteur Elektor Magazine



Spannende tijden

Toen we enkele maanden geleden aan de planning van dit nummer begonnen, konden we niet weten dat kunstmatige intelligentie (AI) zo'n grote hype zou worden. Ik bedoel natuurlijk ChatGPT, een chatbot die niet alleen vragen op allerlei gebieden beantwoordt, maar met één druk op de knop ook verhalen, songteksten en nog veel meer kan produceren. Net als een van onze schrijvers (pagina 64) heb ook ik ChatGPT uitgeprobeerd. Ik stelde vragen over PV-eilandsystemen en opslagsystemen voor elektrische energie, en ik was onder de indruk van de resultaten. Niet alleen kon de chatbot inschatten wat de capaciteit van een LiFePo4-accu zou moeten zijn voor een typisch weekendhuis voor twee personen, hij gaf me desgevraagd ook suggesties voor zo'n accu, met fabrikant en type. Bovendien was ChatGPT zich ook 'bewust' van de beperkingen van zijn eigen verklaringen. De bot waarschuwde me dat capaciteitsberekeningen afhankelijk waren van energieverbruik en modulegrootte, en hij raadde me aan om eerst een professional te raadplegen.

Maar ChatGPT heeft natuurlijk geen echt besef van wat hij/zij/het zegt. De uitvoer is een soort gewogen gemiddelde van tientallen miljoenen mogelijke antwoorden, die

allemaal ergens in een of andere vorm op het internet te vinden zijn. De taalkundige vaardigheid is verrassend, maar bijna alles is op deze of vergelijkbare manier al eerder gehoord.

Toch zijn we bij Elektor gefascineerd. Ik kan me goed voorstellen dat de bot ons en u in de toekomst zou kunnen helpen – bijvoorbeeld bij het zoeken naar een bepaald Elektor-artikel in ons archief, want daar ligt tenslotte ter waarde van 62 jaar aan gebundelde elektroniekakennis opgeslagen. Misschien kan hij ook helpen bij het schrijven van standaardcode, het opstellen van onderdelenlijsten en het ontwerpen van standaardschakelingetjes?

Althans op middellange termijn ben ik niet bang dat ChatGPT onze banen – of uw inbreng in ons blad, beste lezers – overbodig zal maken. Tot nu toe krijgt AI geen kippenvel wanneer het iets ontdekt dat nuttig zou kunnen zijn voor eigen ontwikkelingen. AI krijgt geen idee onder de douche, noch schrijft het mij zeurderige e-mails over hoe alles vroeger beter was. En het zijn dit soort emoties die ons motiveren.

Denkt u er anders over? Welke ervaringen hebt u met AI opgedaan, en hebt u een idee waar het allemaal toe zal leiden? Laat het me weten op redactie@elektor.nl!

Elektor op embedded world

Van 14 tot 16 maart vindt in Neurenberg embedded world plaats, een niet te missen evenement voor alle microcontroller- en software-professionals. Zoals gebruikelijk is Elektor daar aanwezig. **Bezoek ons op stand 4A-646!**

Ons team



Internationaal hoofredacteur:
Redactie:

Jens Nickel (redactie@elektor.com)
Eric Bogers (redacteur NL-editie), Rolf Gerstendorf,
Dr. Thomas Scherer, Clemens Valens, Brian Tristram Williams
Mathias Claussen, Ton Giesberts, Clemens Valens
Harmen Heida, Sylvia Sopamena, Patrick Wielders
Erik Jansen

Technische redactie:
Vormgeving & Layout:
Directeur:



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

ESP32-gebaseerde wekker

maakt verbinding met een tijdserver, bestuurt radio en TV

6



Rubrieken

- 3 Spannende tijden**
- 44 Uit het leven gegrepen**
ontwerp(on)logica
- 46 Vreemde onderdelen**
UCN5804 stappenmotordriver
- 48 Developer's Zone**
simuleren met Micro-Cap
- 83 Alle begin...**
...wordt eindelijk actief!
- 118 Ethiek in actie**
De WEEF 2022 Awards: het goede vieren
- 122 Hexadoku**
puzzelen voor elektronici

Achtergrond & info

- FOCUS**
36 FFT met Maixduino
frequentiespectra vastleggen
- 53 PAUL Award 2022**
jonge technische talenten en hun creatieve ontwerpen
- 56 Mijn eerste Software Defined Radio**
in minder dan 15 minuten klaar
- FOCUS**
61 Microcontroller-documentatie verklaard (1)
de opzet van een datasheet

- FOCUS**
92 Video-output met microcontrollers (2)
VGA en DVI

- FOCUS**
110 DVI op de RP2040
vraaggesprek met Luke Wren, chipontwikkelaar bij Raspberry Pi
- 116 Display-HAT Mini**
toont het weer op de Raspberry Pi

Industry

- FOCUS**
64 Hoe verder met AI en embedded systems?
tools, platforms – en worden schrijvers obsoleet?
- 69 Digitalisering van verticale landbouw**
- FOCUS**
74 Elektor infographics
embedded en AI vandaag – en morgen
- FOCUS**
76 Een introductie tot TinyML
- FOCUS**
78 KwickPOS
kassasysteem op basis van Raspberry Pi
- FOCUS**
80 High-performance in elke klasse
computer-on-module standaarden



Mijn eerste Software
Defined Radio
in minder dan
15 minuten klaar

56



Video-output
met microcontrollers
VGA en DVI

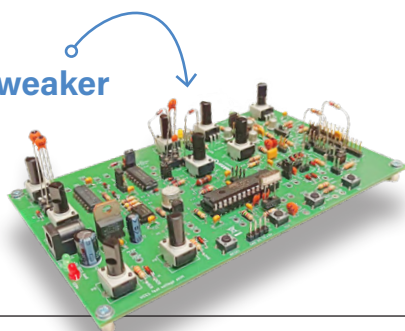
92

Projecten

- FOCUS**
- 6 Cloc 2.0**
de wekker die je altijd al wilde hebben
- FOCUS**
- 14 PIO in de praktijk**
experimenten met de programmeerbare I/O van de RP2040
- FOCUS**
- 22 Armeluis chiptweaker**
we krijgen je wel (goedkoop) aan de praat...
- FOCUS**
- 30 Echte USB-toevalsgenerator**
twee PIC's voor de prijs van één AVR
- 32 Pimp my mike...**
opvoeren doe je zelf
- FOCUS**
- 86 I²C-communicatie met Node.js en een Raspberry Pi**
bekijk sensordata in een browser
- FOCUS**
- 103 Het Metronom real-time besturingssysteem**
een RTOS voor AVR-processoren

Armeluis chiptweaker
we krijgen je wel
(goedkoop) aan
de praat...

22



Binnenkort

Elektor mei/juni 2023

Zoals u van ons gewend bent, bieden we u weer een interessante mix van projecten, schakelingen, achtergrondinformatie en tips en trucs van en voor elektronici en makers. De focus ligt daarbij op testen en meten.

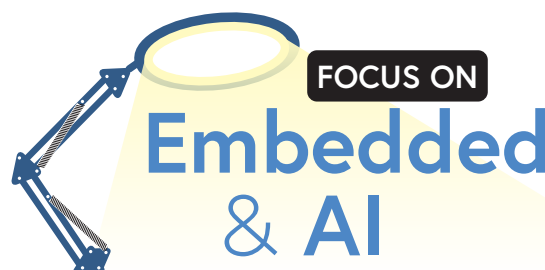
Een greep uit de inhoud:

- > Energielogger
- > Oscilloscoop-probe voor RF
- > Universele adaptieve afstandsbediening met ESP32
- > De Android WiFi-API
- > BL808: RISC-V voor AI, video en nog veel meer
- > eCO2 Telegram-bot
- > Terug naar de basis: analoge signalen en microcontrollers
- > Super-servotester
- > Programmeren van spraakgestuurde IoT-applicaties

...en nog veel meer!

Aankondigingen onder voorbehoud.

Elektor mei/juni 2023 verschijnt omstreeks 11 mei 2023.





Cloc 2.0: de wekker die je altijd al wilde hebben

Yves Bourdon (Frankrijk)

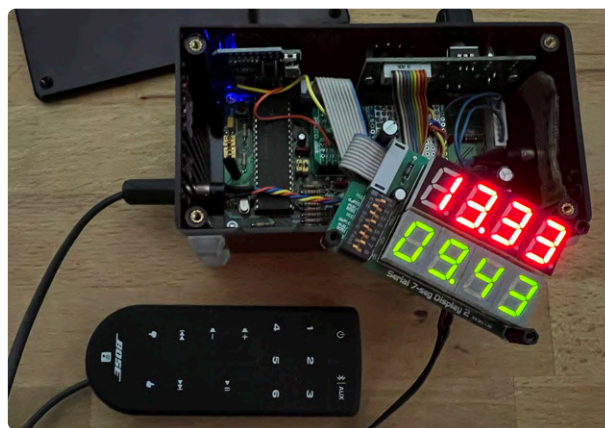
Klokken behoren waarschijnlijk tot de populairste projecten in de doe-het-zelf elektronica (naast natuurlijk radio-ontvangers). Het zou me niet verbazen als elke lezer van dit artikel al wel eens een klok heeft gebouwd.

Ik heb er in mijn leven een heleboel gebouwd, veel daarvan met een alarmfunctie. Daardoor weet ik wat ik van een wekker verlang. Omdat mijn ideale wekker niet in de handel verkrijgbaar is, heb ik hem zelf gebouwd.

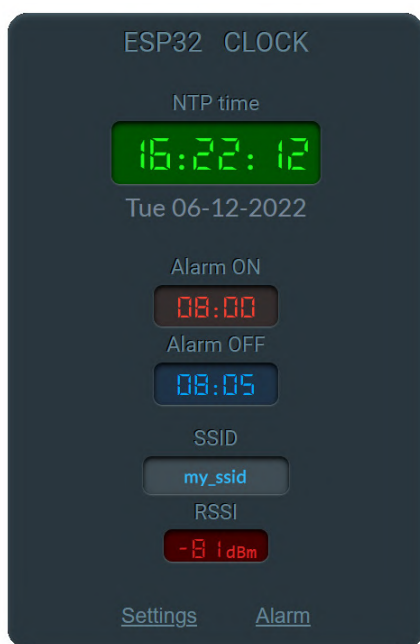
Cloc 2.0 is het resultaat en het onderwerp van dit artikel.

Mijn eerste klok bouwde ik in 1971 met Nixie-buizen uit een oud IBM-computerdisplay dat ik van iemand had gekregen. Op de print zat een indrukwekkend aantal componenten, en de nauwkeurigheid bedroeg ongeveer tien seconden per dag. In de jaren daarna heb ik nog veel klokken ontworpen en gebouwd, waaronder de beroemdste van Elektor.

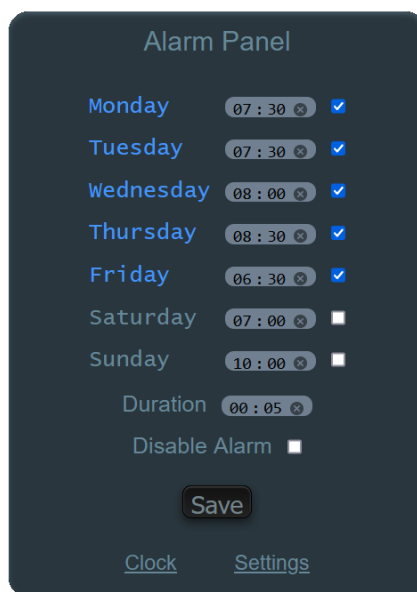
Mijn Nixie-klok deed ook dienst als wekker, en die functionaliteit heb ik in de loop der jaren steeds verder ontwikkeld door de mens-machine interface zoveel mogelijk te vereenvoudigen. Mijn laatste wekker was gebaseerd op een PIC32-processor met twee displays (**figuur 1**). Deze had een ESP8266-module om de tijd automatisch in te stellen van een online-tijdserver via NTP. De PIC32 drukte op de knoppen van een echte afstandsbediening om mijn favoriete radiostation aan te zetten of muziek af te spelen. Met een draai- en twee drukknoppen kon de wekker worden ingesteld. Dit vormde de basis van het hieronder beschreven project: *Cloc 2.0*.



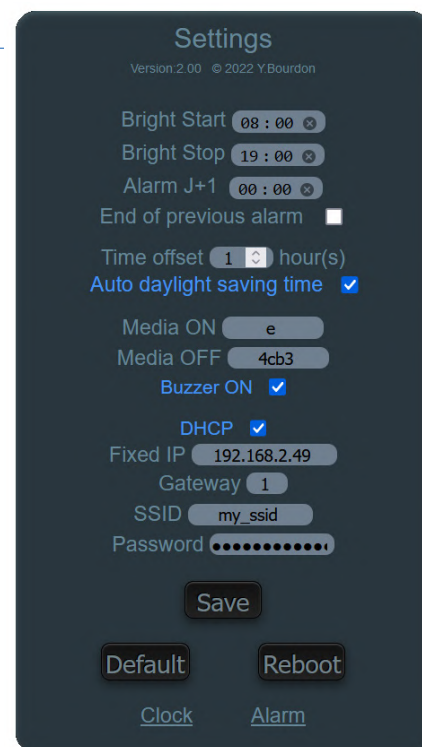
Figuur 1. Cloc 1.0, de voorloper van Cloc 2.0. Deze was gebaseerd op een PIC32 en gebruikte een echte afstandsbediening als IR-uitgang.



Figuur 2. Het Main-paneel toont naast de huidige tijd en datum tevens de belangrijkste instellingen.



Figuur 3. In het Alarm-paneel kun je de wektijd voor elke dag van de week instellen.



Figuur 4. Alle door de gebruiker te configureren parameters zijn te vinden in het Settings-paneel.

Mooi en nabouwwriendelijk

Ik wilde een gemakkelijk te bouwen wekker maken, zonder SMD-onderdelen, die zo gemakkelijk mogelijk in een goed verkrijgbare standaard-behuizing past. Ik wilde ook de retro-uitstraling van de zeven-segment-displays behouden, met een variabele helderheid die de wekker 's nachts goed afleesbaar maakt zonder storend licht als je graag in het donker slaapt.

De web-interface

Cloc maakt verbinding met een WiFi-netwerk om toegang te krijgen tot een tijdserver (waar ook ter wereld). Hij kan verbinding maken via DHCP of met een vast IP-adres. Als de verbinding tot stand gekomen is, kun je de webserver benaderen om de tijd te weten te komen, de alarmen in te stellen en de parameters van de klok aan te passen. De geoptimaliseerde grafische HTML-interface werkt op de meeste standaardbrowsers en draagbare apparaten. De interface omvat drie panelen (of schermen, zo je wilt): Main, Alarm en Settings.

Main-paneel

Het Main-paneel (figuur 2) toont de momentele tijd en datum, en de begin- en eindtijd van het komende alarm. Het toont ook de SSID van het WiFi-netwerk waarmee het is verbonden, plus de signaalsterkte RSSI. De kleur van dit laatste item (groen, oranje of rood) hangt af van de waarde ervan. Het Main-paneel wordt ongeveer elke seconde ververs.

Alarm-paneel

Het Alarm-paneel (figuur 3) wordt gebruikt om de alarmtijden voor elke dag van de week in te stellen. Het uitvinken van een weekdag schakelt het alarm voor die dag uit. Het is ook mogelijk om met één klik alle alarmen uit te schakelen.

Ook de duur van het alarm kan worden ingesteld. Aan het begin van

een alarm wordt het **Media ON**-commando verzonden via de infrarood-LED; aan het einde van het alarm wordt het IR **Media OFF**-commando verstuurd. Bovendien kan het alarm de zoemer laten klinken (in te schakelen in het Settings-paneel).

Telkens wanneer een alarm-parameter wordt gewijzigd, klinkt er een korte pieptoon. Als de waarde is veranderd ten opzichte van de vorige waarde, wordt deze opgeslagen in EEPROM.

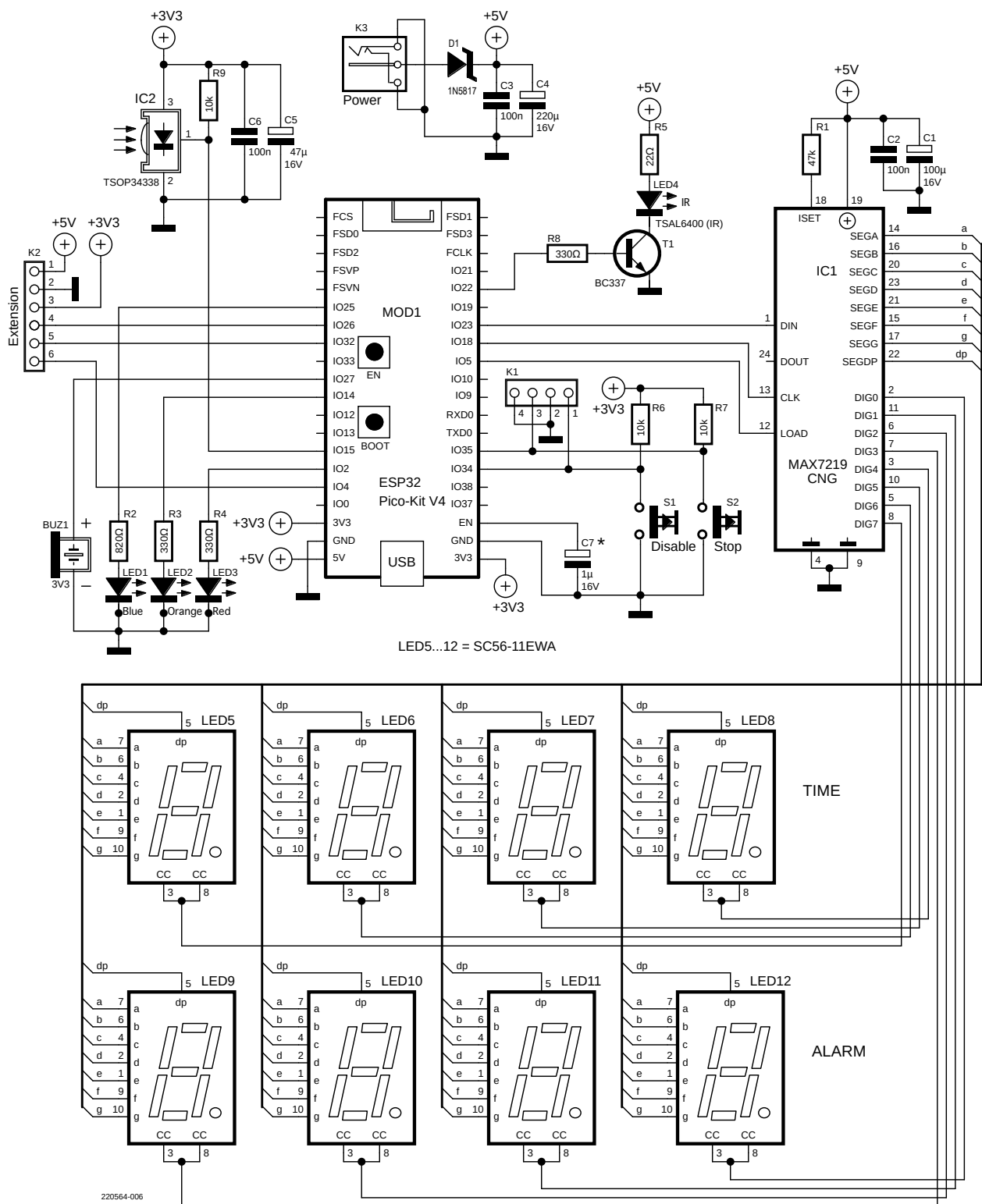
Settings-paneel

De volgende parameters kunnen worden geconfigureerd (figuur 4).

- > Tijd gedurende welke het display maximaal helder is (**Bright Start** en **Bright Stop**).
- > Het moment waarop de komende alarmtijd wordt getoond (**Alarm J+1**), bijvoorbeeld 's avonds als je naar bed gaat of onmiddellijk na het laatste alarm (door het vakje **End of previous alarm** aan te vinken).
- > Zomertijd: automatisch of handmatig.
- > Infrarood (IR) **Media ON**- en **Media OFF**-code om een ander apparaat (zoals een radio) te bedienen.
- > In-/uitschakelen van de zoemer.
- > WiFi-netwerkgegevens met DHCP of een vast IP-adres.

Druknoppen

Cloc heeft twee drukknooppjes als gebruikersinterface: S1 (**Disable**) en S2 (**Stop**). Bij normaal gebruik wordt met S1 het alarm in- of uitgeschakeld (het onderste display gaat aan of uit). Je kunt het alarm ook via de webinterface (de)activeren. Als S1 bij het inschakelen van de Cloc ingedrukt wordt gehouden, worden de standaardwaarden hersteld zoals die in het programma zijn vastgelegd (en die je voor het programmeren van de klok aan je eigen behoeften moet aanpassen!). Dit moet je doen als je de klok voor het eerst inschakelt.



Figuur 5. De ESP32-PICO-KIT zorgt voor de internetverbinding en de tijd, terwijl de MAX7219 acht zeven-segment-cijfers in twee rijen aanstuurt. Let op de IR-interface rond IC2 (ingang) en LED4 (uitgang).

Als er in de normale modus een alarm is afgegaan, wordt dit onmiddellijk gestopt met een druk op S2, het onderste display stopt dan met knipperen. Als je op S2 drukt zonder dat er een alarm afgaat, wordt de IR *Media ON/OFF*-code naar het geselecteerde apparaat gestuurd. Zo kun je de klok gebruiken als eenvoudige afstandsbediening.

Als S2 wordt ingedrukt tijdens het inschakelen, gaat de klok naar de Access Point-modus (AP) op adres 192.168.4.1. Je kunt er dan verbinding mee maken met een mobiele telefoon, een tablet of een computer om toegang te krijgen tot de webinterface waar je elke door de gebruiker programmeerbare parameter van het apparaat kunt configureren.

Beschrijving van de schakeling

Het schema van Cloc 2.0 is getekend in **figuur 5**. Bijna tweederde daarvan wordt in beslag genomen door de twee zeven-segment-displays en bijbehorende driver IC1. Om de acht cijfers, vier per display (tijd en alarm) aan te sturen, heb ik de MAX7219 gebruikt. Deze heeft een 2-draads SPI-bus (DIN wordt niet gebruikt) en is eenvoudig te programmeren dankzij open-source bibliotheken. Met weerstand R1 kan de maximale helderheid van het display worden ingesteld (hoe hoger de waarde, hoe zwakker het display). Merk op dat high-brightness zeven-segmentdisplays wel heel erg veel licht geven, misschien te veel voor een wekker.

De voor Cloc gekozen ESP32-module is de ESP32-PICO-KIT, die niet alleen veel I/O-poorten heeft maar ook de perfecte afmetingen voor dit project. Condensator C7 is alleen nodig als de ESP32-module na het opstarten in de reset-modus blijft hangen. Sommige (oudere?) ESP32-modules schijnen last te hebben van dit probleem, maar ik heb er nooit iets van gemerkt. Als je besluit hem te monteren, monteer hem dan liggend en van R6 en R7 af wijzend.

De alarm-buzzer BUZ1 (zonder geïntegreerde oscillator) wordt aangestuurd door een PWM-sigitaal op poort IO27 van de ESP32.

De poorten IO34 en IO35 van de ESP32 zijn verbonden met de drukknoppen S1 en S2 en connector K1. In een typische opbouw zouden S1 en S2 bovenop de klok worden gemonteerd in plaats van op de print, vandaar de aanwezigheid van K1. Er zijn weerstanden van 10 k Ω gebruikt (R6 en R7), hoewel de ESP32 ingebouwde pull-up weerstanden heeft. Maar omdat die waarden hebben van enkele honderden kilo-ohms, is een alleen daarop vertrouwend knop iets gevoeliger voor ruis.

Drie (debug-)LED's (LED1, LED2 en LED3) geven de status van de WiFi-netwerkverbinding (blauw), de verbinding met de webserver-interface (oranje) en het IR in- en uitgaand verkeer (rood) aan.

IR-interface

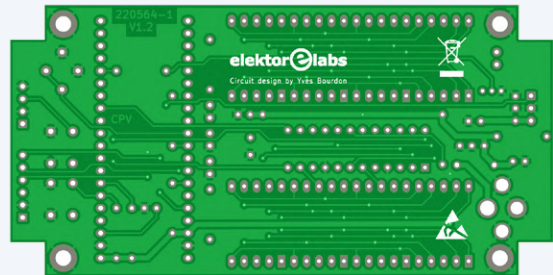
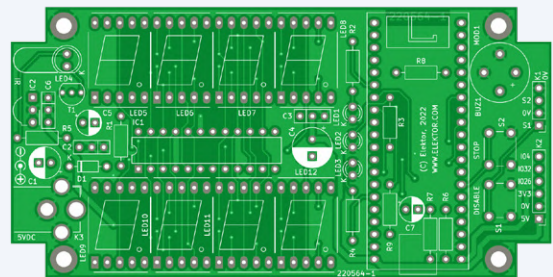
C5 en C6 ontkoppelen infraroodontvanger IC2 om een zo schoon mogelijk IR-sigitaal te krijgen. De infrarode TX-LED (LED4) wordt aangestuurd door transistor T1. De stroom er doorheen wordt door R5 beperkt tot ongeveer 150 mA. De IR-uitgang is krachtig genoeg om alle apparatuur in de buurt op dezelfde manier te bedienen als de oorspronkelijke afstandsbediening dat zou doen.

Voeding

De externe 5V_{DC}-voeding moet minstens 500 mA kunnen leveren. Diode D1 beschermt de schakeling tegen ompolen. Twee elektrolytische condensatoren (C1 en C4) vlakken stroompieken af. De kleine parallel aan de elco's geschakelde condensatoren van 100 nF (C2 en C3) dempen ruis bij hogere frequenties.



ONDERDELENLIJST



Weerstanden:

R1 = 47 k
R2 = 820 Ω
R3,R4,R8 = 330 Ω
R5 = 22 Ω
R6,R7,R9 = 10 k

Condensatoren:

C1 = 100 μ /16 V
C2,C3,C6 = 100 n
C4 = 220 μ /16 V
C5 = 47 μ /16 V
C7 = 1 μ /16 V

Halfgeleiders:

D1 = 1N5817
IC1 = MAX7219
IC2 = TSOP34338
LED1 = blauw, 3 mm
LED2 = oranje, 3 mm
LED3 = rood, 3 mm
LED4 = TSAL6400, IR LED, 5 mm
LED5-12 = 7-segment cijfer 0,56", CC (bijv. SC56-11EWA)
T1 = BC337

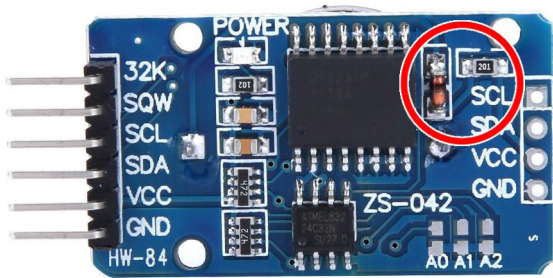
Diversen:

BUZ1 = Piëzo-zoemer zonder oscillator, 12 mm, 3,3 V
K1 = eenrijige 4-polige pinheader
K2 = eenrijige 6-polige pinheader
K3 = voedingsconnector
MOD1 = ESP32-PICO-KIT
S1, S2 = druktoets 6x6 mm

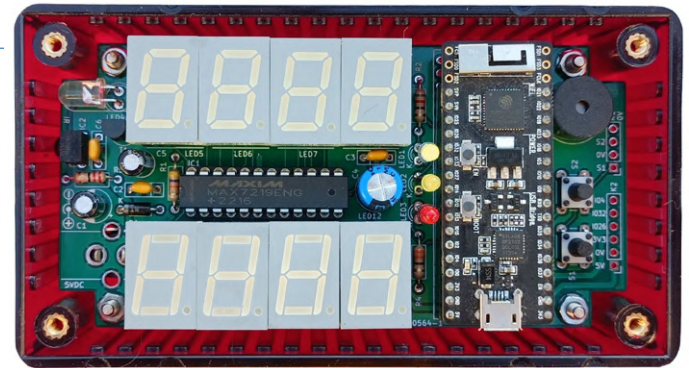
print 220465-1

behuizing Hammond 1591CTDR (transparant rood)

DS3231 real-time klokmodule



Figuur 6. Als de (optionele) DS3231 real-time klokmodule (RTC) wordt gebruikt met een niet-oplaadbare CR2032-knoopcel, moet je de weerstand of de diode (of beide) in de rode cirkel verwijderen om beschadiging van de batterij te voorkomen.



Figuur 7. De opgebouwde print past goed in de behuizing. Je kunt de print in het deksel monteren om de schroeven van de behuizing te verbergen. De voedingsconnector kan aan beide zijden van de printplaat worden gemonteerd.

Uitbreidingen

Via connector K2 kan de klok worden uitgebreid met bijvoorbeeld een real-time klokmodule (zie de volgende paragraaf). Hiervoor zijn drie I/O-poorten beschikbaar: IO4, IO26 en IO32.

Optionele real-time klok (RTC)

Op connector K2 kan een DS3231 RTC-module worden aangesloten. De RTC neemt het over als de NTP-server om een of andere reden onbereikbaar is (bijvoorbeeld WiFi is uitgevallen, of er is een probleem bij de internetprovider). De RTC-module wordt automatisch gedetecteerd en geconfigureerd door de software en eenmaal per dag (om 12:00 uur) bijgewerkt. De DS3231 is een krachtige chip met ingebouwde compensatie voor drift van het kristal dankzij een geïntegreerde temperatuursensor.

De DS3231 RTC-module heeft een I²C-interface en voor de aansluiting op K2 zijn slechts vier signalen nodig: GND, 3,3 V, SDA en SCL. Als een RTC-module wordt gedetecteerd, wordt poort IO26 het SDA-signaal, terwijl poort IO32 het SCL-signaal van de I²C-bus verzorgt.

Pas op!

Wees voorzichtig, deze RTC-modules kunnen gevaarlijk zijn! De module is inderdaad van een laadschakeling voorzien, maar vaak is een niet-oplaadbare CR2032-batterij gemonteerd, die op termijn kan exploderen. Er zijn twee oplossingen voor dit probleem. De gemakkelijke maar duurdere is de batterij te vervangen door een oplaadbare LIR2032-accu. De tweede oplossing is het verwijderen van de diode of de 200Ω-weerstand (of beide) bij de SCL-pin van de 4-polige connector, zie **figuur 6**.

Mechanische details

Een goede behuizing voor Cloc is de 1591CTDR van Hammond, die 120 x 63 x 38 mm meet. Het is een rode (IR-transparante) behuizing die perfect geschikt is voor de rode en/of oranje zeven-segment-displays voor de (wek)tijd. Ook het zenden en ontvangen van IR-signalen profiteert van een dergelijk optisch filter. Merk op dat licht met een gering (of zonder) rood-aandeel (zoals de blauwe LED) nauwelijks zichtbaar is bij deze filtering.

En nu we het toch over LED's hebben, de ESP32-PICO-KIT heeft een (veel te) heldere rode power-LED die je waarschijnlijk wilt verbergen of demonteren.

De print is ontworpen voor deze behuizing. Het KiCad6-project kan worden gedownload van [1]. De print, gemonteerd op vier afstandhouders van 16 mm, past perfect in de behuizing, zoals te zien is in **figuur 7**.

In de bodem hoeven slechts vier gaten van 3,2 mm te worden geboord om de print te bevestigen, aan de zijkant is een gat van 8 mm nodig voor de voedingsconnector; twee gaten van 12 mm aan de bovenkant maken het mogelijk om twee fraaie drukknoppen te monteren. De optionele RTC-module kan onder de hoofdprint worden aangesloten op K2 met de componentzijde naar boven met behulp van de niet-gebruikte vierpolige header (de 6-polige schuine header kan eventueel worden verwijderd om kortsluiting te voorkomen). Je kunt de module ook op de hoofdprint plakken met dubbelzijdig plakband (zoals ik heb gedaan).

De software-ontwikkelomgeving

Ik heb de Arduino IDE gebruikt om de software voor Cloc 2.0 te ontwikkelen, met het ESP32 boards-package geïnstalleerd. Om compilatie- of geheugenbeheerfouten te voorkomen, moet je het ESP32 PICO-D4 board kiezen met een CPU-frequentie van 240 MHz. Voor het *Partition Scheme* kies je *Default* (dat wil zeggen 4 MB met SPIFFS). Als je de juiste seriële poort selecteert, zou het compileren en laden van het programma probleemloos moeten verlopen!

De volgende externe bibliotheken moeten in de IDE worden geïnstalleerd voordat het programma wordt gecompileerd. Ik heb in de broncode aangegeven waar ze gevonden kunnen worden. De meeste kunnen ook worden verkregen met behulp van de *Library Manager* van de IDE.

- AsyncElegantOTA
- AsyncTCP
- ESPAsyncWebServer
- IRremote (zie regel 126 van `Alarm_Clock.ino`)
- LedControl
- RTCLib

Belangrijk: `LedControl.h` moet worden aangepast. Regel 30 (`#include <avr/pgmspace.h>`) moet worden weggecommentarieerd. Zie ook het commentaar in de broncode.

De download [1] bevat een map genaamd *libraries* die alle gebruikte bibliotheken bevat, inclusief de al gemodificeerde `LedControl.h`. Kopieer deze bibliotheken naar de submap *libraries* van de map Arduino sketchbook (je kunt zien waar die staat door naar het Preferences-venster van de IDE te gaan).

Raadpleeg [2] om te leren hoe je de *ESP32 Sketch Data Upload* optie toevoegt aan de IDE. Die heb je nodig om de webinterface van de klok in het datageheugen van de ESP32-module te laden. Vergeet dat niet, anders werkt de webserver niet.

Over-the-Air (OTA)

Om de firmware van de wekker op afstand bij te werken, ga je met een browser naar [IP_of_cloc/update](#), waarbij [IP_of_cloc](#) het IP-adres van de wekker is. Je kunt nu een executable op je computer kiezen en deze naar de klok uploaden. Een gedetailleerde uitleg van de OTA-interface is beschikbaar op [3].

Het programma nader bekeken

Het programma is vrij goed gedocumenteerd (helaas vaak in het Frans), en na het lezen van het volgende zou je gemakkelijk je weg moeten vinden. Je zou de seriële poort van de ESP32 kunnen monitoren, omdat er veel debug-informatie via die poort wordt verstuurd.

Na het opnemen van de bibliotheken worden de defaultwaarden gedefinieerd die worden gebruikt bij de eerste start (door S1 (alarm uit) ingedrukt te houden, of vanaf de Settings-pagina na het aanklikken van Defaults. De volgorde van de constanten is gelijk aan de volgorde van de parameters in het Settings-paneel (regels 40 tot 65 van [Alarm_Clock.ino](#)). Controleer met name het vaste IP-adres en de gateway, alsmede de WiFi-gegevens. Het Alarm-paneel (regels 67 tot 84) is op dezelfde manier georganiseerd.

Bij het inschakelen wordt eerst de functie [setup](#) uitgevoerd. Het is gemakkelijk om de initialisatievolgorde van de Cloc-hardware te volgen. De functie [initRTC](#) controleert of er een DS3231 RTC-module is aangesloten. Als toets S1 op dat moment wordt ingedrukt, worden de default-waarden geladen.

Als dit de eerste keer is dat de klok wordt opgestart of als de Stop-knop (S2) wordt ingedrukt tijdens het opstarten, gaat de klok naar de Access Point-modus (AP). In dit geval kun je met een browser verbinding maken met zijn webinterface op 192.168.4.1, zoals aangegeven door de klok.

Specificaties

- › Gebaseerd op een ESP32-processor.
- › Twee onafhankelijke zeven-segment-displays voor de tijd van de dag en de alarmtijd.
- › De tijd wordt automatisch ingesteld met behulp van een online-tijdserver.
- › Verschillende alarmtijden voor elke weekdag.
- › Alarmuitgang: zoemer en IR-code voor bijvoorbeeld radio, hifi-installatie of TV.
- › Twee drukknoppen voor interactie met het apparaat.
- › Geïntegreerde webserver voor configuratie op afstand via WiFi.
- › Alle instellingen worden opgeslagen in EEPROM.
- › Over-the-Air modus (OTA) maakt het mogelijk de firmware op afstand bij te werken.
- › Optionele DS3231-gebaseerde real-time klokmodule (RTC) met backup-batterij.

Na het ophalen van het IP-adres (DHCP of vast) en de verbinding met het WiFi-netwerk zoeken we naar de NTP-tijd. Als die wordt gevonden en er een RTC aanwezig is, worden deze gesynchroniseerd. Het IP-adres van de klok wordt kort weergegeven voordat wordt overgeschakeld naar de tijdweergave. Als op het tijddisplay "HH:HH" begint te knipperen en op het alarmdisplay "AA:AA", kon de NTP-server niet worden bereikt en moet je het apparaat opnieuw opstarten (als er geen RTC aanwezig is; anders wordt de RTC-tijd weergegeven).

De initialisatie eindigt met het starten van de webserver en de [AsyncOTA](#)-service, waarmee de firmware en SPIFFS op afstand kunnen worden bijgewerkt.

De functie [loop](#) voert de rest van het programma uit in een oneindige lus. Als de RTC aanwezig is ([flagRTC=1](#)), wordt de RTC-tijd gelezen. Zonder RTC wordt de NTP-tijd ([NTPflag=1](#)) geprobeerd. Zodra de klok is aangesloten, leest het programma de RTC-tijd, terwijl [strTime](#) de NTP-tijd bevat als "hh:mm:ss" als deze geldig is of als "0" als dat niet het geval is.

Vervolgens wordt de toestand van de knoppen gecontroleerd ([checkButtons](#)) en wordt gekeken of de klok alarm moet geven en of er een IR-code is ontvangen. Aangezien de server asynchroon is, moeten we vlaggen gebruiken om parameterwijzigingen en alarm-triggers aan het hoofdprogramma te signaleren.

Het scherm wordt elke seconde ververs ([dispTime](#)) en de alarmen worden gecontroleerd ([checkAlarm](#)). Ook controleert het programma of het display maximaal helder moet zijn of niet ([checkBrightness](#)) en of het de RTC moet synchroniseren met de NTP-tijd ([checkSyncRTC](#)). Dit gebeurt eenmaal per dag om 12:00 ([syncRTC=12*60](#)). De functie [checkTimer](#) controleert of een alarm moet worden geactiveerd. Hiervoor worden de tijd van de dag en de alarmtijd omgezet in minuten ([computeMinutes](#)) alvorens ze te vergelijken ([testRange](#)).

Zoals al opgemerkt, is de server asynchroon. Telkens wanneer het Settings- of Alarm-paneel wordt ververs, worden de weer te geven gegevens ([ParameterProcessor](#) en [TimerProcessor](#)) geretourneerd. Hiervoor moeten de `%value%`-tags in de HTML-pagina worden vervangen door de werkelijke waarden.

Voor het Main-paneel is het een beetje anders, omdat we bijna in real time de tijd van de dag, de datum en de start- en stoptijd van het alarm moeten weergeven. Een in JavaScript geschreven functie vraagt ongeveer elke seconde het hoofdprogramma op om de weer te geven waarden op te halen ([onDataUpdate](#)). Telkens als dat gebeurt, knippert LED2 kort om aan te geven dat de server wordt benaderd ([flashRqstBeacon](#)). Om te controleren of de pagina inderdaad de waarden in real time weergeeft, kun je op S1 (Alarm Disable) drukken. De alarmtijd wordt tegelijkertijd weergegeven of uitgeschakeld op zowel het zeven-segment-display als op de HTML-pagina.

Over de IRremote-bibliotheek

De [IRremote](#) bibliotheek werkt prima, maar vereist enkele integratieregels. Verplaats de initialisatieregels niet, tenzij je weet wat je doet. Om wakker te worden met muziek in plaats van een piep, gebruik ik een Bose-radio. Daarom heb ik de ontvangst en decodering beperkt tot Bose-afstandsbedieningen (zie regel 125 van [Alarm_Clock.ino](#)). Je kunt deze regel verwijderen om andere afstandsbedieningen te decoderen. Ik raad echter aan de IR-ontvangst te beperken tot het merk van je radio of hifi-systeem. Zie [4] voor mogelijke opties.

Ook de IR-transmissie is beperkt tot het Bose-formaat. Daarom is het zeer waarschijnlijk dat je de code moet aanpassen aan het merk van de hifi-apparatuur die je gebruikt. Vervang de regel `IrSender.sendBoseWave` in de functies `startMedia` en `stopMedia` in `Alarm_Clock.ino` door een functie die overeenkomt met je eigen merk (zie ook [4]).

Om erachter te komen welke IR-codes je voor je (audio-)systeem moet gebruiken, moet je de USB-poort van de ESP32-module aansluiten op een computer met een seriële monitor, bijvoorbeeld de in de Arduino IDE ingebouwde *Serial Monitor*. Richt de afstandsbediening op de IR-ontvanger van de klok en druk op een toets om de bijbehorende code in hexadecimaal formaat op de computer weer te geven. Voer de codes in hexadecimale vorm in de velden *Media ON* en *Media OFF* van het Settings-paneel in.

IR-activiteit wordt aangegeven door LED3 die kort knippert. Deze LED knippert dus ook als er een alarm actief is.

Een DHZ-oplossing

Ik gebruik deze nieuwe versie van Cloc 2.0 nu enkele maanden, en ik heb me nog nooit verslapen omdat hij niet goed werkte. De mogelijkheid om alarmtijden voor elke dag van de week in te stellen is erg handig.

Ik raad sterk aan de RTC-optie toe te voegen, want als de NTP-server niet reageert (bijvoorbeeld omdat je modem om een of andere reden offline is gegaan), zal de wekker de ingestelde tijd verliezen zodra weer een NTP-synchronisatie plaatsvindt. Je moet de wekker dan stoppen en opnieuw starten als de NTP-tijd weer beschikbaar is. Voor nog meer betrouwbaarheid heb ik een kleine NTP-server gebouwd die de GPS-tijd ontvangt [5]. Dus zelfs als de internetverbinding wegvalt, kan mijn wekker zichzelf automatisch instellen.

Ik hoop dat velen van jullie deze klok zullen maken. Aarzel niet om me te contacteren op mijn privé e-mail (yb.electronique@orange.fr) als je moeilijkheden ondervindt of als iets aan mijn favoriete wekker verbeterd! ◀

220564-03

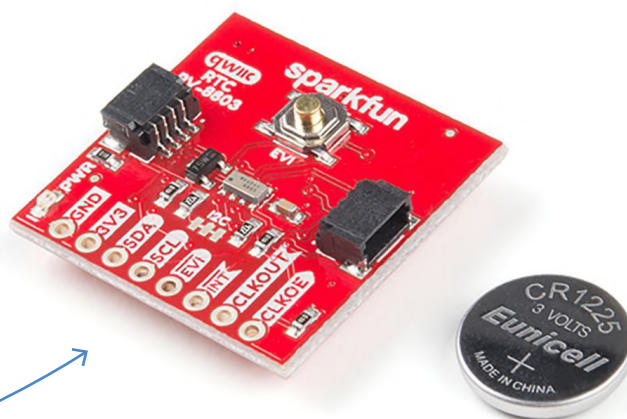
Vragen of opmerkingen?

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



Gerelateerde producten

- > **ESP32-PICO-Kit V4 (SKU 18423)**
www.elektor.nl/18423
- > **Elektor Arduino Electronics Bundle (SKU 19440)**
www.elektor.nl/19440
- > **SparkFun Real Time Clock Module - RV-8803 (Qwiic) (SKU 19646)**
www.elektor.nl/19646



WEB LINKS

- [1] Downloads voor Cloc 2.0 bij Elektor Labs: <https://www.elektormagazine.com/labs/cloc-le-reveil-20>
- [2] ESP32 SPIFFS-gegevens uploaden: <https://randomnerdtutorials.com/install-esp32-filesystem-uploader-arduino-ide/>
- [3] ESP32 OTA-updates (Over-the-Air): <https://randomnerdtutorials.com/esp32-ota-over-the-air-arduino/>
- [4] IRremote-bibliotheek: <https://github.com/Arduino-IRremote/Arduino-IRremote>
- [5] NTP-server bij Elektor Labs: <https://www.elektormagazine.com/labs/ntp-server>

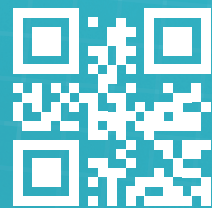
MagPi, het officiële Raspberry Pi-magazine



12 maanden
100+ Projekten
1 Prijs

€ 54,95

- ✓ Zes edities van MagPi Magazine
- ✓ Toegang tot het online MagPi-archief



NU TE BESTELLEN OP
WWW.MAGPI.NL/ABO

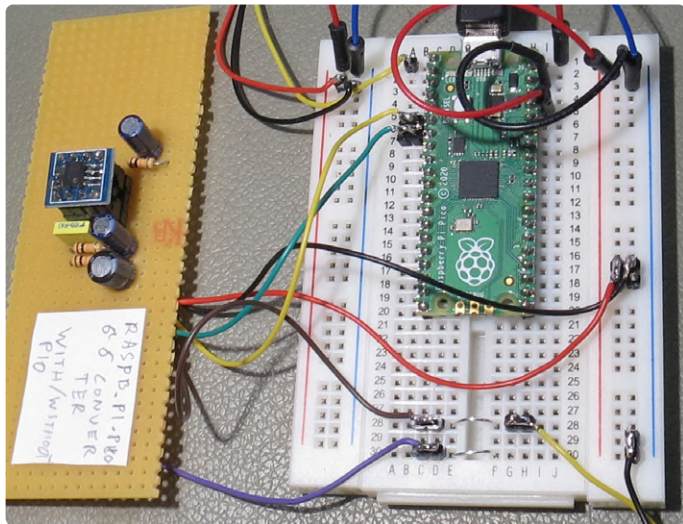
MagPi 
www.magpi.nl Magazine

PIO in de praktijk

experimenten met de programmeerbare I/O van de RP2040

Prof. dr. Martin Ossmann (Duitsland)

De Raspberry Pi Pico is een populair ontwikkelboard dat veel rekenkracht biedt bij een gering stroomverbruik. En met een prijs van net € 4 per stuk krijg je veel waar voor je geld. De RP2040-microcontroller op het board heeft een (tot nu toe) unieke eigenschap: twee PIO I/O-blokken met acht toestandsmachines. Aan de hand van enkele praktijkvoorbeelden bekijken we welke voordelen dit voor een toepassing biedt.



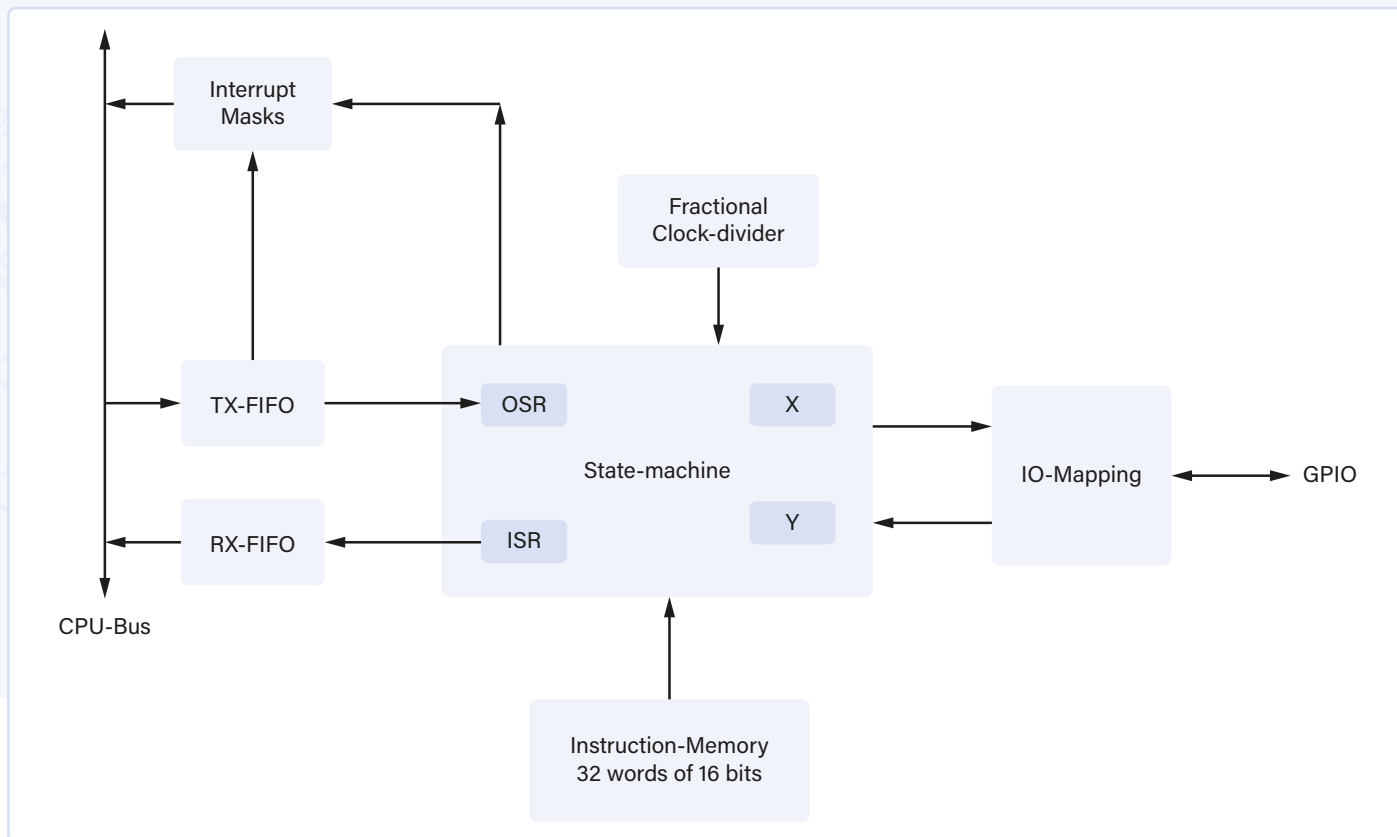
Figuur 1. Het Raspberry Pi Pico-board op een breadboard, en daarnaast de delta-sigma-print.

Als je al wat ervaring hebt met het programmeren van microcontrollers, zul je ongetwijfeld een ingebouwde programmeerbare I/O-unit op prijs stellen om de belasting van de hoofd-processorcores te reduceren. De grote vraag is of deze unit niet onnodig ingewikkeld in het gebruik is en daarom niet de moeite waard. Je zult zeker een beter beeld krijgen van de mogelijkheden als je de hier beschreven praktische voorbeelden op de in de RP2040-processor ingebouwde PIO's hebt doorlopen. **Figuur 1** toont mijn experimentele hardware-opstelling waarop ik de besproken voorbeelden heb ontwikkeld.

De PIO-unit

De flexibiliteit van de programmeerbare input/output-unit maakt het mogelijk de I/O-pinnen te gebruiken om andere soorten communicatieprotocollen te implementeren naast de bestaande standaarden zoals SPI en I²C. De unit bestaat uit twee PIO-blokken, elk met vier I/O-processoren. Elke afzonderlijke I/O-processor is geïntegreerd in de chiparchitectuur zoals geschetst in **figuur 2**.

In totaal zitten er twee PIO's in de RP2040; elk bevat vier toestandsmachines (state machines, SM's) en een instructiegeheugen van 32 woorden om de toestandsmachines aan te sturen. Elke SM heeft twee 32bit-registers X en Y en een 32-bit ingangsschuifregister (ISR) en een 32-bit uitgangsschuifregister (OSR). De PIO-kernen zijn elk verbonden met de RP2040-CPU via twee 32 bit brede FIFO's met een diepte van vier woorden. De TX FIFO-unit wordt gebruikt voor het verzenden en de RX FIFO-unit voor het ontvangen van data.



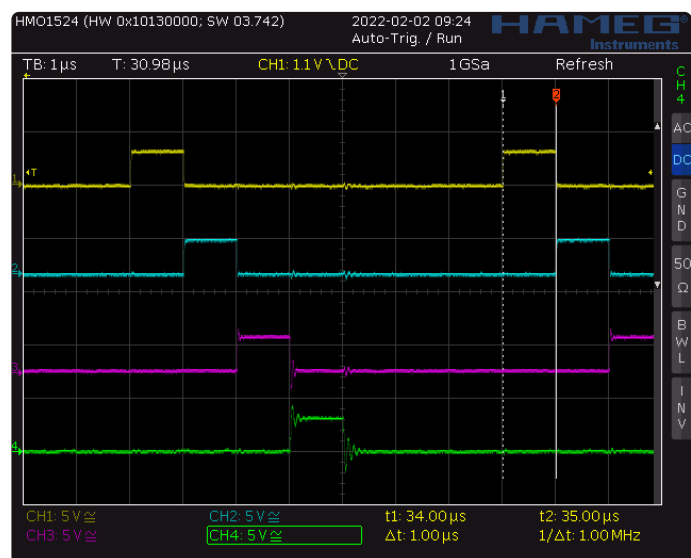
Figuur 2. Blokschema van een van de twee I/O-processoren (bron: [4]).

Als gegevens slechts in één richting gaan, kunnen de twee FIFO's worden geconfigureerd om als een enkele acht woorden diepe FIFO te werken. Elke PIO-processor heeft een 16,8-bit fractionele deler voor het genereren van het kloksignaal (zie verderop). Dit geeft de flexibiliteit om de kloksnelheid van elke processor in te stellen op een waarde tussen 2 kHz en 125 MHz.

De PIO-processoren verstaan slechts negen instructies. Deze zijn 16 bit breed en worden opgeslagen in een 32×16 bit instructiegeheugen dat de vier SM's in elk PIO-blok aanstuurt. De afzonderlijke PIO-pro-

gramma's zijn kort genoeg om in dit kleine geheugen te passen. De instructies worden hier door de RP2040 CPU opgeslagen. Deze kleine programma's worden geschreven met behulp van een speciaal maar vrij eenvoudig PIO-assemblerprogramma. Elke instructie heeft voor de uitvoering één klokcyclus nodig.

Een enkele PIO-processor kan slechts een paar GPIO-pinnen aansturen, en gebruikt daarvoor korte adressen. De IO-mapping bepaalt welke pinnen dat specifiek zijn. Er zijn verschillende programmeerbare constanten die voor het betreffende gebied (input, output, side set en set) bepalen wat de eerst geadresseerde pin is. Hierdoor kunnen de IO-pinnen van een PIO-processor zeer flexibel worden gedefinieerd. Ook kunnen meerdere processoren toegang krijgen tot dezelfde pin.



Figuur 3. De golfvormen van het eerste voorbeeld.

Wat heb je nodig

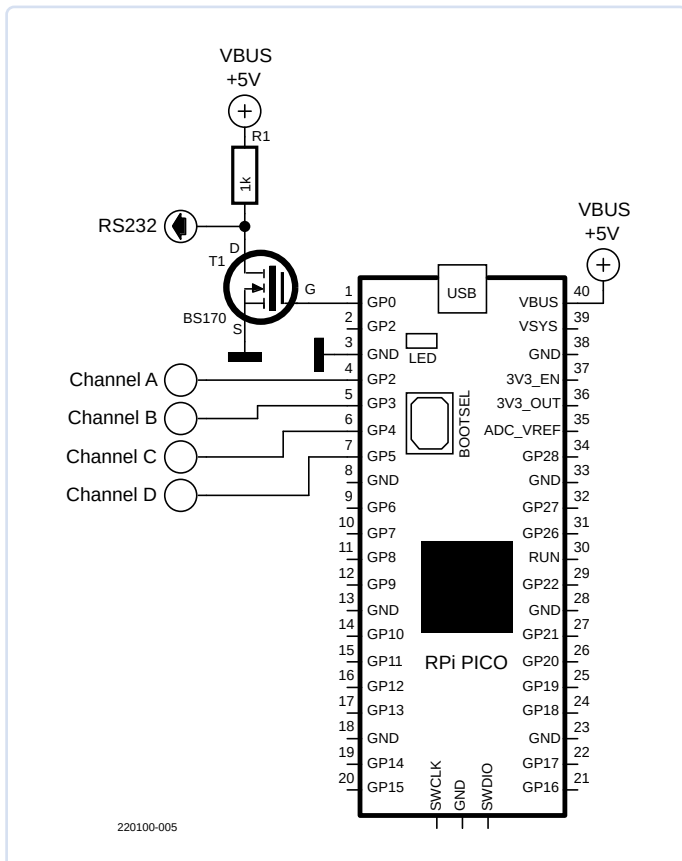
Hier wordt VS Code gebruikt als ontwikkelomgeving. De installatie en het gebruik van deze tool zijn goed beschreven in [1]. Daar vind je ook informatie over het gebruik van de projectgenerator. Hiermee wordt een Raspberry Pi-project gemaakt met alle benodigde bestanden (broncode, configuratiebestanden, make-setup enzovoort).

Je moet ook aangeven of de uitvoer van de `printf()` functie via USB of UART loopt. Hier configureer je ook welke bibliotheken je moet gebruiken. In dit artikel gebruiken we bijvoorbeeld de PIO-tool. Hulp bij de installatie van VS Code kun je vinden in [2]. Er is ook een gedetailleerde beschrijving van welke extra tools (Make, Git enzovoort) geïnstalleerd moeten worden.

Alle softwarevoorbeelden kunnen worden gedownload van [3].

Aan de slag

We willen dat dit eerste programma een puls van 1 µs via vier uitgangen na elkaar uitvoert, zoals te zien in **figuur 3**. We hebben een vierkanaals



Figuur 4. Aansluiting van een vierkanaals 'scoop op de Raspberry Pi Pico om de golfvormen van het eerste voorbeeld te bekijken.

oscilloscoop aangesloten op de vier uitgangen om de resulterende uitgangssignalen te zien. **Figuur 4** toont de aansluiting van de oscilloscoop op de I/O-pinnen van de Raspberry Pi Pico.

Listing 1 bevat het eerste voorbeeld-PIO-programma. Na drie regels assembler directives staat de eerste uitvoerbare instructie op regel 4. Deze en volgende regels genereren vervolgens de vier pulsen.

```

001 .program pioTest
002 .side_set 2 opt
003 .wrap_target
004 nop          side 0b01 // GP2=1, GP3=0
005 nop          side 0b10 // GP2=0, GP3=1
006 set pins,0b01 side 0b00 // GP2=0, GP3=0,
    GP4=1, GP5=0
007 set pins,0b10 // GP4=0, GP5=1
008 set pins,0b00 [2] // GP4=0, GP5=0
009 .wrap

```

Regels 4 en 5 bevatten de `nop`-instructies, die zoals je waarschijnlijk al vermoedt niets doen, maar regel 4 wordt uitgebreid met de tekst `side 0b01`. Dit geeft aan dat op deze regel iets anders wordt 'terzijde' wordt gedaan. In dit geval wordt side pin 0 '1' gemaakt en side pin 1 '0' gemaakt. Elke PIO-instructie kan op deze manier worden uitgebreid. Zo

kun je gemakkelijk veel meer pinnen tegelijk aansturen. Het is belangrijk om aan te geven welke de side-set pinnen zijn, met behulp van configuratie-opdrachten geschreven in C. De definitie is in C gedaan in **listing 2** met behulp van de functie `sm_config_set_sideset_pins()` in regel 5. Deze geeft aan dat de side-set pinnen beginnen bij GP2.



Listing 2

```

001 #define set_base 4 // pins start at GP4
002 #define set_count 2 // number of pins is 2
003 sm_config_set_set_pins (&c,set_base,set_count);
004
005 #define side_set_base 2 // side-pins start at GP2
006 sm_config_set_sideset_pins(&c, side_set_base);
007
008 float div = 125.0;
009 sm_config_set_clkdiv(&c, div);

```

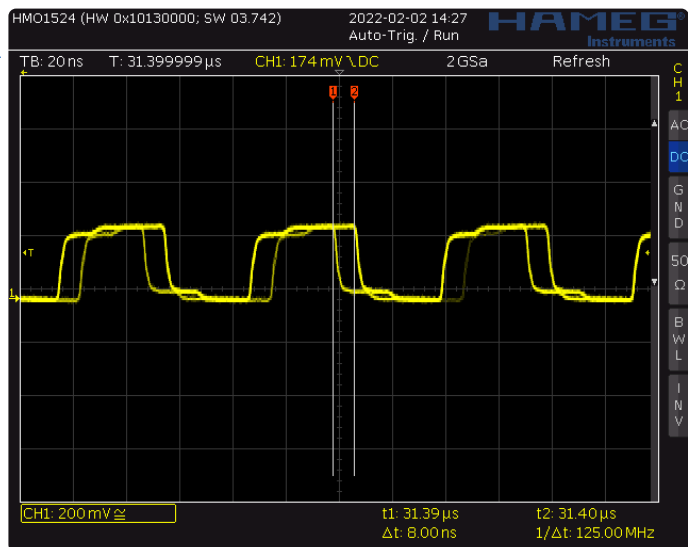
De `nop`-instructie in regel 4 van listing 1 genereert een opgaande flank via side-set op side pin 1 (= GP3). Regel 6 bevat een set-instructie die de pinnen 0 en 1 van de set-pins zet. De puls bereikt dan pin GP4. Side-pinnen 0 (GP2) en 1 (GP3) worden via side-set gereset. In de configuratieroutine in de tweede listing in regel 3 wordt bepaald dat de set-pinnen starten bij pin nummer GP4. De `set`-instructie kan worden gebruikt om registers of IO-pinnen te laden met constante waarden. De modifier `[2]` na de `set`-instructie op regel 8 van de eerste listing zorgt ervoor dat de instructie wordt gevolgd door een vertraging van twee klokcycli. Met deze specificatie tussen rechte haken kan elke instructie met 1 tot 31 klokcycli worden vertraagd.

Nu moet je alleen nog een geschikte klok gebruiken om ervoor te zorgen dat elke puls precies 1 μ s duurt. Daartoe moet je de systeemklok van 125 MHz delen door 125. Met de resulterende klok van 1 MHz duurt elke cyclus precies 1 μ s. Dit wordt gedefinieerd in de configuratieroutine van de tweede listing hierboven in regels 8 en 9.

De assembler-directives `.wrap_target` en `wrap` in listing 1 zorgen ervoor dat de instructies ertussen in runtime in een eindeloze lus worden herhaald. De sprong terug naar het begin van de lus (`.wrap_target`) kost geen tijd. Je kunt 'zero-cycle-overhead loops' programmeren. In ons programma duurt het zeven klokcycli (vijf instructies + twee cycli vertraging) om de lus eenmaal te doorlopen.

Genereren van de klok: fractionele deler

Zoals al opgemerkt, wordt de klok van de PIO-processor afgeleid van de 125MHz-systeemklok met behulp van een 'fractionele deler'. De 16bit-deler kan tot $2^{16} = 65.536$ tellen en heeft acht binaire cijfers na de decimale punt. De nauwkeurigheid is dus $1/256$. De 'fractionele deler' wordt op dezelfde manier gebruikt als in DDS-sigtaalgeneratoren. De fractie na de decimale punt wordt continu opgeteld en bij elke carry wordt een klokflank gegenereerd. **Listing 3** demonstreert het instellen van een fractionele kloksnelheid.



Figuur 5. Het gegenereerde kloksignaal vertoont jitter.



Listing 3

```
001 float clockFrequency=28e6;
002 float div=125e6/clockFrequency;
003 sm_config_set_clkdiv(&c,div);
```

Door de manier waarop de klok wordt gegenereerd, fluctueert de klokperiode rond een systeemklok-periode en vertoont de klok-golfvorm enige jitter. In het voorbeeld willen we een kloksnelheid van 28 MHz, maar aangezien 28 geen heeltallige deler van 125 is, is ook de deelfactor geen geheel getal, maar heeft die een rest. De klok-deelfactor is $125/28 = 4,4642857...$

De derde listing laat zien hoe deze deler wordt gebruikt voor de configuratie. Als demo wordt een eenvoudige blok-golf gegenereerd met het PIO-programma in **listing 4** hieronder.



Listing 4

```
001 .program theProgram
002 .side_set 1 opt
003 nop side 0
004 nop side 1
```

De oscilloscoop toont in **figuur 5** de klokjitter heel duidelijk. Bij hogere waarden van de klokdeler wordt de jitter proportioneel kleiner en legt minder gewicht in de schaal, zodat hij vaak kan worden verwaarloosd. Met de fractionele deler kunnen dan baudrates zoals de typische 115.200 bit/s met voldoende nauwkeurigheid worden gegenereerd.

Seriële gegevensoverdracht: een TX-UART

Het volgende voorbeeld is wat praktischer: het gaat om het serieel versturen van gegevens via een RS232-interface. De interface is aangesloten op pin GP4 zodat GP0 van de ingebouwde UART vrij blijft voor debug-doeleinden. Aangezien de interface inverterend is, moet je een geïnverteerd signaal genereren (idle state = hoog). Wanneer het ASCII-karakter wordt verzonden, ziet de pulsvorm op pin GP4 er dan zo uit als in **figuur 6**.

Nast het TXD-signaal moet de UART een 1 uitvoeren op een busy-lijn tijdens het verzenden van de acht databits. GP5 (side-pin 0) moet als busy-lijn worden gebruikt. Het bijbehorende PIO-programma staat in **listing 5** hieronder.



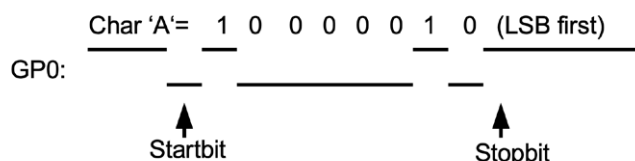
Listing 5

```
001 .program theProgram
002 .side_set 1 opt
003 pull [1]
004 set pins,0 side 1 [2] // startBit=0
005 set x,7 // loop for 7+1 times
006 bit:
007 out pins,1 [2] // LSB first
008 jmp x-- bit
009 set pins,1 side 0 [2] // stopBit=1
```

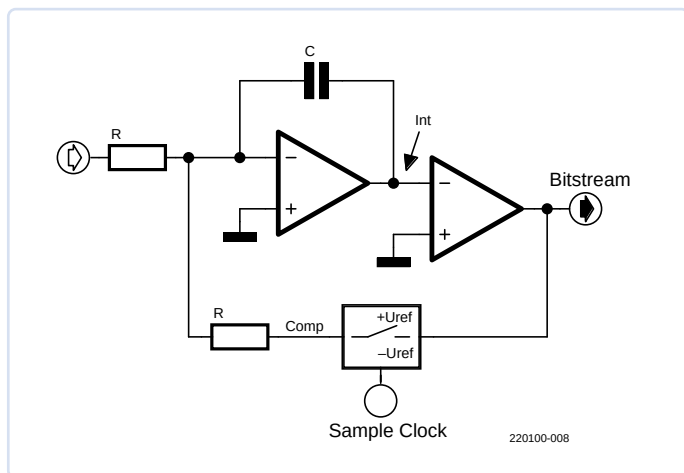
De **pull**-instructie in regel 3 haalt een 32bit-woord uit de TX FIFO op en slaat dat op in de OSR. Als er momenteel geen woord in de TX FIFO staat, wacht de instructie gewoon op een woord en voegt het dan rechtstreeks in de OSR in. De uitgangspen wordt door de set-instructie in regel 4 op 0 gezet om als startbit te fungeren.

De acht minst significante bits van het woord in de OSR worden nu in een lus naar buiten gevoerd. Hiervoor is het nodig de lus-teller te initialiseren. Het X-register dient hier als lus-teller. De **set**-instructie in regel 5 initialiseert deze met de waarde 7.

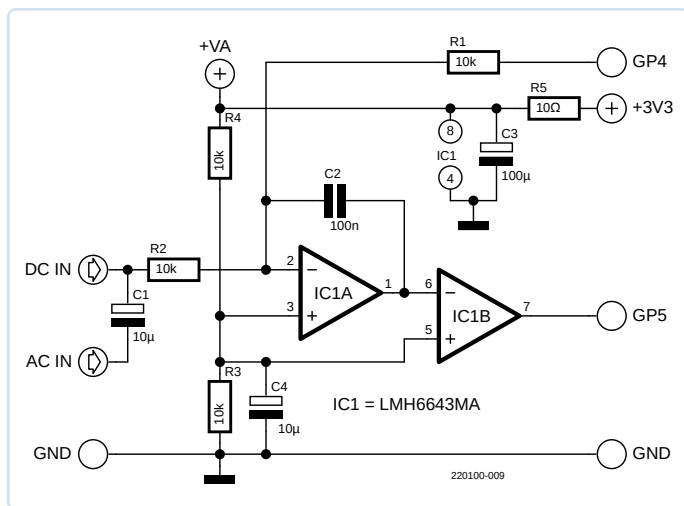
Het label **bit** in regel 6 markeert het begin van de lus. De eerste en enige instructie in de lus is **out** in regel 7. Deze voert het LSB van de OSR uit op out-pin 0 en verschuift de bits in de OSR één positie naar rechts. Naast de set-pinnen en de side set-pinnen vormen de out-pinnen de derde methode om de GPIO pins te gebruiken. Pin GP4 is ook ingesteld als out-pin.



Figuur 6. Pulsvolgorde om het teken 'A' te verzenden.



Figuur 7. Principeschema van een delta/sigma-ADC.



Figuur 8. Praktische uitvoering van de delta/sigma-ADC van figuur 7.

De voorwaardelijke sprong in regel 8 met de optie `x--` verlaagt het X-register met 1 en springt dan naar het label `bit` (het begin van de lus) wanneer `x` groter is dan 0. Op deze manier wordt de lus acht keer doorlopen. De laatste instructie van de TX-zendroutine is `set` in regel 9, die het stopbit maakt door set-pin 0 op 1 te zetten.

Bij de twee side-extensions wordt het Busy-sig-naal gegenereerd door side-pin 0 (GP5).

De vertragsings-extensions tussen rechte haken zorgen ervoor dat elk bit precies vier klokcycli lang is. De baudsnelheid is dus gelijk aan de kloksnelheid gedeeld door vier. Het instellen van `clkDiv` op `125.000.000 / (4 * 115.200)` resulteert dus in de gewenste baudrate van 115.200 bit/s.

De functie `pio_sm_put_blocking(...)` wordt gebruikt om karakters met een C-programma naar de PIO-UART te sturen, zoals de `PIOprint()` functie in **listing 6** demonstreert. Hier wordt een karakter doorgegeven aan de TX FIFO-unit. Als die vol is, blokkeert hij en wacht op meer ruimte.



Listing 6

```
001 void PIOprint(const char *s) {
002     while (*s)
003         pio_sm_put_blocking(pio, sm, *s++);
004 }
```

De TX UART-routine laat goed zien dat je met slechts zes instructies zinvolle functionaliteit kunt bereiken, waarbij de toevoeging van side set- en delay-instructies een belangrijke rol spelen.

Een delta/sigma-ADC

In deze paragraaf wordt een delta/sigma A/D-omzetter geïmplementeerd met behulp van een PIO. Het principe van een dergelijke omzetter is te zien in **figuur 7**. De schakeling probeert de ingangsspanning te compenseren met een schakelbare spanningsbron. Een integrator aan de ingang integreert het foutsignaal en een volgende comparator bepaalt het teken van de compensatiespanning gedurende het volgende tijdsinterval. Aan de uitgang ontstaat een bitreeks die het gemiddelde van de ingangsspanning volgt. **Figuur 8** toont een praktische schakeling volgens dit principe.

Deze schakeling wordt aangesloten op GP4 en GP5 van de Raspberry Pi Pico. De bitvolgorde wordt bepaald via PIO; acht bits worden samen als één byte opgeslagen in de RX FIFO-unit. De microcontroller haalt vervolgens de bytes uit deze unit op en verwerkt ze verder. Het bijbehorende PIO-programma is te zien in **listing 7**.



Listing 7

```
001 .program hello
002 Loop1:
003     set    x,7
004 inLoop:
005     in     pins,1
006     mov    osr,~isr
007     out    pins,1
008     jmp    x-- go1    [20]
009     push   noblock
010     jmp    Loop1      [23]
011 go1:
012     jmp    inLoop     [26]
```

De buitenste lus met het label `Loop1` wordt eindeloos herhaald. De binnenste lus maakt altijd acht iteraties. Elke iteratie begint met de instructie na het label `inLoop`. De `in`-instructie in regel 5 leest één bit van de ingangspin en schrijft die naar de ISR. In regel 6 wordt de ISR geïnverteerd gekopieerd naar de OSR. In regel 7 wordt het minst significante bit uitgevoerd naar de uitgangspin, waardoor de volgende compensatiestap in de integrator wordt gegenereerd.

Het gelezen bit is het volgende uitgangsbite van de deltasigma-omzetter. Acht bits worden verzameld in de ISR. De `jmp`-instructie in regel 8 bestuurt de binnenste lus. Wanneer deze 8 maal is doorlopen, wordt de `push`-instructie van regel 9 uitgevoerd. Deze instructie slaat de ISR op in de RX FIFO-unit die de MCU vervolgens kan legen. De delay-extensions tussen rechte haken zorgen ervoor dat elke delta/sigma-stap 50 klokcycli duurt. De klokdeeler is ingesteld op 25. De bemonsteringsfrequentie van de delta/sigma-omzetter is dus $125 \text{ MHz} / (25 \cdot 50) = 100 \text{ kHz}$.

Als demo hebben we een sinusgolf van 50 Hz omgezet met behulp van de delta/sigma-converter. **Figuur 9** toont het resulterende spectrum. De afstand tussen het bruikbare signaal en de ruis is ongeveer 60 dB. Helemaal niet slecht voor zo'n simpele schakeling!

Bouw een signaalgenerator

In dit voorbeeld bouwen we een eenvoudige R2R-DAC met behulp van een ladder netwerk volgens **figuur 10**. Dit wordt aangesloten op de I/O-pinnen van de Raspberry Pi Pico om analoge uitgangssignalen te produceren. De generator moet een continu en periodiek uitgangssignaal genereren. Hiertoe moeten gegevens herhaaldelijk vanuit een tabel in het geheugen naar de D/A-omzetter worden getransporteerd. Dit werkt bijzonder goed met behulp van DMA (*Direct Memory Access*). De DMA-controller van de Raspberry Pi Pico brengt de gegevens over naar de TX FIFO-unit. Het PIO-programma bestaat uit een enkele uitvoerbare instructie, zoals te zien in **listing 8**.

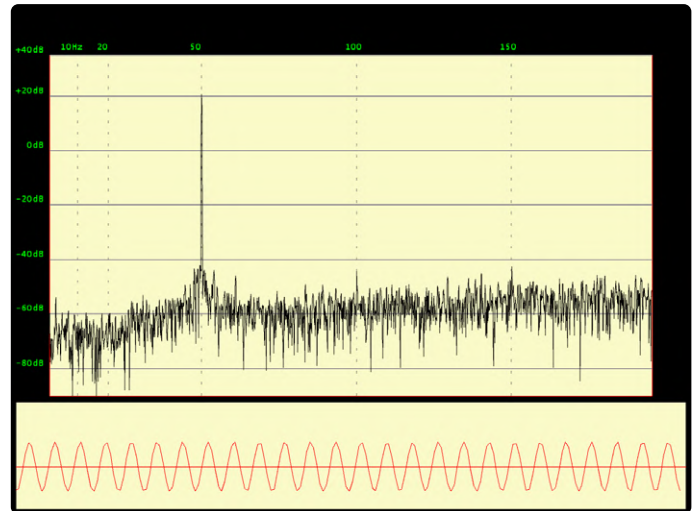


Listing 8

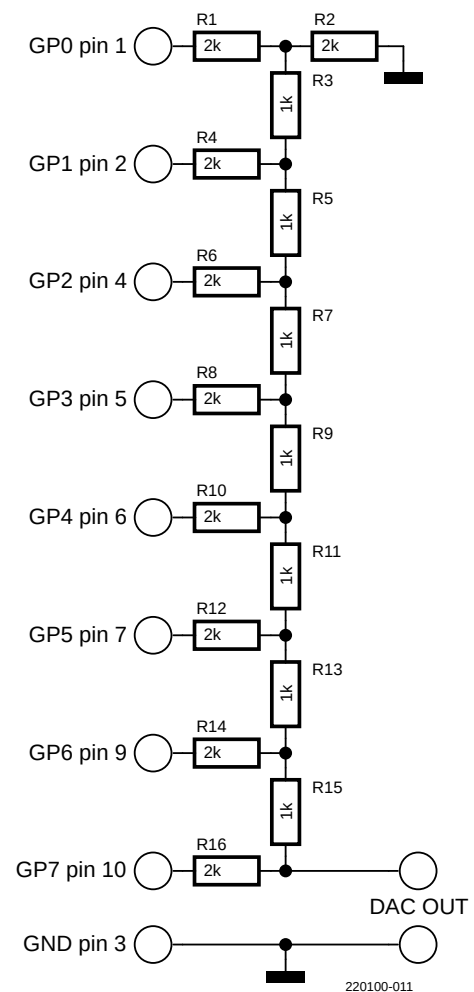
```
001 .program pio_serialiser
002
003 .wrap_target
004     out pins,8 // use autopull
005 .wrap
```

De `out`-instructie brengt de acht minst significante bits van de OSR naar de acht uitgangspinnen GP0...GP7. De OSR wordt dan automatisch opnieuw geladen vanuit de TX FIFO-unit omdat de *auto-pull* PIO-functie was geactiveerd in de configuratieroutine. De volgende acht bits worden dan uitgevoerd met de volgende instructie. De uitvoerlus is dus precies één PIO-klokcyclus lang. De gegevens moeten daarom zo snel mogelijk van het hoofdgeheugen naar de TX FIFO-unit worden getransporteerd. Daartoe zijn twee DMA-controllers van de RP2040 in serie geschakeld in de vorm van een data- en een besturings-DMA. Deze laatste zorgt ervoor dat de data-DMA-controller snel weer wordt opgestart. Met deze techniek kan elke vier systeemkyccli ($125 \text{ MHz} / 4$) een nieuwe byte worden uitgevoerd. De eenvoudige D/A-omzetter van **figuur 10** is echter niet geschikt voor een dergelijke hoge uitvoersnelheid.

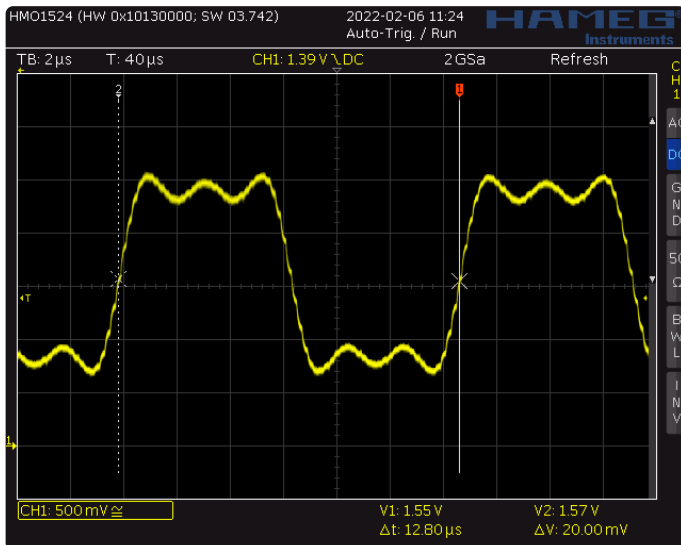
In **figuur 11** is `clkDiv` ingesteld op 25. De resulterende DAC-klok is $125 \text{ MHz} / 25 = 5 \text{ MHz}$. Aangezien één periode van het uitgangssignaal 64 samples lang is, wordt de gegenereerde frequentie berekend



Figuur 9. Het spectrum van een 50Hz-sinus na delta/sigma-conversie.



Figuur 10. R2R-DAC-laddernetwerk.



Figuur 11. De grondfrequentie plus twee harmonischen levert een blok golf op.

als $5 \text{ MHz} / 64 = 78,125 \text{ kHz}$. De Fourier-benadering van een blok golf die is opgebouwd uit de grondgolf plus twee harmonischen werd gebruikt om de uitgangs-golfvorm te produceren.

Registerinitialisatie

Het `set`-commando kan worden gebruikt om de registers X en Y met vaste waarden te initialiseren. Aangezien de commando's in 16 bit worden gecodeerd, kan het `set`-commando alleen constanten in een bereik van 0 tot 31 verwerken. Om X of Y met grotere constante waarden te initialiseren, kun je een truc gebruiken.

Door vanuit een C-programma de functie `pio_sm_exec(pio, sm, instruction)` aan te roepen, onderbreek je de instructievolgorde van het PIO-blok `pio` en de toestandsmachine `sm`, en wordt de opdracht `instruction` ingevoegd. De reeks opdrachten van **listing 9** kan worden gebruikt om register X te initialiseren met de waarde 500.000.



Listing 9

```
001 pio_sm_put_blocking(pio, sm, 500000);
002 pio_sm_exec(pio, sm,
    pio_encode_pull(false, false));
003 pio_sm_exec(pio, sm,
    pio_encode_mov(pio_x, pio_osr));
```

Eerst stop je de 500.000 in de TX FIFO-unit, en daarna laat je de `pull`-instructie los op de PIO-controller, die de 500.000 uit de FIFO-unit in de OSR overbrengt. Tenslotte zet de instructie `mov X,OSR` de waarde

zoals gewenst in het X-register. Nu kan de eigenlijke PIO worden gestart. Het programma van **listing 10**, dat een LED laat knipperen, dient als voorbeeld.



Listing 10

```
001 .program blink
002 .side_set 1 opt
003     mov y,x     side 1
004 yLoop1:
005     jmp y--     yLoop1
006     mov y,x     side 0
007 yLoop2:
008     jmp y--     yLoop2
```

MicroPython

De Raspberry Pi Pico is een platform dat zeer geschikt is voor de ontwikkeling van programma's die in MicroPython zijn geschreven. Er bestaat een overeenkomstige Python-interface voor PIO-programmering. Deze wordt niet geprogrammeerd met de PIO-assembler, maar met de juiste Python-syntaxis, die sterk doet denken aan een assembler-programmeertaal. Een voordeel van het gebruik van MicroPython is dat je complete projecten in één bestand kunt opslaan. Een voorbeeldprogramma is te vinden in de **MicroPython-listing**.

Het eigenlijke PIO-programma is gecodeerd als de Python-functie `pio_prog()`. Het bestaat uit de code in de lus tussen `wrap_target()` en `wrap()`. Het programma schakelt side-set pin 0 aan en uit – in dit geval de LED-pin GP25. De aan/uit-tijd wordt geregeld door een Y-wachtlus. De beginwaarde van het Y-register staat in het X-register. De waarde in het X-register wordt extern geregeld via de TX FIFO-unit. Dit wordt gedaan door de `pull(noblock)`-functie. Als er zich een woord in de TX FIFO-unit bevindt, wordt het naar het OSR-register verplaatst. De parameter `noblock` zorgt ervoor dat de inhoud van het X-register in het OSR wordt geplaatst wanneer de FIFO-unit leeg is en de opdrachtreeks wordt voortgezet.

Tot slot

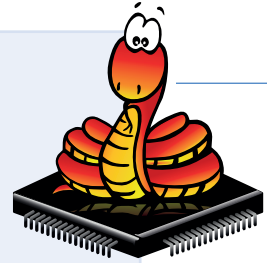
Hier sluiten we deze inleiding over de PIO-programmering van de RP2040-MCU af. We hebben laten zien dat krachtige interfaces kunnen worden geprogrammeerd met slechts een paar eenvoudige instructies. De PIO-programmeermogelijkheid van deze MCU's maakt ze veelzijdiger, zodat je misschien niet eens het gebruik van FPGA's hoeft te overwegen om de mogelijkheden van de processor uit te breiden. Dit artikel kan slechts een bescheiden overzicht geven van het programmeren van de PIO's. Er valt nog veel meer te ontdekken. Als je een goede praktijkgids wilt, bekijk dan onder andere [1][2][4][5][6][7], waar ook tal van andere voorbeelden in staan. ◀

220100-03



MicroPython-listing

```
001 from machine import Pin
002 from rp2 import PIO, StateMachine, asm_pio
003 from time import sleep
004
005 @asm_pio( sideset_init=(PIO.OUT_LOW))
006 def pio_prog():
007     wrap_target()
008     pull(noblock)
009     out(x, 32)
010
011     mov(y, x) .side(1)
012     label("Loop1")
013     jmp(y_dec, "Loop1")
014
015     mov(y, x) .side(0)
016     label("Loop2")
017     jmp(y_dec, "Loop2")
018     wrap()
019
020 class PIOAPP:
021     def __init__(self, sm_id, pinA, periodLength, count_freq):
022         self._sm = StateMachine(sm_id, pio_prog, freq=count_freq, sideset_base=Pin(pinA))
023         self._sm.active(1) # start PIO execution
024
025     def set(self, value): self._sm.put(value) # put val into TX-FIFO
026
027
028 pio1 = PIOAPP(0, pinA=25, periodLength=1, count_freq=1_000_000)
029 print("ready1")
030 while True:
031     pio1.set(200000) # 200 ms On/Off time
032     sleep(1) # for one second
033     pio1.set(100000) # 100 ms On/Off time
034     sleep(1) # for one second
035     print("idle")
036     print("idle")
```



Vragen of opmerkingen?

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



Gerelateerde producten

- > **Raspberry Pi Pico RP2040 (SKU 19562)**
www.elektor.nl/19562
- > **Dogan Ibrahim, Raspberry Pi Pico Essentials (SKU 19673)**
www.elektor.nl/19673



WEBLINKS

- [1] Aan de slag met Raspberry Pi Pico: <https://datasheets.raspberrypi.com/pico/getting-started-with-pico.pdf>
- [2] Handleiding voor de setup van de toolchain door Shawn Hymel: <https://tinyurl.com/yde5dd8a>
- [3] Software-download: <http://www.elektormagazine.nl/220100-03>
- [4] Datasheet van de RP2040: <https://datasheets.raspberrypi.com/rp2040/rp2040-datasheet.pdf>
- [5] Raspberry Pi Pico SDK: <https://datasheets.raspberrypi.com/pico/raspberry-pi-pico-c-sdk.pdf>
- [6] SDK-documentatie: <https://raspberrypi.github.io/pico-sdk-doxygen/index.html>
- [7] RP2040 MicroPython: <https://docs.micropython.org/en/latest/library/rp2.html>



Armeluis chiptweaker

we krijgen je wel (goedkoop) aan de praat...

By Luka Matic (Croatia)

Hardware-beveiligingsmodules (hardware security modules, HSM's) zijn elektronische apparaten die gebruik maken van verschillende hardware-beveiligingen tegen het ongeoorloofd lezen van de geheime gegevens die ze bevatten. Zij voeren gewoonlijk handelingen uit zoals versleuteling of digitale ondertekening, maar kunnen ook andere taken uitvoeren. Bankkaarten of toegangskarten zijn typische voorbeelden van HSM's. Deze moeten pincodes en privésleutels zo lang mogelijk geheim houden voor het geval de kaart wordt gestolen of verloren gaat. Bovendien hebben veel MCU's voor algemeen gebruik ook geheugenbeschermingsfuncties om het illegaal kopiëren van de eigen firmware te voorkomen. De hier gepresenteerde armeluis chiptweaker is een tool om de beginselen van niet-invasieve aanvallen op HSM's te leren kennen. Hiermee kun je proberen een microcontroller met geheugenbeveiliging te ontgrendelen.

Zoals uitgelegd in een vorig artikel [1], kunnen onze arme spionnen Alice en Bob er niet op hopen hun eigen HSM's te bouwen, omdat daarvoor dure gespecialiseerde apparatuur en kennis vereist zijn waarover ze waarschijnlijk niet beschikken. Voor hun geheime operaties gebruiken ze hun eigen hardware, gebouwd met goedkope en gemakkelijk verkrijgbare componenten waarop ze kunnen vertrouwen. Dit werkt goed voor hen, zolang de juiste Operations Security-procedures (OpSec) worden gevolgd. Aan de andere kant kunnen ze ook niet om het gebruik van HSM's zoals bankkaarten heen, en ze kunnen op een HSM stuiten waarvan ze de geheimen te weten willen komen. De armeluis chiptweaker (Poor

Man's ChipTweaker, PMCT) zal daarom heel goed van pas komen, omdat het hen in staat stelt de principes van niet-invasieve aanvallen op HSM's te leren.

Hardware-aanvallen

Bij aanvallen op HSM's kunnen we drie typen onderscheiden:

1. **Niet-invasief:** de HSM wordt niet geopend of uit elkaar gehaald. De aanval wordt uitgevoerd met behulp van verschillende elektrische signalen op de gewone communicatie- en voedingsaansluitingen van de HSM. Hiervoor is de PMCT ontworpen. Dit is een goedkope manier.

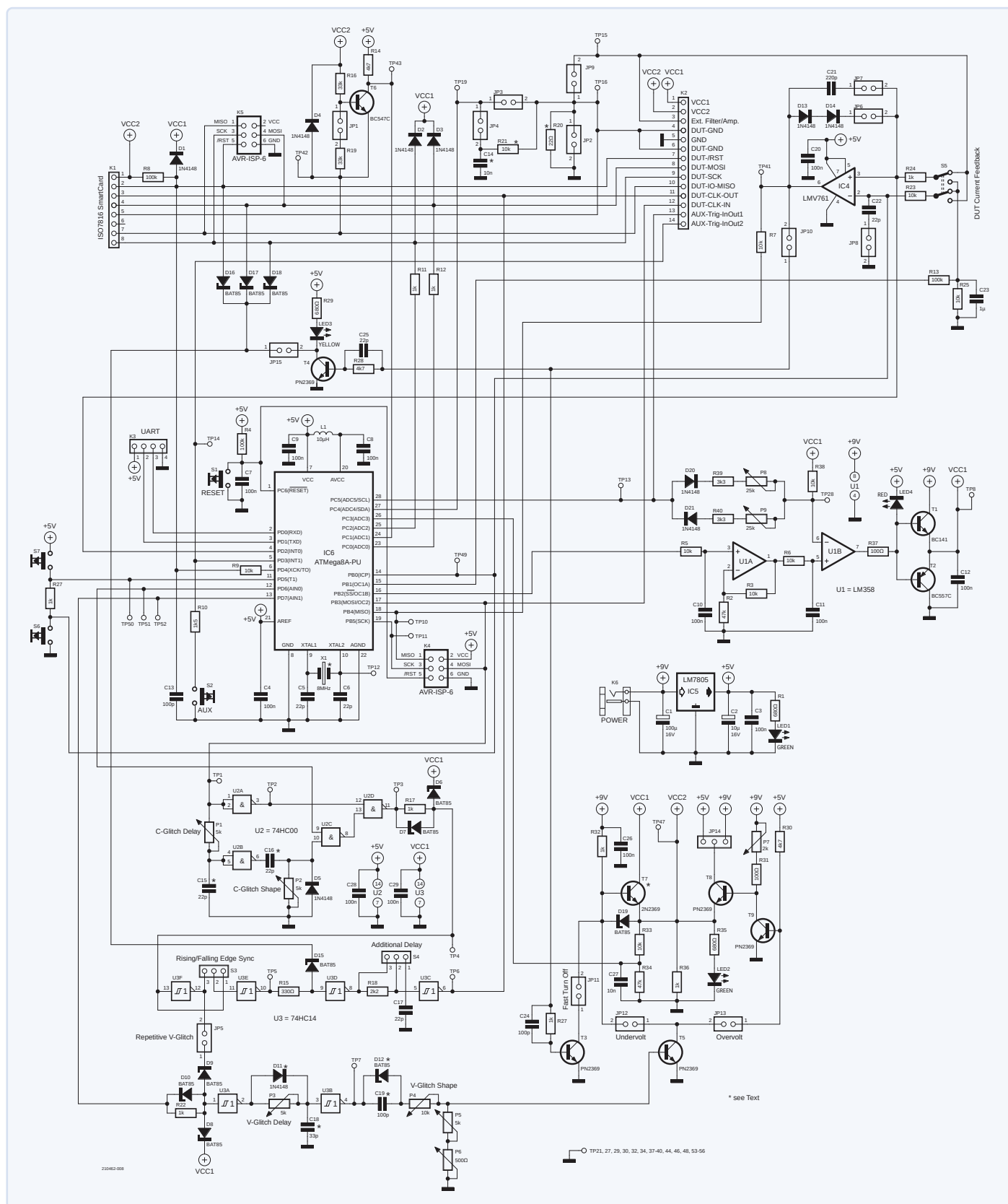
2. **Semi-invasief:** de afdekking van het halfgeleiderkristal van de HSM wordt verwijderd. De elektrische contacten op de silicium-*die* worden niet blootgelegd. De aanval wordt uitgevoerd door bepaalde delen van de HSM-microschakelingen aan lichtpulsens bloot te stellen. Deze manier is al duurder.
3. **Invasief:** het halfgeleiderkristal ligt volledig bloot, zodat microprobes kunnen worden aangesloten op de contacten op de *die*. De aanval wordt uitgevoerd door elektrische signalen in de HSM-microschakelingen te injecteren. Dit is een kostbare aanval waarvoor zeer gespecialiseerde apparatuur nodig is.

Er zijn natuurlijk krachtige digitale *ChipWhisperers* verkrijgbaar [2]. Maar wat ik wilde doen was iets maken op basis van oudere technologie, vergelijkbaar met wat dr. Skorobogatov heeft gedaan [3], waarbij alle analoge en digitale signalen volledig toegankelijk zijn en wat wellicht een betere manier is om de principes van niet-invasieve aanvallen te begrijpen.

Specificaties

Niet-invasieve aanvallen zijn gebaseerd op verschillende foutinjectiemethoden, zoals het verzenden van slecht geformatteerde gegevens of het verzenden van signalen met een onjuiste amplitude en frequentie naar een van de pinnen van de HSM (inclusief de voedingspinnen). Deze kunnen de DUT (*Device under Test*, of beter *Device under Attack*) vele ongecontroleerde acties laten uitvoeren en hopelijk zijn geheimen doen prijsgeven.

Niet-invasieve aanvallen vereisen veel gegevensverwerking, tweakken en jagen in het donker voor elke specifieke DUT, maar als het mogelijk is om hem niet-invasief te kraken, kan de goedkope en eenvoudige PMCT je daarbij helpen. Hij heeft de volgende mogelijkheden:



Figuur 1. Het schema van de Poor Man's ChipTweaker. Let op het gebruik van snelle transistoren van het type 2369. Om redenen van warmteafvoer moet T7 een TO-18 type zijn (metalen behuizing, 2N2369). De plastic PN2396-types die voor de andere worden gebruikt zijn wellicht goedkoper en/of gemakkelijker te vinden. Andere met een * aangegeven onderdelen moeten in een busstrip worden gemonteerd om gemakkelijk te kunnen experimenteren met verschillende waarden.

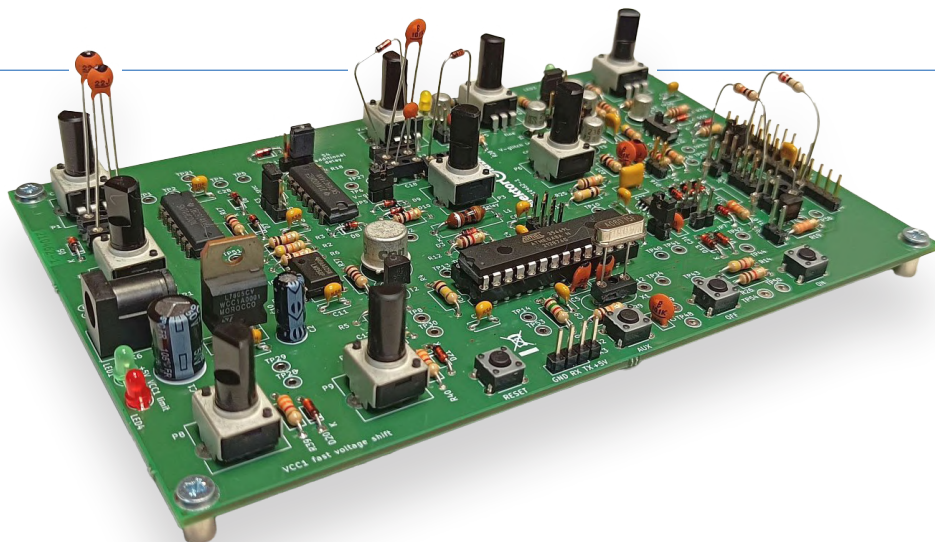
1. **Klokgitch-generator:** het injecteren van te snelle signalen op de klokingang (CLK) van de HSM kan ervoor zorgen dat de CPU een machine-instructie overslaat of verkeerd uitvoert, en dus bijvoorbeeld een PIN-controle overslaat.
2. **Spanningsglitch-generator:** het wijzigen van de voedingsspanning van de HSM tot buiten het nominale bereik kan ook een ongecontroleerde actie veroorzaken. De PMCT kan langzame, middellange en snelle spanningsglitches leveren, zowel onder-spanning als overspanning.
3. **Variabele 1,0...6,0 V DUT-voeding:** gestuurd door de microcontroller van de PMCT.
4. **Snelle DUT-turn-off comparator:** boven of onder een bepaalde voedingsstroomdrempel (voor bijvoorbeeld een EEPROM-schrijfsabotage-aanval), armed-disarmed of direct getriggerd door de hoofd-MCU.
5. **Bit-banged SPI en bidirectionele een-pin UART-interface** voor communicatie met de DUT (inclusief alle typen SmartCards), met bidirectionele level shifters in het bereik van 1,5 V tot 6,0 V.

Naast bovengenoemde aanvallen kan de PMCT in combinatie met een digitale 'scoop (zoals een LabNation SmartScope), een 100...200 MHz analoge geheugenscoop (zoals de Tektronix 466) en een PC met software als MATLAB voor offline gegevensanalyse allerlei soorten timing- en voedingsanalyses uitvoeren.

In tegenstelling tot de volledig digitale ChipWhisperers zijn alle analoge en digitale signalen toegankelijk voor studie en analyse, en kan de glitch-timing nog nauwkeuriger worden afgesteld dankzij de continue analoge afstelling met potentiometers zonder kwantiseringstapjes. Daarom kan een trage ATmega8 het hele aanvalsproces coördineren, en is er geen FPGA nodig. Glitches met een duur van minder dan 10 ns kunnen worden gegenereerd; deze veroorzaken fouten in CPU's die zijn ontworpen voor frequenties tot 50 MHz en hoger. Veel SmartCards en MCU's werken slechts tot maximaal 20 MHz.

Beschrijving van de hardware

In het midden van het schema (figuur 1) zien we de microcontroller-eenheid (MCU, IC6, ATmega8) die de aanvalsprocessen coördineert. De schakeling rond IC1 is een variabele spanningsbron voor de DUT. Zijn uitgangs-



Figuur 2. De Poor Man's ChipTweaker is opgebouwd op een comfortabele print. De in busstrips gemonteerde onderdelen zijn in het schema aangegeven met '*'.

spanning VCC1 wordt geregeld via PWM door timer-uitgang OC1B van IC6. Hierdoor kunnen langzame spanningssschommelingen worden opgewekt. MCU-pen PC5 kan een van drie toestanden hebben – L, H en high-Z – waardoor hij de versterking van IC1B snel kan veranderen en dus drie verschillende waarden van VCC1 kan produceren, ingesteld door P8 en P9. Dit is de manier om een doorsnee spanningsglitch (orde van grootte van μ s) te genereren. De rode LED4 gaat branden als waarschuwing en beperkt VCC1 tot ongeveer 6 V, wat niet te hoog is voor een 5V-apparaat.

Glitches genereren

De DUT wordt normaal gesproken gevoed door VCC2 via transistor T7 (een 2N2369 in een TO-18 behuizing). T5 kan diens basis snel laag trekken (plaats jumper JP12), wat resulteert in een snelle onderspanning-glitch (in de orde van grootte van 10 ns). Als JP13 wordt geplaatst, zal T8 VCC2 snel hoog trekken, waardoor een snelle overspanning-glitch ontstaat. Snelle spanningsgolven worden gegenereerd door C19, P4, P5 en P6. De vertraging van de spanningsglitch ten opzichte van de CLK-puls wordt ingesteld met P3. Als MCU-uitgang PD7 hoog wordt gemaakt, wordt een enkele spanningsglitch afgegeven. Als deze hoog blijft, en JP5 wordt kortgesloten, worden bij elke CLK-puls spanningsglitches afgevuurd.

IC2 wordt gebruikt om de DUT op een klokgitch te trakteren. De PWM-uitgang OC2 van de MCU genereert de CLK-puls voor de DUT. Als poort PD6 hoog is, geeft IC2C bij elke CLK-puls een korte glitch door aan IC2D. De polariteit van de glitch kan worden gekozen met een jumper of schakelaar op S3.

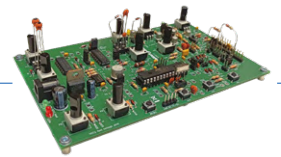
IC4 is een snelle comparator met positieve terugkoppeling (versterkt door D13, D14 en C21) die de DUT in een paar ns kan uitschakelen. Dit kan ook worden bereikt met een commando van de MCU (PD5 of PD2 om in te schakelen, PB0 om de DUT uit te schakelen) of door op de knoppen S6 en S7 te drukken.

Aanval via het stroomverbruik

IC4 kan ook worden geconfigureerd om de DUT uit te schakelen als het stroomverbruik van de DUT (gemeten over weerstand R20) hoger of lager (geselecteerd met S5) is dan een drempelwaarde (bij C23, gemeten via PWM op de OC1A pen). Dit wordt typisch gebruikt om te voorkomen dat de DUT naar zijn interne EEPROM schrijft (wat extra stroom verbruikt). De uitgang van IC4 activeert T3, die T7 uitschakelt. Hij activeert ook transistor T4 (auxiliary turn-off) om alle SmartCard-lijnen laag te trekken en mogelijke fantoom-terugvoeding te voorkomen.

Analoge ingang ADC4 bewaakt de stroom van de DUT. Aangezien dit een langzame ADC is, kan het signaal worden gefilterd door R21/C14, of het kan eventueel nog worden verwerkt door een sneller extern filter/versterker (op connector K2). Voor een timingaanval of vermogensanalyse wordt dit signaal meestal opgenomen door een oscilloscoop voor offline verwerking en analyse met een programma als MATLAB.

Aangezien de MCU op 5 V werkt terwijl de DUT op de (meestal) lagere spanning VCC2 werkt, is de bidirectionele level shifter (rond T6) nodig voor de I/O-pin 7 van de DUT op connector K1 (een bidirectionele UART-poort, zoals de meeste SmartCards tegenwoordig gebruiken). Pinnen 4 en 8 worden gebruikt



voor SmartCards met SPI-interface (bijvoorbeeld de FunCard). Omdat ze unidirectioneel zijn, zijn hun level shifters eenvoudiger (diodes D2 en D3).

De print

Om de bouw van je eigen Poor Man's ChipTweaker te vergemakkelijken is er een print ter grootte van een eurokaart voor ontworpen (**figuur 2**). (Zie [4] voor het KiCad-project en de Gerber-bestanden.) Alle onderdelen zijn van het through-hole type, behalve IC4, dat een SOIC-8 behuizing heeft. Het volbouwen van de print zou geen problemen mogen opleveren, maar er zijn enkele zaken waar je op moet letten:

- > In plaats van de 3-polige jumpers 'Sxx' zijn SPDT schuif- of wipschakelaars wellicht handiger.
- > Misschien gebruik je liever liever instelpotmeters dan gewone potmeters. Trimmers zijn gemakkelijker verkrijgbaar in meer waarden.
- > T7 moet een 2N2369 zijn in een metalen TO-18 behuizing in verband met de warmteafvoer. De PN2396-transistoren in TO-92 behuizing kunnen worden vervangen door 2N2369-types, afhankelijk van wat je te pakken kunt krijgen.
- > Onderdelen met een sterretje (*) (zoals D11 en C18, enzovoort, met uitzondering van T7) worden verondersteld in een busstrip te worden gemonteerd om gemakkelijk te kunnen experimenteren met verschillende waarden en types.

Het board heeft veel testpunten, altijd met een massaverbinding in de buurt. Probeer de nummering ervan niet te begrijpen; daar ontbreekt elke logica.

Aanvallen!

De DUT voor onze demo-toepassingen is een standaard FunCard (met een Microchip AVR AT90S8515 MCU plus een AT24C EEPROM). Zonder serieuze beveiligingen is een aanval relatief eenvoudig. Laad de FunCard met de firmware van [4] via ISP-connector K5. Gebruik vervolgens een van de twee ATmega8 firmware-varianten (voor het aanvallen van een FunCard, ook bij [4]) om de MCU IC6 van de ChipTweaker te programmeren via ISP-connector K4 (en niet K5). Je kunt nu wat basisprocedures uitproberen. Sluit een seriële terminal aan op de

UART-poort K3 en configureer deze voor 9600-8-N-1 (of 9600n81 zo je wilt). Nadat je op de reset-knop S1 hebt gedrukt, zou je een scherm moeten krijgen zoals in **figuur 3**. De opties 0 en 1 zijn voor het versturen van eenvoudige commando's naar de MCU en de FunCard. Alle commando's zijn 4 bytes lang, waarbij de laatste byte altijd gelijk is aan 0x0d (CR). Zie **tabel 1** en **tabel 2**.

```
> 0 : Send a single command to Smartcard
> 1 : Send a single command to MCU
> 2 : Find minimum Vcc2
> 3 : Adjust minimum Vt
> 4 : Funcard no glitch demo
> 5 : Funcard clock glitch demo
> 6 : Funcard volt glitch demo
> 7 : Funcard volt glitch demo
> 8 : Volt glitch loop test
> 9 : Volt & Clock glitch loop test
> A : Fast Vcc2 test
> B : Brute-force PIN demo, Funcard MCU internal counter
> b : Brute-force PIN demo, Funcard AT24C external counter
> R : Reset MCU & Smartcard

Vcc2 RAW set to:0xDD Vcc2 feedback: 5,092V
Vt RAW set to:0xFF Vt scaled set to: 0,452V Vi feedback: 0,166V
Smartcard fclk division factor set to RAW:0x00

> Select Option (0-X): 5

No glitch demo.

Answer To Reset: 0x45 0x67 0x78 0x9A 0xBC 0xDE 0xF0 0x12 0x34 0x56 0x78 0xDE 0xAD 0x01 0x23 0x00 0x00 0x00 0x00 0x00
```

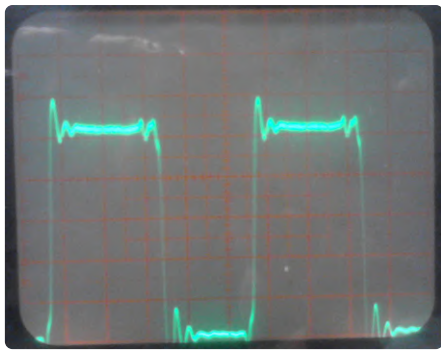
Figuur 3. Selecteer de aanval die je wilt uitvoeren uit een menu.

Commando	Beschrijving
V2x	'x' is een 8bit-getal dat de PWM-verhouding voor OC1B bepaalt om de VCC2-spanning in te stellen.
Vtx	'x' is een 8bit-getal dat de PWM-verhouding voor OC1A bepaalt om de Vt-spanning in te stellen, een drempel voor C23 om IC4 te activeren.
Fxy	'x' is een ruwe frequentie-deelfactor, voor het genereren van een CLK-puls voor de DUT op OC2. Indien ingesteld op 0, loopt de OC2 op de helft van de MCU-kristalfrequentie; 'y' wordt genegeerd.
GLx	De laagste 3 bits van 'x' stellen de MCU-pinnen PD7, PD6 & PC5 in; voor testdoeleinden.
onx	Schakel de DUT in; 'x' wordt genegeerd.
ofx	Schakel de DUT uit; 'x' wordt genegeerd.
Sxy	Sla de huidige instellingen op in de EEPROM van de MCU; 'x' en 'y' worden genegeerd.

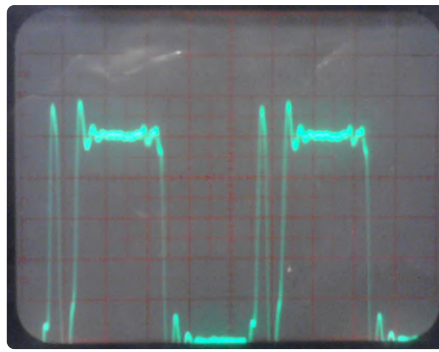
Tabel 1. Commando's voor de MCU. Elk commando wordt met 0x0d (CR) afgesloten.

Commando	Beschrijving
Rxy	Lees een byte van adres 'x' van de interne EEPROM van de AT90S; 'y' wordt genegeerd.
Wxy	Schrijf byte 'y' naar adres 'x' van de interne EEPROM van de AT90S.
Pxy	Test een PIN-code 'xy' (in BCD-formaat). De PIN wordt opgeslagen in de interne EEPROM van de AT90S.
rxxy	Lees een byte van adres 'x' van de AT24C-EEPROM; 'y' wordt genegeerd.
wxy	Schrijf byte 'y' naar adres 'x' van de AT24C-EEPROM.
pxy	Test een PIN-code 'xy' (in BCD-formaat). De PIN is opgeslagen in AT24C-EEPROM.

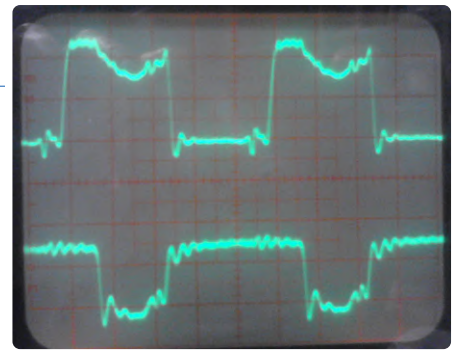
Tabel 2. Commando's voor de SmartCard. Elk commando wordt met 0x0d (CR) afgesloten.



Figuur 4. Een schoon kloksignaal zonder glitches. De oscilloscoop was ingesteld op 50 ns/div op de horizontale as en 1 V/div op de verticale as.



Figuur 5. Een kloksignaal met glitches (50 ns/div horizontaal, 1 V/div verticaal). Regel potentiometers P1 en P2 af om een glitch te maken zoals hier getoond.



Figuur 7. De spanning-glitch komt 30 tot 40 ns na de opgaande flank van de klokpuls. Het bovenste spoor is het kloksignaal, het onderste is de voedingsspanning VCC2 (50 ns/div horizontaal; 1 V/div verticaal).

in verschillende stadia van de berekeningen levert verschillende fouten op. Misschien is het resultaat van de wiskundige bewerking verkeerd, of wordt er gewoon een machine-instructie overgeslagen.

Zie **figuur 4**, **figuur 5** en **figuur 6** voor de signalen en de resultaten. Een effectieve glitch voor de FunCard is een actief lage puls met een duur tussen 10 ns en 20 ns die ongeveer 10...20 ns na de opgaande flank van de CPU-klok wordt afgevuurd (zie figuur 5).

Demo: spanning-glitch aanval

Dit is een demo van een eenvoudige aanval met een spanning-glitch. De FunCard voert dezelfde bewerkingen uit als in het vorige voorbeeld. Een effectieve onderspanning-glitch heeft een duur van minstens 50 ns en levert goede resultaten op met de timing

van figuur 7. Hier begon de glitch 30...40 ns na de opgaande flank van de klokpuls. De voedingsspanning voor de FunCard was ingesteld op 2,24 V. De spanningsdip was ongeveer 1,5 V diep.

Stel de vorm van de spanningstoename in met P4, P5 en P6. Gebruik P3, S3 en S4 om de glitchvertraging en -synchronisatie in te stellen. De resultaten worden op dezelfde manier weergegeven als in het vorige voorbeeld (**figuur 8**).

De geheugenbeveiliging kraken om de firmware te extraheren

Elke beschermde MCU of SmartCard vereist een andere aanvalsprocedure. Het gaat gepaard met veel vallen-en-opstaan en jagen in het donker om mogelijke zwakke plekken te vinden. De procedure die werkt op veel Micro-

chip AVR-microcontrollers is een langzame onderspanning-glitch.

Dat gaat als volgt. Wanneer de FunCard MCU (een AT90S8515) beschermd is, worden beide memory lock-bits op 0 geprogrammeerd. Het flash-geheugen kan dan niet worden gelezen, alleen gewist. In seriële programmeermodus (de reset-ingang naar 0 V getrokken) zal het **Chip Erase**-commando eerst het flash-geheugen wissen (alle flash-bytes resetten naar 0xFF) en dan de lock-bits wissen (uitschakelen door ze op 1 te zetten).

De ontwerpfout die in veel AVR-MCU's (niet alle) aanwezig is, stelt ons in staat de beveiliging op de volgende manier te omzeilen. Als de voedingsspanning wordt verlaagd tot onder het nominale minimum (2,7 V), en wel tot 1,6...1,7 V, is het niet meer mogelijk om het flash-geheugen te wissen, maar het wissen van de lock-bits lukt nog wel. De aanval wordt gestart bij 1,1 V en de spanning wordt geleidelijk opgevoerd. De lock-bits werden met succes verwijderd bij 1,62 V zonder het flash-geheugen te wissen!

Wanneer je probeert de firmware of de apparaathandtekening te lezen van een beschermde AVR-microcontroller, zal het antwoord "0x00, 0x01, 0x02, 0x03" zijn en dan weet je dat het geheugen beschermd is (**figuur 9**). Zodra je de juiste handtekening krijgt ("0x1E, 0x93, 0x01" voor de AT90S8515) zijn de memory lock-bits gewist en kan het programmeergeheugen worden gelezen met elke ISP-programmer.

Voedingsanalyse bij een bankkaart

Bankkaarten worden steeds geavanceerder en gebruiken meerdere methoden om kritieke geheugengebieden fysiek te beschermen. Ze proberen ook de voedingsstroom te maskeren, zodat kritieke handelingen niet gemakkelijk kunnen worden achterhaald. Als het tijdstip van bepaalde procedures, zoals het controleren van de ingevoerde PIN-code, precies kan

```
> 0 : Send a single command to Smartcard
> 1 : Send a single command to MCU
> 2 : Find minimum Vcc2
> 3 : Adjust minimum Vt
> 5 : Funcard no glitch demo
> 6 : Funcard clock glitch demo
> 7 : Funcard volt glitch demo
> 8 : Volt glitch loop test
> 9 : Volt & Clock glitch loop test
> A : Fast Vcc2 test
> B : Brute-force PIN demo, Funcard MCU internal counter
> b : Brute-force PIN demo, Funcard AT24C external counter
> R : Reset MCU & Smartcard

Vcc2 RAW set to:0xDD Vcc2 feedback: 5,092V
Vt RAW set to:0xFF Vt scaled set to: 0,452V
Smartcard fclk division factor set to:RAW:0x00

> Select Option (0-X): 6

Clock glitch demo.

Answer To Reset: 0x45 0x67 0x78 0x9A 0xBC 0xDE 0xF0 0x12 0x34 0x56 0x78 0xDE 0xAD 0x01 0x23 0x00 0x00 0x00 0x00
0xF62F 0x01 0x25 0x25 OK!
0xF34B 0x02 0x3E 0x3E OK!
0xF62F 0x03 0x25 0x25 OK!
0xF62F 0x04 0x25 0x25 OK!
0xF5E4 0x05 0xD9 0xD9 OK!
0xF62F 0x06 0x25 0x25 OK!
0xF62F 0x07 0x25 0x25 OK!
0xF576 0x08 0x6B 0x6B OK!
0x48E8 0x09 0x30 0x30 OK!
0x47E8 0x0A 0x2F 0x2F OK!
0xF576 0x0B 0x6C 0x6C OK!
0x47E8 0x0C 0x2F 0x2F OK!
0xF62F 0x0D 0x25 0x25 OK!
0x0000 0x0E 0x00 0x00 OK!
0x5555 0x0F 0x55 0x55 Error!
0xF62F 0x10 0x25 0x25 OK!
0xF62F 0x11 0x25 0x25 OK!
0xF62F 0x12 0x25 0x25 OK!
0xF62F 0x13 0x25 0x25 OK!
```

Figuur 6. Terminalvenster met de resultaten van de klok-glitch aanval. **Kolom 1** is het 16bit-resultaat van de wiskundige bewerking zoals uitgevoerd door de DUT. Het is fout als de glitch succesvol was. **Kolom 2**: glitch-vertraging. Glitches op verschillende tijdstippen geven verschillende resultaten. **Kolom 3**: checksum als de som van de twee bytes van het 16bit-resultaat zoals berekend door de FunCard. **Kolom 4**: checksum berekend door de MCU van de PMCT. **Kolom 5**: **'Fout!'** als kolommen 3 en 4 niet overeenkomen.


```

> 0 : Send a single command to Smartcard
> 1 : Send a single command to MCU
> 2 : Find minimum Vcc2
> 3 : Adjust minimum Vt
> 5 : Funcard no glitch demo
> 6 : Funcard clock glitch demo
> 7 : Funcard volt glitch demo
> 8 : Volt glitch loop test
> 9 : Volt & Clock glitch loop test
> A : Fast Vcc2 test
> B : Brute-force PIN demo, Funcard MCU internal counter
> b : Brute-force PIN demo, Funcard AT24C external counter
> R : Reset MCU & Smartcard

Vcc2 RAW set to:0x60 Vcc2 feedback: 2,288V
Vt RAW set to:0xFF Vt scaled set to: 0,462V Vi feedback: 0,053V
Smartcard fclk division factor set to RAW:0x00

> Select Option (0-X): 7

Volt glitch demo.

Answer To Reset: 0x45 0x67 0x78 0x9A 0xBC 0xDE 0xF0 0x12 0x34 0x56 0x78 0xDE 0xAD 0x01 0x23 0x00 0x00 0x00 0x00 0x00
0xF62F 0x01 0x25 0x25 OK!
0xF62F 0x02 0x25 0x25 OK!
0xF62F 0x03 0x25 0x25 OK!
0xF62F 0x04 0x25 0x25 OK!
0x3F4B 0x05 0x8A 0x8A OK!
0xF62F 0x06 0x25 0x25 OK!
0xF62F 0x07 0x25 0x25 OK!
0x482F 0x08 0x77 0x77 OK!
0x492F 0x09 0x78 0x78 OK!
0x482F 0x0A 0x77 0x77 OK!
0xF676 0x0B 0x6C 0x6C OK!
0x482F 0x0C 0x77 0x77 OK!
0xF62F 0x0D 0x25 0x25 OK!
0x0000 0x0E 0x00 0x00 OK!
0xF642 0x0F 0x38 0x38 OK!
0x0000 0x10 0x00 0x00 OK!
0xF62F 0x11 0x2F 0x25 Error!
0xF62F 0x12 0x25 0x25 OK!
0xF62F 0x13 0x25 0x25 OK!

```

Figuur 8. Terminalvenster met de resultaten van de spanning-glitch aanval. De kolommen zijn op dezelfde wijze opgemaakt als voor de klok-glitch aanval (zie figuur 6).

```

CLEAR [x] AutoScroll [x] Reset Cnt [13] Cnt = 1721 HEX ASCII StartLog StopLog Req
Stage:0x01 Vcc2 RAW:0x43 Init:0x53
Reading firmware:0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09
Reading signature... 0x00 0x01 0x02 0x03
Y-erase the lockbits, N-read again, F-finish?
Vcc2 feedback: 1,575V

Stage:0x01 Vcc2 RAW:0x44 Init:0x53
Reading firmwar03 03 x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09
Reading signature... 0x00 0x01 0x02 0x
Y-erase the lockbits, N-read again, F-finish?
Vcc2 feedback: 1,598V

Stage:0x01 Vcc2 RAW:0x45 Init:0x53
Reading firmware:0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09
Reading signature... 0x00 0x01 0x02 0x03
Y-erase the lockbits, N-read again, F-finish?
Vcc2 feedback: 1,622V

Stage:0x01 Vcc2 RAW:0x46 Init:0x53
Reading firmware:0x0C 0x1C 0x1A 0x19 0x18 0x17 0x22 0x2C 0x14 0x13
Reading signature... 0x1E 0x93 0x01 0xFF
Y-erase the lockbits, N-read again, F-finish?

Stage:0x01 Vcc2 RAW:0x47 Init:0x53
Reading firmware:0x0C 0x1C 0x1A 0x19 0x18 0x17 0x22 0x2C 0x14 0x13
Reading signature... 0x1E 0x93 0x01 0xFF
Y-erase the lockbits, N-read again, F-finish?

Transmit

```

Figuur 9. Het kraken van de beveiliging van het flash-geheugen.



Figuur 10. Voedingsstroom (geel spoor) van een bankkaart bij controle van een onjuiste PIN-code.



worden vastgesteld, dan weten we wanneer glitches kunnen worden afgevuurd om te proberen de beveiligingen te omzeilen. Enkele van de bekende beveiligingsmethoden bij bankkaarten zijn:

1. Gebruik van een interne RC-klok voor veiligheidskritische bewerkingen, en overschakeling op externe CLK alleen voor nauwkeurige timing, bijvoorbeeld bij communicatie via de UART. Dit voorkomt klok-glitch aanvallen.
2. Het gebruik van zeer snelle ladingspompen op kleine interne (pF-bereik) voedingscondensatoren om een stabiele voedingspanning te handhaven. Dit voorkomt spanningsglitch-aanvallen.
3. Het willekeurig veranderen van de halve perioden van de interne RC-klok (op basis van een interne echte toevalsgenerator, TRNG). Op die manier zullen de kritieke bewerkingen (zoals het controleren van de PIN-code) niet altijd op hetzelfde moment plaatsvinden, wat aanvallen bemoeilijkt.
4. Het toevoegen van willekeurige ruis aan de voedingsstroom. Dit voorkomt stroomanalyse-aanvallen (figuur 10).
5. Wanneer een PIN wordt gecontroleerd, wordt eerst de PIN try-teller in de EEPROM van de SmartCard verlaagd, wordt vervolgens de PIN gecontroleerd en wordt daarna de PIN try-teller weer verhoogd als de PIN in orde is. Oude kaarten schreven alleen naar EEPROM om de teller te verlagen bij een verkeerde PIN-invoer. Dit maakte brute force-aanvallen om de PIN-code te kraken mogelijk door snel de voeding uit te schakelen na elke verkeerde poging.
6. Gebruik van uiterst zorgvuldig gemaakte functies voor veiligheidskritische bewerkingen (hier is assembler-programmering nodig!), die altijd hetzelfde aantal CPU-klokcycli en (mogelijk) dezelfde hoeveelheid stroom verbruiken, ongeacht de inputvariabelen. Dit voorkomt timing- en stroomanalyse-aanvallen.
7. Het verlengen van een veiligheidskritische operatie, die normaal gesproken minder dan 1 ms duurt, tot bijvoorbeeld 200 ms. Een 'echt' signaal van 1 ms zit dus verborgen in 199 ms verwerking van 'garbage'.

Een moderne bankkaart niet-invasief uitschakelen (als dat al mogelijk is) vereist uitgebreide analyse van geregistreerde gegevens en lang priegelwerk. Lees het boek *Power Analysis Attacks* [5] voor uitgebreidere



en uiterst gedetailleerde informatie over voedingsanalyse-aanvallen.

Testen en leren

De Poor Man's ChipTweaker PMCT is geen hi-tech schakeling, maar kan worden gebruikt voor het uitproberen en leren van niet-invasieve aanvallen op HSM's. Ik hoop dat je dit artikel interessant vond, in ieder geval als een goed uitgangspunt. Het ontwerpen en aanvallen van single-chip HSM's is tegenwoordig voor alle betrokkenen erg moeilijk. Juist daarom blijft dit gebied open voor verder onderzoek.

Kraak ze! 

210462-03

Vragen of opmerkingen?

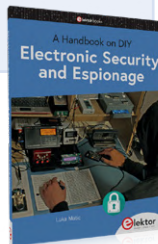
Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur via luka.matic@fer.hr of naar de redactie van Elektor via redactie@elektor.com.



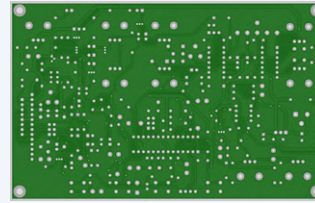
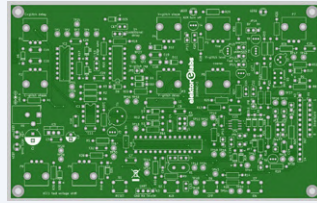
Gerelateerde producten

➤ **LabNation SmartScope USB Oscilloscope (SKU 17169)**
www.elektor.nl/17169

➤ **Luka Matic, *Electronic Security and Espionage*, Elektor 2021 (SKU 19903)**
www.elektor.nl/19903



ONDERDELENLIJST



Weerstanden:

R1,R29,R35 = 680 Ω
R2,R34 = 47 k
R3,R5,R6,R7,R9,R23,R25,R33,R38 = 10 k
R4,R8,R13 = 100 k
R10 = 1k5
R11,R12,R17,R22,R24,R26,R27,R32,R36 = 1 k
R14,R28,R30 = 4k7
R15 = 330 Ω
R16,R19 = 33 k
R18 = 2k2
R20* = 22 Ω
R21* = 10 k
R31,R37 = 100 Ω
R39,R40 = 3k3
P1,P2,P3,P5 = potmeter, 5 k, verticaal
P4 = potmeter, 10 k, verticaal
P6 = potmeter, 500 Ω, verticaal
P7 = potmeter, 2 k, verticaal
P8,P9 = potmeter, 25 k, verticaal

Condensatoren:

C1 = 100 μ/16 V, steek 3,5 mm
C2 = 10 μ/16 V, steek 2,5 mm
C3,C4,C7,C8,C9,C10,C11,C12,C20,C26,C28,C29 = 100 n, steek 5 mm
C5,C6,C17,C22,C25 = 22 p, steek 2,5 mm
C13,C24 = 100 pF, steek 2,5 mm
C14* = 10 n, steek 5 mm pitch
C27 = 10 n, steek 5 mm
C15*,C16* = 22 p, steek 5 mm
C18* = 33 p, steek 5 mm
C19* = 100 p, steek 5 mm
C21 = 220 p, steek 2,5 mm
C23 = 1 μ, steek 5 mm

Spoelen:

L1 = 10 μH



Funcard (bron: www.cellularcenter.it).

Halfgeleiders:

D1,D2,D3,D4,D5,D13,D14,D20,D21 = 1N4148
D11* = 1N4148
D6,D7,D8,D9,D10,D15,D16,D17,D18,D19 = BAT85
D12* = BAT85
IC1 = LM358
IC2 = 74HC00
IC3 = 74HC14
IC4 = LMV761, SOIC8
IC5 = 7805, TO220
IC6* = ATmega8A-PU
LED1,LED2 = LED, 3 mm, groen
LED3 = LED, 3 mm, geel
LED4 = LED, 3 mm, rood
T1 = BC141, TO39
T2 = BC557C
T3,T4,T5,T8,T9 = PN2369, TO92
T6 = BC547C
T7 = 2N2369, TO18

Diversen:

JP1,...JP13,JP15 = 2-polige pinheader, raster 2,54 mm
JP14,S3,S4 = 3-polige pinheader, raster 2,54 mm
K1 = 8-polige pinheader, raster 2,54 mm
K2 = 14-polige pinheader, raster 2,54 mm
K3 = 4-polige pinheader, raster 2,54 mm
K4,K5 = 2-rijige 6-polige pinheader, raster 2,54 mm
K6 = voedingsconnector
S5 = schuifschakelaar (C&K JS202011CQN)
X1* = kristal 8 MHz

* monteer met busstrip voor gemakkelijk experimenteren

WEB LINKS

- [1] Luka Matic, "Verzendbox met manipulatie-indicatie", Elektor mei/juni 2020:
<https://www.elektormagazine.nl/magazine/elektor-149/58607>
- [2] ChipWhisperers van NewAE Technology Inc.: <https://www.newae.com/chipwhisperer>
- [3] Sergei P. Skorobogatov, "Copy Protection in Modern Microcontrollers": https://www.cl.cam.ac.uk/~sps32/mcu_lock.html
- [4] Downloads bij dit artikel op Elektor Labs:
<https://www.elektormagazine.com/labs/poor-mans-chipwhisperer-or-a-smartcard-tweaker>
- [5] S. Mangard, E. Oswald, T. Popp, Power Analysis Attacks: Revealing the Secrets of Smart Cards, Springer, 2007:
<https://amzn.to/3RWBtuy>

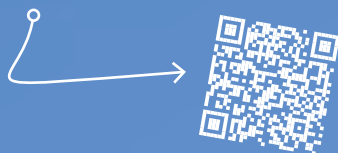
Elektor TV Shows



Elektor Engineering Insights

Elektor Industry Insights is een 'go-to' bron voor engineers en professionals die op de hoogte willen blijven over de wereld van de elektronica. Tijdens elke aflevering spreekt Elektor-redacteur Stuart Cording met experts uit de elektronica-industrie over technische uitdagingen en oplossingen.

www.elektormagazine.com/elektor-engineering-insights



Elektor LabTalk

Heb jij een passie voor DIY-elektronica, embedded systemen of algemene engineering theorie? Discussieer mee met het Elektor Labs team en onze redacteurs. Ze geven technische tips en bespreken toekomstige elektronica-projecten en de inhoud van Elektor Magazine. En uiteraard beantwoorden ze al jouw vragen!

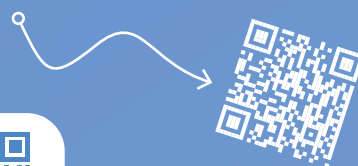
www.elektormagazine.com/elektor-lab-talk



elektor academy

Will jij jouw elektronikavaardigheden verbeteren? Dan is de Elektor Academy de 'place to be'. Onze expert Stuart Cording helpt u op weg via diverse cursussen over de meest uiteenlopende onderwerpen.

www.elektormagazine.com/elektor-academy



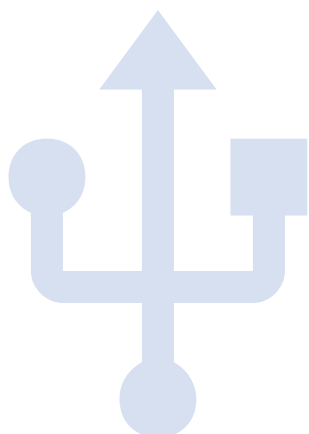
Blijf op de hoogte en volg
Elektor TV

www.youtube.com/c/ElektorIM



Echte USB-toevalsgenerator

twee PIC's voor de prijs van één AVR



Matthias Wolf (Duitsland)

Een goede toevalsgenerator (True Random Number Generator, TRNG) is handig om te hebben. Veilige gegevensversleuteling is bijvoorbeeld bijna onmogelijk zonder. Gaming- en goktoepassingen vereisen ook TRNG's van topklasse. Elektor heeft in 2017 een analoge TRNG gepresenteerd; in deze update voegen we daar USB aan toe.

De Echte Random Number Generator (TRNG) met goedkope componenten van Luka Matic [1] gebruikte een SD-kaart om de gegenereerde toevalsreeks op te slaan. Hier presenteren we een uitbreiding van dit project waarbij de SD-kaart is vervangen door een USB-interface.

De nieuwe schakeling (**figuur 1**) heeft twee PIC-microcontrollers van Microchip in plaats van een enkele ATtiny2313A: één voor de analoge signaalverwerking (PIC16F19156) en één voor de USB-interface (PIC18F25K50). Beide MCU's communiceren met elkaar via een SPI-bus, galvanisch geïsoleerd met snelle optocouplers om te voorkomen dat ruis van het digitale USB-gedeelte het analoge signaal verstoort.

Filterkalibratie

Er is een kleine PC-applicatie geschreven om de analoge filters op een gemakkelijke manier in realtime te kalibreren (**figuur 2**).

Mijn instellingen voor de analoge filters zijn als volgt:

- S5: C37,C38,C39,C40,C41 ON met C39 = 109 pF
- S6: C45,C46,C47 ON met C47 = 25 pF
- S7: schakelaar 1 ON met P6 = 409 Ω

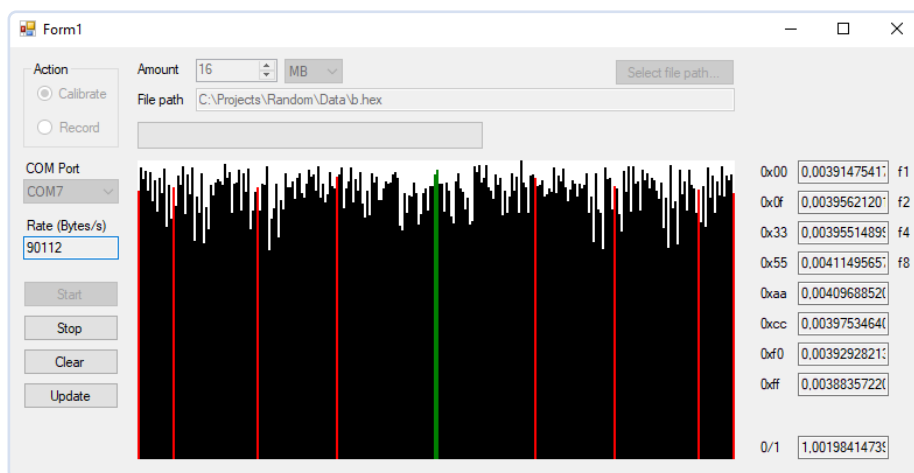
Deze instellingen zijn afhankelijk van de componenten die zijn gebruikt om de TRNG te bouwen; eventueel heb je andere nodig. Welke zenerdiodes worden gebruikt is ook belangrijk. In eerste instantie gebruikte ik de BZX384C12-E3-08 van Vishay, maar diens ruis was onbruikbaar.

Ten slotte heeft de PIC16, vergeleken met de originele TRNG van Luka, een iets hogere bemonsteringsfrequentie (800 kHz, 1,25 µs per sample).

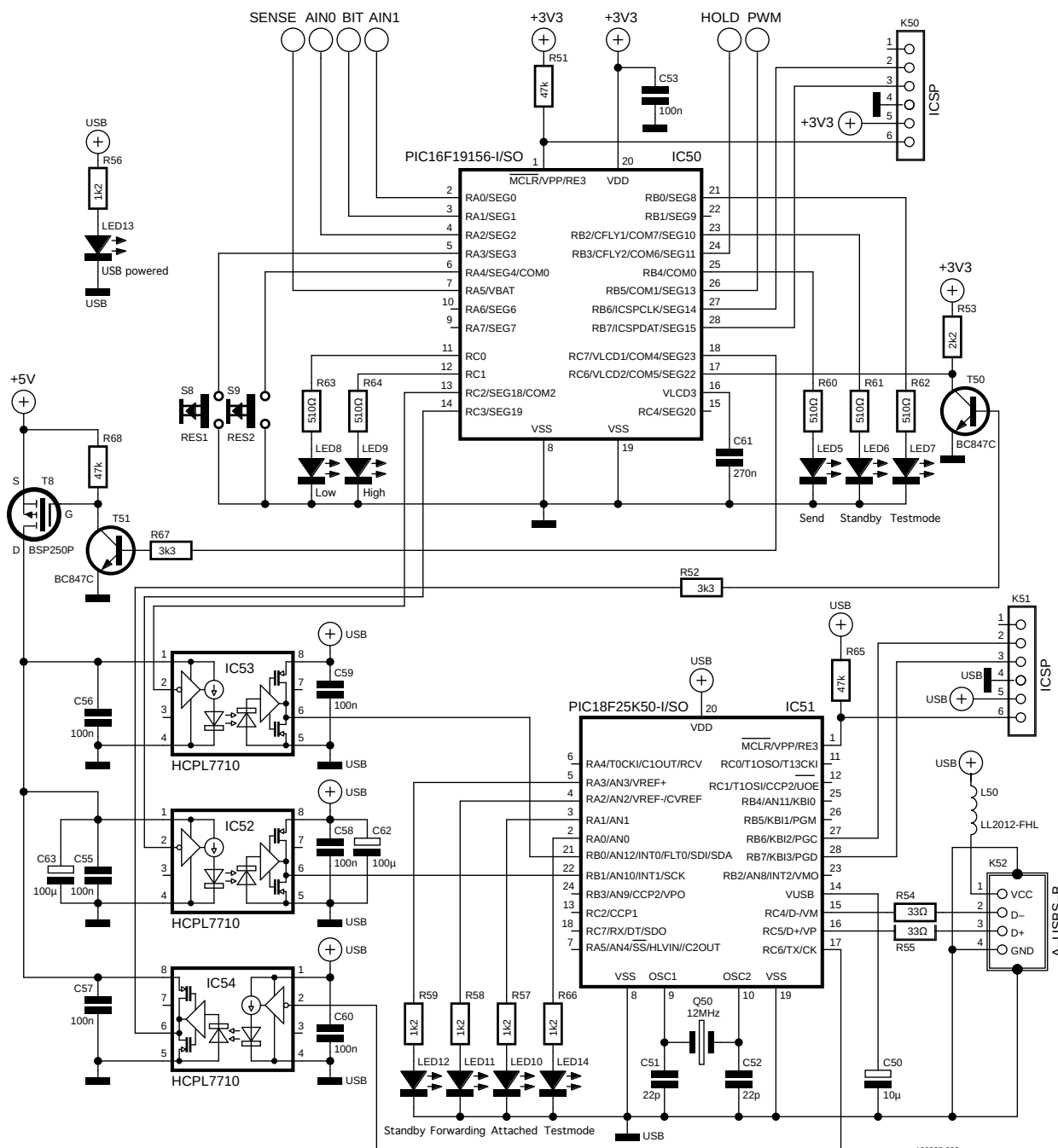
Haal de bestanden op!

Alle projectbestanden kunnen worden gedownload van [2]. Ze omvatten de schema's (drie pagina's), het Target 3001 CAD-project, firmware voor de PIC16 (analoge sampling, MPLAB X IDE met PCM C-compiler van CCS), firmware voor de PIC18 (USB-verwerking, MPLAB X IDE met XC8), en de desktop-applicatie (C#, Visual Studio 2017 community-editie). ◀

190235-03



Figuur 2. Screenshot van de PC-desktoptoepassing voor het kalibreren van de analoge filters en het opslaan van toevalsdata.



Figuur 1. Het digitale gedeelte van de originele TRNG uit 2017 met zijn ATtiny2313A MCU is vervangen door twee PIC-microcontrollers. De USB-poort vervangt de SD-kaarthouder.

Vragen of opmerkingen?

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



Gerelateerde producten

- > **L. Matic, *Electronic Security and Espionage***
(Elektor 2021, SKU 19903)
www.elektor.com/electronic-security-and-espionage

WEB LINKS

- [1] L. Matic, "Echte Random Number Generator", Elektor maart/april 2017:
<http://www.elektormagazine.nl/magazine/elektor-201703/40197>
- [2] Dit project op Elektor Labs: <https://www.elektormagazine.com/labs/usb-random-number-generator>

Pimp my mike...

opvoeren doe je zelf

Dr. Thomas Scherer (Duitsland)

Het jaar 2020 zal niet alleen 'vanwege Corona' niet snel vergeten worden, maar ook omdat plotseling veel mensen online moesten confereren – en niet alleen degenen die plotseling vanuit het home office moesten werken. Dat het beeld vaak verruist is – kan gebeuren. Maar hoe zit het met de geluidskwaliteit?

Nu *Skype* niet langer een vreemd woord is en ook de termen *Zoom* en *Teams* op ieders lip zijn, hebben veel tijdgenoten de afgelopen maanden veel ervaring opgedaan met videoconferenties – maar niet alleen positieve. Naast verkeerde bediening door sommige deelnemers zijn de beperkte bandbreedte van de internetverbindingen, de slechte kwaliteit van de videobeelden (slechte camera's en/of slecht licht) en *last but not least* het vaak beroerde geluid ergerlijk. Dit geldt vooral voor degenen die met een laptop onderweg zijn. In tegenstelling tot een PC is het met slechte laptop-hardware niet zo vanzelfsprekend om een betere webcam aan te schaffen.

Laptop-frustratie

Ik heb vrienden, kennissen en familieleden over de hele wereld, en daarom doe ik al lange tijd veel aan 'video-telefonie'. Omdat ik 'voor onderweg' ook draagbare computers gebruik, heb ik veel gevloekt over de kwaliteit van de videotransmissie met die dingen.

En vooral een ietwat duurder product van de fabrikant met de 'aangevroten' appel viel negatief op: nadat ik vorig jaar mijn oudere MacBook Pro had ingeruild voor een nieuwe MacBook Air, kon ik nauwelijks geloven wat een onzeglijk korrelig en donker beeld Apple daar voor veel geld waagde te verkopen. Ik had natuurlijk een extra webcam

kunnen kopen en deze via USB kunnen aansluiten, maar dat is toch de reden waarom je iets draagbaars koopt – dat je niet een heleboel extra's mee hoeft te zeulen? Bovendien heeft de MB Air alleen USB-C-interfaces. Zeer modern, maar webcams met USB-C zijn schaars en de noodzaak om ook nog een USB-C/USB-A-converter mee te moeten nemen was voor mij de spreekwoordelijke druppel. En zo geschiedde dat de nieuwe laptop werd verkocht en ik berouwwol terugkeerde naar het oudere model.

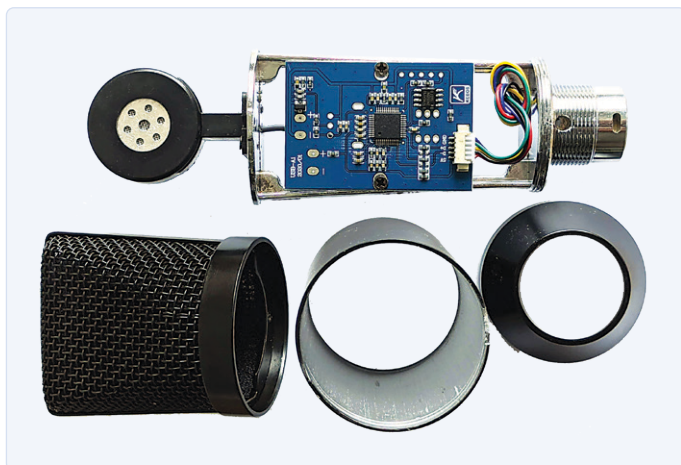
Het probleem van inferieure camera's (en microfoons) in klapcomputers is overigens niet beperkt tot Apple. Windows 10-gebruikers kunnen er ook mee te maken krijgen. Ik ken in ieder geval één (duur) Samsung-notebook waarvan het beeld verruist is en het geluid verschrikkelijk. Er zijn echter ook draagbare computers met een acceptabele videokwaliteit. Het is jammer dat precies deze aspecten worden verwaarloosd in de tests in de diverse tijdschriften. Het belang van goed beeld en geluid had in ieder geval sinds begin dit jaar duidelijk moeten zijn. Het is overigens niet echt begrijpelijk waarom elke goedkope mobiele telefoon een beduidend betere camera aan de voorkant en een redelijk bruikbare microfoon heeft – en laptops niet!

Oplossing

De eenvoudigste oplossing om alle beeld- en geluidsproblemen op te lossen is de aanschaf van een webcam. Voor een PC is dit sowieso nodig, maar een dergelijke aanschaf is ook het overwegen waard voor een bedrijfslaptop die voor het thuishkantoor ter beschikking wordt gesteld. Om tevreden te zijn met een extreem goedkope *no-name* webcam uit China moet je heel veel geluk hebben. Maar zowel Logitech



Figuur 1. Geleverd worden de microfoon zelf, de windkap, een klembeugel met statief en de USB-kabel.



Figuur 2. De nieuwe microfoon was snel gedemonteerd.

als Microsoft en andere merkfabrikanten bieden bruikbare webcams vanaf ongeveer € 35 aan. De ingebouwde microfoons zijn niet perfect, maar goed genoeg en meestal voldoende voor spraakoverdracht.

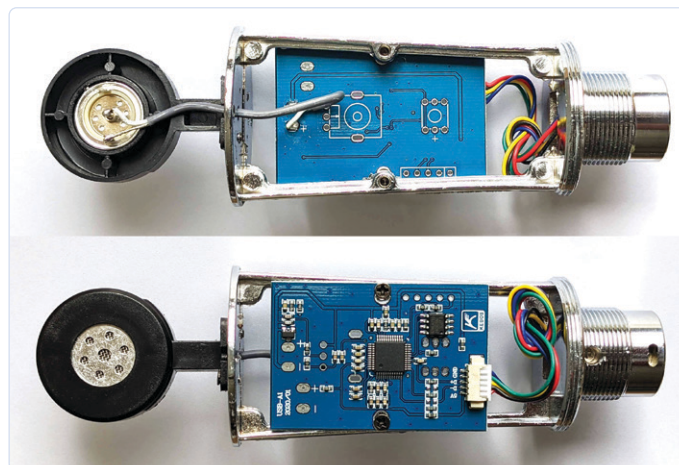
Als de kwaliteit van de camera voldoende is maar het geluid niet voldoet, kun je ook overwegen om een externe microfoon aan te schaffen. De industrie heeft deze markt ontdekt en biedt een aantal modellen met USB in plaats van klinkstekkers aan. Dankzij de geïntegreerde A/D-converter en de integratie van de benodigde stuurprogramma's in alle gangbare besturingssystemen levert aansluiting op een PC onder Windows, MacOS of Linux geen problemen op. U kunt natuurlijk ook 'normale' microfoons met een analoge uitgang gebruiken – in ieder geval op de meeste PC's. Bij moderne laptops is echter steeds vaker de microfooningang wegbezuinigd. Een goedkope USB-converter die zo'n ingang heeft en meestal ook een audio-uitgang kan dan uitkomst bieden. Alleen heb je dan weer zo'n klein ding dat kwijtraakt en waar je lang naar kunt zoeken. Niets voor mij.

USB-microfoon

Nadat ik de afgelopen jaren nooit echt tevreden was met verschillende goedkope microfoons, heb ik onlangs een zogenaamde 'grootmembraan-microfoon' met USB-stekker besteld. Deze condensatormicrofoons worden ook gebruikt in opnamestudio's en zouden daarom op zijn minst bruikbaar moeten zijn. Wikipedia biedt veel informatie over microfoons onder (inderdaad) het trefwoord 'microfoon' [1].

En afgezien daarvan: door de enorme groei van de *social media* produceren veel (jonge) mensen podcasts of exploiteren ze actief een eigen videokanaal. Dat vereist niet alleen een goede videocamera, maar ook een goede microfoon. Juist voor dergelijke doeleinden worden tegenwoordig 'podcast-sets' aangeboden, bestaande uit een microfoon met een groot membraan en een standaard, soms zelfs met een spin, een microfoonhengel en een plopscherm.

Dus ik wilde mijn geluk beproeven in deze sector. Twee zaken vallen op in het audiowereldje: enerzijds is niet elk gehanteerd argument ook steekhoudend, en anderzijds kan kwaliteit al snel erg duur worden. De prijs/prestatie-verhouding is helaas geen langzaam oplopende rechte lijn – de toename is veeleer exponentieel. En aangezien beter geluid me maar een beperkt bedrag waard was, googelde ik naar goedkope grootmembraan-USB-microfoons. En vond wat ik zocht.



Figuur 3. Voor- en achterkant van de ingewanden van de microfoon.

CAD Audio U29

Deze microfoon wordt geleverd inclusief statief en windkap en is met € 34 verrassend goedkoop. Daar kon ik geen buil aan vallen, dacht ik. Dus ik bestelde hem en twee dagen later werd ik na het uitpakken verblijd met de aanblik van **figuur 1**. Visueel maakt de microfoon een goede indruk: geen plastic maar helemaal van metaal.

Ik heb hem meteen op mijn PC aangesloten, de freeware Audacity gestart en er een opname mee gemaakt en ter vergelijking ook een opname gemaakt met de microfoon van mijn webcam (Logitech C525). Ik was niet teleurgesteld. Ik kon het duidelijk betere geluid horen en wel zo duidelijk dat een meting niet nodig was. Niet slecht!

Ik was benieuwd of mijn gesprekspartners een verschil zouden merken op Skype. En ook daar werd het verschil opgemerkt. Twee 'proefpersonen' konden heel duidelijk horen welke van de twee microfoons actief was toen ik omschakelde. Het verschil? De CAD-microfoon benadrukte het laag niet zo en had veel duidelijkere hoge tonen.

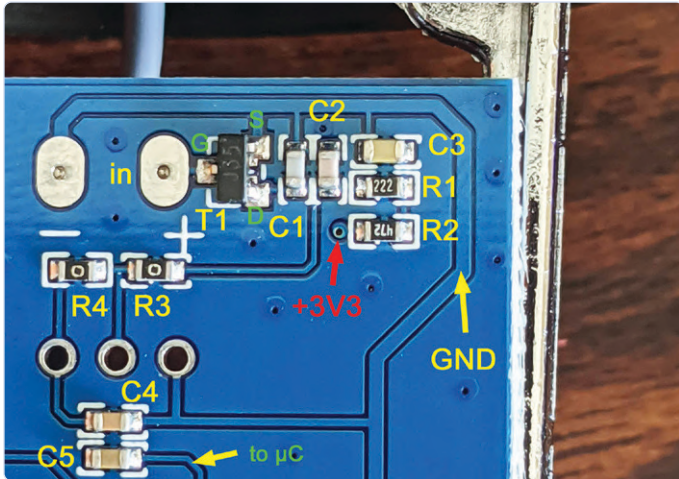
Niveaus en gevolgen

Eigenlijk had het verhaal hier kunnen eindigen. Tijdens de opnames merkte ik echter dat het niveau van de CAD-microfoon iets meer dan 5 dB lager was dan dat van de webcam, waardoor de microfoon 'zachter' klonk. In de programma's voor videoconferencing is het weliswaar mogelijk het niveau op te voeren, maar dan moest ik de regelaar 'helemaal opendraaien'.

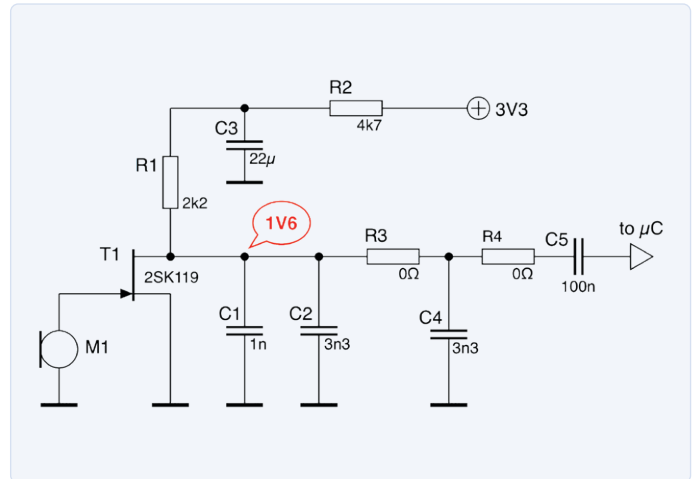
Dit is niet de eerste keer dat ik lage signaalniveaus bij microfoons tegenkom. Maar als elektronicus weet je wat je moet doen, daarom heb ik 'achteraf' versterkers in microfoons ingebouwd of de versterking van bestaande elektronica vergroot door componenten te wijzigen. Dat zou toch ook met deze microfoon moeten kunnen?

Zo gedacht was de microfoon al gedemonteerd, met het resultaat van **figuur 2**. Bovenaan het eigenlijke microfoonkapsel, waarvan de membraan een diameter van ongeveer 0,5" heeft, naar mijn mening aan de kleine kant voor 'grootmembraan'. **Figuur 3** toont de achterzijde van het kleine kapsel. Het zij zo.

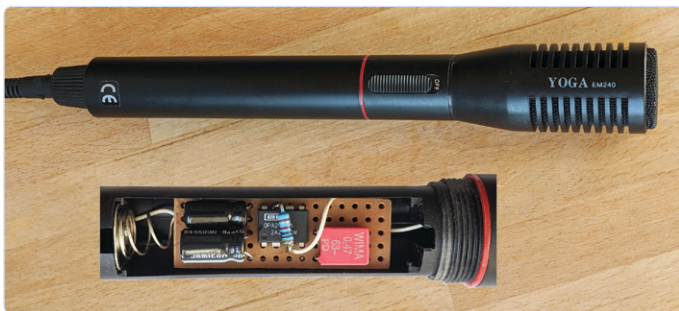
De print is interessanter: de chip in het midden is een SoC met audio-ingang en USB-interface. In tegenstelling tot de seriële EEPROM rechts-onder draagt de SoC waarschijnlijk omwille van de geheimhouding geen typenummer. Jammer! Maar de kleine zwarte SMD-driebener bovenin is de eigenlijke voorversterker.



Figuur 4. Detail van de impedantie-omzetter, met annotaties.



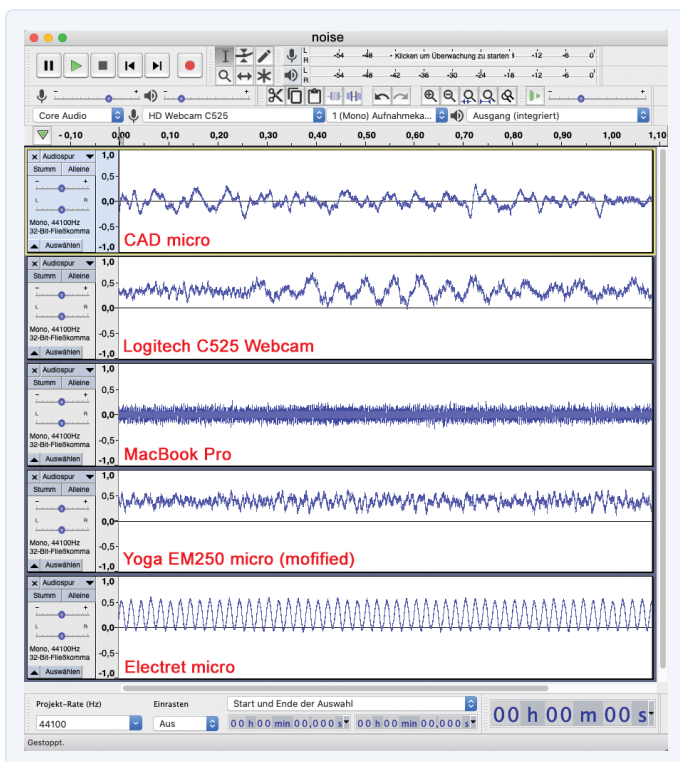
Figuur 5. Het schema van de impedantie-omzetter.



Figuur 6. De goedkope Yoga EM240-microfoon met ingebouwde voorversterker.



Figuur 7. Een klein elektretkapsel met een 5mm-membraan.



Figuur 8. Verschillende ruisniveaus van de diverse microfoons.

Een eerste meting op de twee pads '-' en '+' midden bovenaan toonde aan dat er precies 0,0 V op de microfoonmembraan staan. Dus geen voorspanning. Het moet dus een elektret-condensatormicrofoon zijn. Ondanks het goede geluid was ik een beetje teleurgesteld. De SMD-driebener bleek 'J35' te zijn, een N-kanaals JFET van het type 2SK1109 met een drainstroom van ongeveer 200 µA bij kortgesloten gate.

Impedantie-omzetter

Figuur 4 toont het gedeelte van de print rond deze JFET. Impedantie-omzetting is nodig omdat de capaciteit van de microfoonmembraan erg laag is en de impedantie voor audiosignalen erg hoog. Dankzij de gemeenschappelijke source-schakeling versterkt T1 – uitgangspunt voor een hoger uitgangsniveau. In figuur 5 zien we de schakeling van de onderdelen van figuur 4. R2 filtert samen met C3 de voedingspanning. Over R1 en R2 valt in totaal ongeveer 1,7 V; de drainstroom is dus ongeveer 245 µA.

Wat opvalt: zonder de DC-instelling te verstoren kunnen we R1 en R2 omwisselen, waardoor de versterking tweemaal zo groot wordt ($\approx +6$ dB). Dat levert geen problemen op, aangezien het laagdoorlaatfilter van 2,2 kΩ en 22 µF nog steeds bij een acceptabele 3,2 Hz ligt. Het laagdoorlaatfilter van R1 = 4,7 kΩ en C1+C2+C4 ligt dan echter lager dan 5 kHz. Dit kan worden veranderd door C2 of C4 te verwijderen.

Resultaten

Na de aanpassing was het niveau nu zelfs iets hoger dan van mijn webcam en was het geluid ongewijzigd. Dus de procedure is goed verlopen.

Nu wilde ik weten hoe de gemodificeerde CAD-microfoon klinkt in vergelijking met andere microfoons. Dus maakte ik opnames met mijn webcam, mijn MacBook Pro en ook met een oude goedkope microfoon (**figuur 6**) en een goedkoop kaal elektret-microfoonkapsel (**figuur 7**). Het is niet verwonderlijk dat de MacBook Pro (subjectief) het slechtst klonk. Het gebrek aan laag maakte dat mijn sonore stem blikkerig klonk. Jaren geleden had ik een ruisarme voorversterker in de Yoga-microfoon ingebouwd omdat deze ondanks het lage niveau in de oorspronkelijke toestand teveel ruiste.

Om u een indruk te geven heb ik voor dit artikel een paar korte mp3-bestanden gemaakt die u van de projectpagina bij dit artikel kunt downloaden [3]. U hoort een korte zin en ongeveer een seconde ruis – alles met hetzelfde niveau zodat alleen de klank verschilt. **Figuur 8** toont de vijf verschillende ruisniveaus, elk zo'n 30..40 dB versterkt. Alleen de microfoons van CAD, Logitech en Yoga vertonen 'echte' ruis. De Logitech vertoont ook een gesuperponeerd gezoem, en bij de MacBook Pro HF en het elektretkapsel is netbrom te horen.



Figuur 9. Zo staat de CAD-microfoon correct: rechts (of links) naast de monitor.

Conclusie

Ben ik tevreden met het geluid van de CAD-microfoon? Mijn antwoord is een tweeduidig 'ja maar'...

Aan de ene kant is mij een kleinmembraan-microfoon verkocht als grootmembraan-exemplaar. Papier en HTML zijn geduldig. Aan de andere kant was hij niet duur en eigenlijk beter dan de microfoon van de webcam, en dat is ook wat waard.

Overigens: een microfoon als deze wordt niet recht van voren 'besproken' maar vanwege de richtkarakteristiek van opzij, zoals te zien in figuur 9. ◀

200434-04

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

- Eenvoudige voorversterker voor professionele microfoons
<https://www.elektormagazine.nl/magazine/elektor-201705/40298>
- Eenvoudige voorversterker voor professionele microfoons – kale print
<https://www.elektor.nl/simple-preamp-for-professional-microphones-bare>
- Duitstalig boek: Audioschaltungstechnik (PDF)
www.elektor.de/audioschaltungstechnik-pdf



WEBLINKS

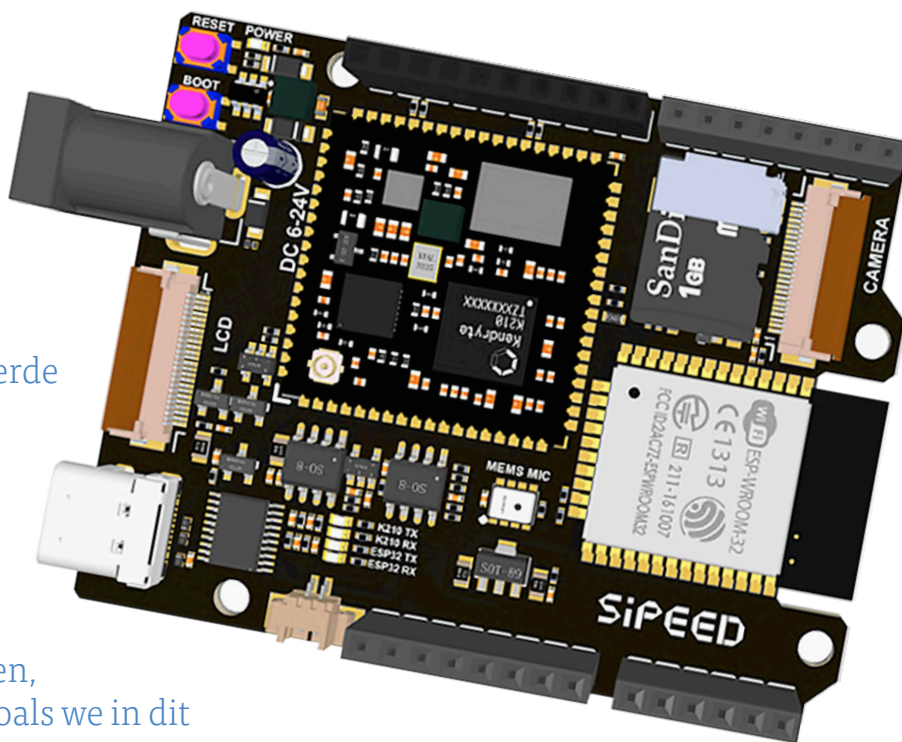
- [1] Wikipedia over microfoons:
<https://nl.wikipedia.org/wiki/Microfoon>
- [2] Audacity: <http://www.audacity.de>
- [3] Audiobestanden: <http://www.elektormagazine.nl/200434-04>

FFT met Maixduino

frequentiespectra vastleggen

Tam Hanna (Slowakije)

Een snelle Fouriertransformatie (Fast Fourier Transform, FFT) is niet alleen bruikbaar voor audiotoeepassingen. Wanneer u de geluiden van een turbine of de spanningsniveaus van slecht gebufferde bussen analyseert, kunt u in veel gevallen conclusies trekken over toerentallen en dergelijke. Het Maixduino-board met K210-SoC dat bij Elektor verkrijgbaar is, is niet alleen geschikt voor de eerste experimenten met neurale netwerken, maar ook voor snelle realtime FFT, zoals we in dit artikel laten zien.



De voortdurende ontwikkeling van zowel software als hardware heeft ertoe geleid dat technologieën die ooit alleen het domein waren van particuliere milities en regeringen ook voor gewone stervelingen beschikbaar zijn gekomen.

Het halfgeleiderbedrijf Sipeed, dat tot nu toe vooral bekend stond om zijn experimenten met de RISC-V-processor, biedt in de vorm van de Maixduino nu een nieuw ontwikkelboard aan. Naast een ESP32 voor netwerkcommunicatie bevat het board ook een SoC met de naam *K210 Kendryte*, die door de fabrikant bedoeld is voor gebruik in AI-toepas-

singen. Met het noemen van de naam Kendryte bezondigen we ons niet aan sluikreclame: veel informatie is in Google alleen te vinden onder de zoekterm "kendryte + problem".

Als u meer over de gebruikte SoC wilt weten, kunt u de soms wat langzaam reagerende website [1] bezoeken. In het datablad van de K210 staat vooral een beschrijving van de pinnen, maar het architectuurschema van **figuur 1** is ook interessant.

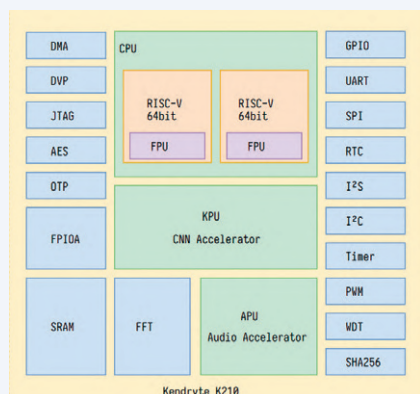
De hoofdprocessor (CPU) is gebaseerd op de open-source RISC-V-architectuur (in de varianten I, M, A en C). Deze kernen worden geflankeerd door twee accelerators. In de eerste plaats is er de KPU, waarachter een *neural network processor* schuilt (zonder verdere informatie in het datablad), die een neuraal netwerk in hardware realiseert. Analoog aan de DPO Tristar-processor die ooit door Danaher in de TDS 5xxD en TDS 7xxD werd gebruikt, hebben we ook in dit geval te maken met een 'one-trick pony'.

De andere accelerator is de audioprocessor met de aanduiding APU. Deze verwerkt audiogegevens met maximaal 192 kHz, slaat deze gegevens via DMA direct op in het hoofdgeheugen en kan ook automatisch FFT-processen toepassen. We zullen deze APU in de volgende stappen gebruiken voor het maken van een kleine FFT.

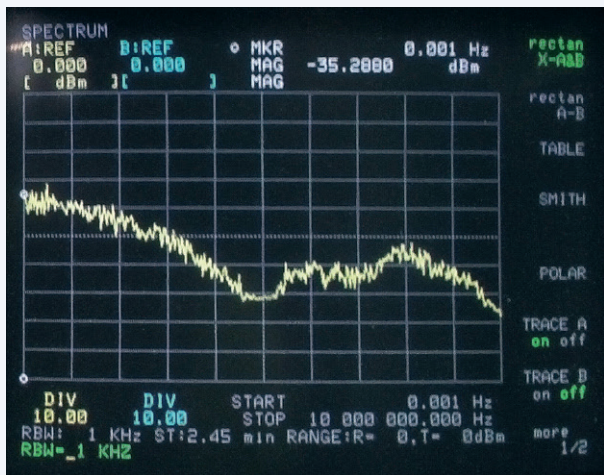
Voor de volledigheid wijzen we nog op de gebruikelijke periferie, maar daar gaan we hier niet nader op in.

Een kwestie van gereedschap

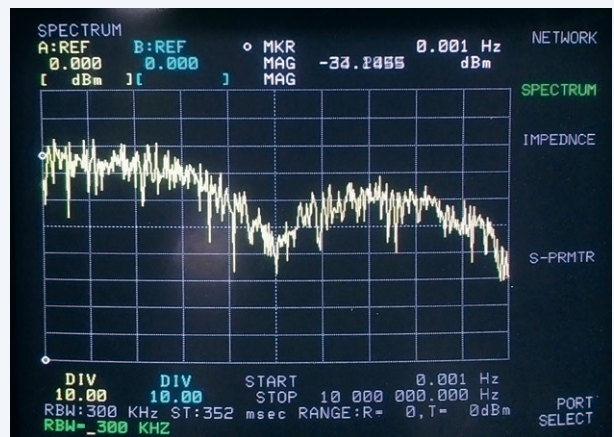
Interessanter voor ons zijn de ontwikkeltools. Op het laagste niveau is er de standalone SDK die onder [2] beschikbaar is voor download – het is een bibliotheek (gelukkig) in C, die ons in staat stelt om direct



Figuur 1: De K210 brengt een aantal uitbreidingen met zich mee (bron: Canaan Inc.).



Figuur 2: Smalle frequentiebanden leveren een hoge resolutie op, maar vergen veel Sweep Time (ST)...



Figuur 3: ...terwijl sneller sweepbare, bredere banden minder nauwkeurige informatie geven.

te communiceren met processor-primitieven. Een niveau hoger is er de ArduinoCore [3], een framework dat bedoeld is voor de Arduino IDE, maar dat ook met Platform.IO kan worden gebruikt. Last but not least is er met MaixPy een van MicroPython afgeleide programmeeromgeving die bedoeld is voor iedereen die veel ervaring heeft met Python (lees eerst het **tekstkader!**).

Om de mogelijkheden van FFT te demonstreren gebruiken we audio-data – Maixduino wordt geleverd met een MEMS-microfoon, die via de I²S-bus is aangesloten op de procescomputer. Dit is geen ‘hoogvlieger’ – de voorbeeldtoepassing die onder [4] is gepubliceerd, was waarschijnlijk een van de eerste applicaties. Laten we beginnen met het insluiten van enkele benodigde headerbestanden:

```
#include <Sipeed_ST7789.h>
#include "i2s.h"
#include "fft.h"
```

Dan moeten we gaan nadenken over hoe de te verwerken buffers opgebouwd moeten zijn. In de datasheet die in de inleiding is genoemd vinden we dat de Kendryte niet met willekeurige groottes overweg kan – als u hardwareversnelling wilt gebruiken, moet u kiezen tussen de buffergroottes 64, 128, 256 of 512.

Ook al stuit het sommige wiskundigen tegen de borst, toch is het verstandig om altijd een klassieke spectrumanalyzer in het achterhoofd (of nog beter op uw werktafel) te hebben als u met FFT aan de slag gaat. In ons geval is de bucket-grootte relevant voor zover deze de fijnheid van de resulterende frequentiebanden beschrijft.

Figuur 2 en figuur 3 zijn gefotografeerd van de aloude HP4195A van de auteur en illustreren de principiële gang van zaken. Terwijl onze analoge spectrumanalyzer in beide gevallen ‘onrustige’ banden laat zien als gevolg van ruis, is de curve bij een FFT in werkelijkheid gelijkmatiger. De frequentieresolutie volgt uit de formule *Sampling Rate / FFT-Size*; wanneer 1024 monsters worden genomen bij 8192 Hz, is de resolutiebandbreedte $RBW = 8 \text{ Hz}$

Net als zo vaak geldt ook bij de FFT dat meer niet noodzakelijkerwijs ook beter is. Het verband tussen het aantal frequentiebanden en het aantal inkomende samples is in principe een factor 2. Als kleine complicatie komt daar nog bij, dat FFT-algoritmes zeer snel zeer complex worden naarmate het aantal samples toeneemt. Dit leidt tot een gedrag zoals geschetst in **figuur 4**, waar de nauwkeurigheid van onze FFT-operatie beperkt wordt door problemen uit de ‘echte’ wereld.

In de volgende stap definiëren we een reeks constanten. In de praktijk wordt aanbevolen om met name de FFT-lengte via een **#define** te specificeren om latere aanpassing van nauwkeurigheid versus snelheid te vergemakkelijken:

```
uint32_t rx_buf[1024];
#define FFT_N 512U
#define FFT_FORWARD_SHIFT 0x0U
#define SAMPLE_RATE 38640
#define FRAME_LEN 512
```

Voor de eigenlijke opslag van de door de DMA geleverde informatie is een groep extra geheugenvelden nodig:

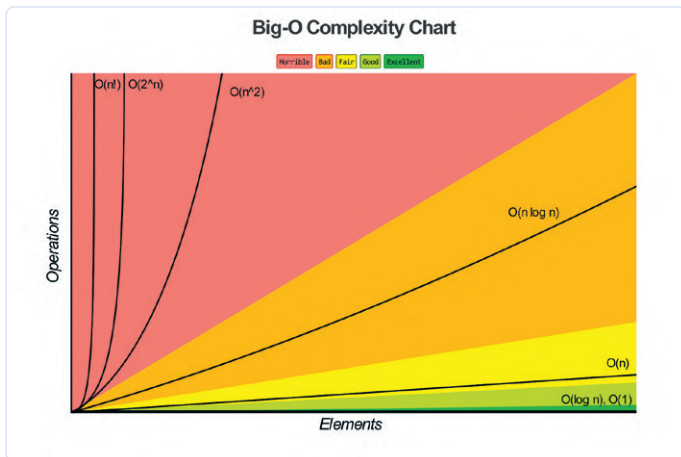
```
int16_t real[FFT_N];
int16_t imag[FFT_N];
int hard_power[FFT_N];
uint64_t fft_out_data[FFT_N / 2];
uint64_t buffer_input[FFT_N];
uint64_t buffer_output[FFT_N];
fft_data_t *output_data;
fft_data_t *input_data;
complex_hard_t data_hard[FFT_N] = {0};
```

Initialisatie van de hardware

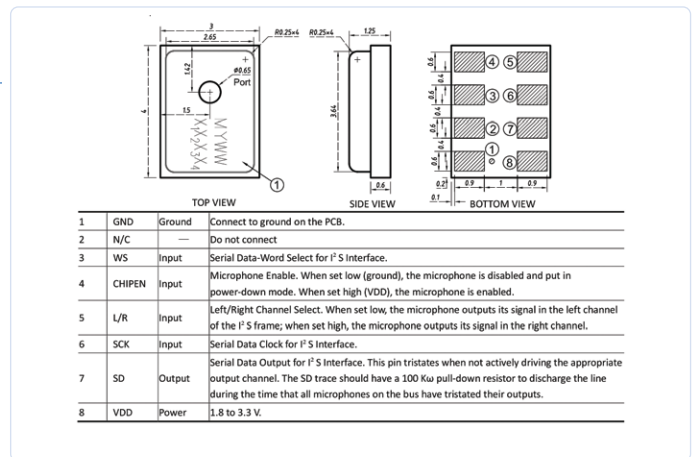
Zoals in de inleiding vermeld, schenkt de fabrikant van de Maixduino ons een MEMS-microfoon van het type MSM261S4030H0. Omdat de omgang met deze kleine microfoons altijd weer interessant is, verwijzen we naar **figuur 5**. De pintoewijzing laat duidelijk zien dat de component zelf voor de signaalaanpassing zorgt en de gegevens via een seriële bus beschikbaar maakt.

Het eerste obstakel is dat de software niet automatisch de datalijnen initialiseert die nodig zijn voor de communicatie met de microfoon. Dit is eenvoudig te verhelpen in het begin van **setup**:

```
void setup()
{
    lcd.begin(15000000, COLOR_RED);
    fpioa_set_function(20, FUNC_I2S0_IN_D0);
    fpioa_set_function(18, FUNC_I2S0_SCLK);
    fpioa_set_function(19, FUNC_I2S0_WS);
```



Figuur 4: Als u oneindig kleine frequentiebanden wilt, heeft u veel processorcapaciteit nodig (foto: <https://www.bigocheatsheet.com/>).



Figuur 5 MEMSensing bespaart de sensorgebruiker veel werk (foto: MEMSensing Microsystems).

De volgende stap is de configuratie van de I²S-hardware. De code wordt in principe bepaald door de hardware van het board; belangrijk is de sample rate die moet worden ingesteld met de methode `i2s_set_sample_rate`. Aangezien we relatief weinig opties hebben gezien de grootte van de FFT, is de sample rate een van de weinige middelen om de frequentiebandbreedte te 'verwerken':

```
i2s_init(I2S_DEVICE_0, I2S_RECEIVER, 0x3);
i2s_set_sample_rate(I2S_DEVICE_0, SAMPLE_RATE);
i2s_rx_channel_config(I2S_DEVICE_0, I2S_CHANNEL_0,
    RESOLUTION_16_BIT, SCLK_CYCLES_32,
    TRIGGER_LEVEL_4, STANDARD_MODE);
```

Daarna ontbreken alleen nog enkele initialisaties:

```
dmac_init();
sysctl_enable_irq();
}
```

Onze volgende taak is het uitvoeren van de eigenlijke FFT-operatie. De eerste stap is ervoor te zorgen dat de door I²S verstrekte gegevens in het geheugen terecht komen:

```
void loop()
{
    int i, sampleID;

    i2s_receive_data_dma(I2S_DEVICE_0, &rx_buf[0],
        FRAME_LEN * 2, DMAC_CHANNEL3);
    dmac_wait_idle(DMAC_CHANNEL3);
```

Door de aanroep van `dmac_wait_idle` geven we de processor opdracht te pauzeren tot het DMA-kanaal vrijkomt. Dit zorgt ervoor dat elke doorloop van `loop` met een consistente en nieuwe database werkt. Als u de aanroep van `dmac_wait_idle` (als experiment) weglaat, krijgt u vreemde aliasing-artefacten op het scherm te zien. Dit is waarschijnlijk te wijten aan het feit dat de FFT-modules volgens het datablad alleen via de DMA-engine kunnen worden gebruikt. Als er hier conflicten optreden, zal onheil uw deel zijn!

Bij het werken met de FFT-engines en vergelijkbare vaste functie-eenheden moet u er altijd op letten dat de hardware de gegevens in een vaste geheugenstructuur levert. In het geval van `tag_fft_data` is dit

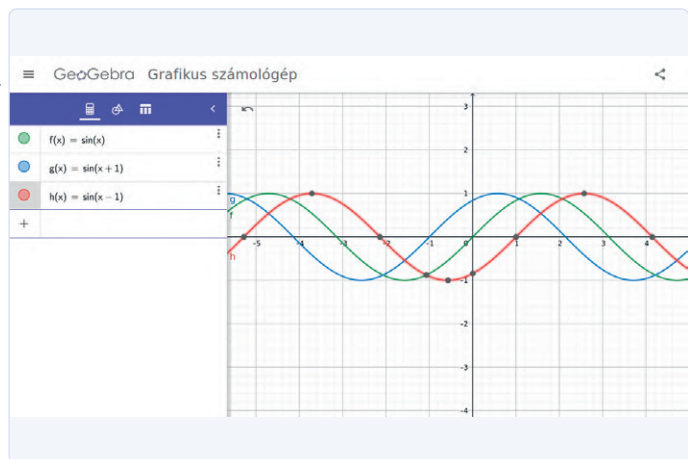
de onderstaande structuur, die op verschillende plaatsen in de SDK wordt gedeclareerd:

```
typedef struct tag_fft_data
{
    int16_t I1;
    int16_t R1;
    int16_t I2;
    int16_t R2;
} fft_data_t
```

Oplettende lezers zullen zich nu afvragen waarom er twee componenten (R en I) verwacht worden – om dit uit te leggen, moeten we allereerst vertellen dat R voor *reëel* staat en I voor *imaginair*. De Fourier-transformatie is in het klassieke (symbolisch-mathematische) geval gedefinieerd als een bewerking op basis van complexe getallen. FFT-algoritmen verwachten daarom – althans in de generieke versie, die hardwareontwikkelaars met het oog op maximale flexibiliteit graag in hardware implementeren – input van zowel reële als imaginaire componenten. Onze audiodata bestaan echter uitsluitend uit reële elementen, en daarom zetten we de waarden van I op dit punt altijd op nul:

```
for ( i = 0; i < FFT_N / 2; ++i)
{
    input_data = (fft_data_t *)&buffer_input[i];
    input_data->R1 = rx_buf[2*i];
    input_data->I1 = 0;
    input_data->R2 = rx_buf[2*i+1];
    input_data->I2 = 0;
}
```

Hier komt de 'recycling' van het FFT-proces goed van pas. De klassieker "Numerical Recipes in C" [5], die tweedehands soms goedkoop en online gratis als PDF verkrijgbaar is, heeft in hoofdstuk 12 (van de tweede druk) hiervoor enkele interessante formules in de aanbieding, voor het begrijpen waarvan echter grondige wiskundekennis vereist is. De beloning van alle inspanning is dan dat een FFT-proces twee reële functies tegelijkertijd verwerkt – de ontwikkelaar kan dan de uitgevoerde data met zeer weinig moeite van elkaar scheiden. In de praktijk is de FFT een van die gebieden waar u als elektronicus tot in alle eeuwigheid mee kunt spelen – deze procedure is in de praktijk vaak effectiever dan een immense wiskundige leerinspanning.



Figuur 6: Deze trillingen zijn ondanks identieke frequentie en amplitude niet identiek.

Als volgende stap willen we de FFT-commando's uitvoeren om dan de resultaten te krijgen:

```
fft_complex_uint16_dma(DMAC_CHANNEL0, DMAC_CHANNEL1,
    FFT_FORWARD_SHIFT, FFT_DIR_FORWARD, buffer_input,
    FFT_N, buffer_output);
```

Let hier op de overgave van de operatie `FFT_DIR_FORWARD`, die de hardware-engine informeert dat we van het tijdsdomein naar het frequentiedomein gaan. In de praktijk zijn er altijd situaties waarin u een signaal vanuit het frequentiedomein terug naar het tijdsdomein wilt overbrengen – in dit geval zou hier de constante `FFT_DIR_BACKWARD` hier worden overgedragen. Vervolgens kunnen we eindelijk onze gegevens 'oogsten':

```
for ( i = 0; i < FFT_N / 2; i++)
{
    output_data = (fft_data_t*)&buffer_output[i];
    data_hard[2 * i].imag = output_data->I1 ;
    data_hard[2 * i].real = output_data->R1 ;
    data_hard[2 * i + 1].imag = output_data->I2 ;
    data_hard[2 * i + 1].real = output_data->R2 ;
}
```

Let erop dat de geretourneerde gegevens een reëel en een imaginair deel hebben. Om dit probleem te begrijpen, gaan we, zij het kort, terug naar de wiskunde. De formule

$$S_f(t) = a_0/2 \sum [a_n \cos(n\omega t) + b_n \sin(n\omega t)]$$

is een van de vele voorstellingen van de fundamentele Fourier-reeks. Het gebruik van I of J in de formule laat zien dat we met complexe eenheden werken – aangezien de FFT in principe slechts een numerieke weergave is van de klassieke Fourier-transformatie, moet deze de regels volgen.

Hoezo complex?

Onze zoëven uitgevoerde sampling-operatie is in de praktijk weliswaar reëel, maar maakt de wiskundige niet gelukkig. Een reden hiervoor is dat er geen 'gedefinieerde' trigger is – door een wiskundige bril bekeken zijn de in **figuur 6** getoonde sinussignalen uit fase. De FFT-operatie drukt dit uit door het feit dat de geretourneerde waarden complexe getallen voorstellen. Hiermee kan zowel de amplitude als de fase van



ONZE PRIJZEN

DE BESTE BESCHERMING TEGEN HOGE KOSTEN

The best part of your project:
www.reichelt.com

Met reichelt haalt u meer uit uw budget

Dankzij de efficiënte, zelf ontwikkelde logistiek en de bundeling van onze inkoopkracht voor geselecteerde kwaliteitsproducten, kunnen wij zeer kleine hoeveelheden voor superprijzen leveren.

The best part of your project

Of het nu gaat om de automatisering van uw productieprocessen of om slimme energievoorziening, met geschikte techniek uit ons assortiment kunnen uw projecten ook in de toekomst efficiënt worden gepland en uitgevoerd.




Portenta breakout uitbreidingsboard

Bestelnr.: ARD SHD ASX00031

46,⁷²

(38,61)




NVIDIA Jetson Nano Module

Bestelnr.: JETSON NANO

180,⁴⁸

(149,16)

Ontdek nu ► www.reichelt.com/best-part

THE BEST PART OF YOUR PROJECT

DIGITALE TRANSFORMATIE

Ontdek nu ► <https://rch.it/digi-nl>




Betaalwijzen:      

Geweldige prijs-kwaliteitverhouding Meer dan 130.000 geselecteerde producten
Betrouwbare levering - vanuit Duitsland over de hele wereld



reichelt

elektronik – The best part of your project

www.reichelt.com

Bestelservice: +31 85 208 62 94

De wettelijke herroepingsregelingen zijn van toepassing. Alle aangegeven prijzen in euro inclusief de wettelijke BTW, excl. verzendkosten voor de totale winkelwagen. Uitsluitend onze Algemene Voorwaarden (zie hiervoor <https://rch.it/AV-NL> of op aanvraag) zijn van toepassing. Afbeeldingen kunnen afwijken. Drukfouten, vergissingen en prijswijzigingen voorbehouden.
reichelt elektronik GmbH & Co. KG, Elektronikring 1, D-26452 Sande, Tel.: +31 85 208 62 94

de betreffende component worden uitgedrukt – de twee sinusgolven zouden ook in de uitkomst verschillend zijn.

Een elektronicus is meestal niet geïnteresseerd in de complexe componenten, dus we werken ze weg door de absolute waarde te berekenen:

```
float pmax=10;
for (i = 0; i < FFT_N; i++)
{
    hard_power[i] = sqrt(data_hard[i].real *
                        data_hard[i].real + data_hard[i].imag
                        *
                        data_hard[i].imag);
}
```

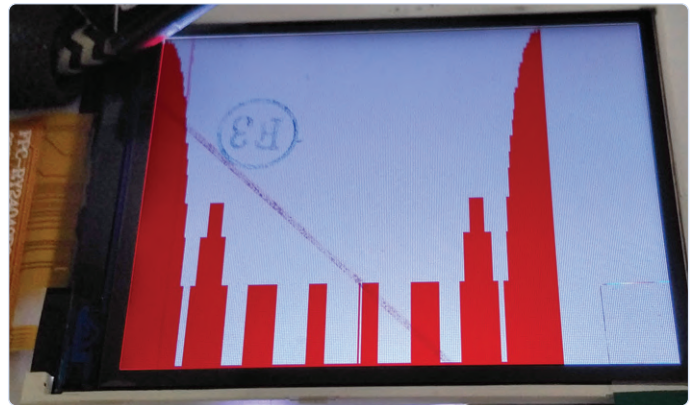
Logaritmen en vierkantswortels nodigen uit tot vereenvoudiging, zodat de berekeningen mathematisch kunnen worden geoptimaliseerd. Eenvoudiger betekent echter niet dat de procedures gemakkelijker te begrijpen zijn – integendeel. Daarom moet de vereenvoudiging omwille van de duidelijkheid hier worden weggelaten.

Als u de eerder genoemde combinatie van twee verschillende functies in één FFT-bewerking uitvoert, zou u op dit punt een extractie moeten uitvoeren en de twee resultaten moeten scheiden (met een in vergelijking met de FFT eenvoudig algoritme). In de volgende stap nemen we nog de logaritme om dB-achtige waarden te krijgen:

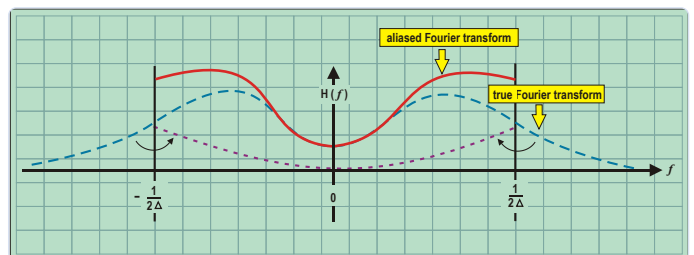
```
hard_power[i] = 20*log10(2*hard_power[i]/FFT_N);
if( hard_power[i]>pmax && i>5)
    pmax = hard_power[i];
}
```

Als volgende moeten we de waarden op een scherm weergeven. Het grootste probleem hier is de display-API van de Arduino, die bij het doorgeven van 'ongeldige' coördinatenwaarden heel wat aan *performance* inboet:

```
lcd.setRotation(0);
lcd.fillScreen(COLOR_WHITE);
for (int i=0;i<256;i++)
{
    int dval = 240 - 240*(hard_power[i]/pmax);
    if (dval<2)dval=1;
    if (dval>238)dval=238;
```



Figuur 7: Deze 'symmetrie' is verwarrend.



Figuur 8: Door het Nyquist-venster 'geëlimineerde' spectrale elementen duiken als artefacten op.

```
    lcd.drawLine( i, 240, i, dval , COLOR_RED);
}
delay(50);
}
```

Hiermee zijn we klaar voor een eerste test, die tot het resultaat in **figuur 7** leidt. Als u wilt kunt u op dit punt nog testtoontjes aan het systeem toevoegen – de beloning is dat u dan de testtoon c.q. de lijn die deze vertegenwoordigt op het scherm kunt verschuiven.

Speigeltje, spiegeltje...

De FFT is een van die onderwerpen waarmee u zich als elektronicus eindeloos kunt bezighouden. Het zal u zeker zijn opgevallen dat een testtoon zowel van de 'linker' als van de 'rechter' kant over het scherm verplaatst.

PYTHON IS NIET GELIJK AAN PYTHON...

Op het gebied van Python voor microcontrollers strijden twee concepten om de heerschappij. Het ene concept is MicroPython, dat een volwaardige runtime implementeert. Het doelsysteem bevat een complete Python-interpreter, die net als een werkstation zelfs bibliotheken van het internet laadt. Het voordeel van deze aanpak is een grotere flexibiliteit, het nadeel is dat er problemen kunnen ontstaan door de beperkte systeembronnen (trefwoord: RAM).

De andere benadering wordt momenteel alleen vertegenwoordigd door Zerynth, dat niet beschikbaar is op Maixduino. Het werkstation creëert een monolithische applicatie die op het doelsysteem wordt gebrand. Tegenover de geringere flexibiliteit staat een grotere stabiliteit – als de bibliotheken op het werkstation staan, kunnen processen die veel systeembronnen in beslag nemen, buiten de microcontroller worden afgehandeld.

Hier liggen verschillende obstakels op de loer. Onze FFT bijvoorbeeld herhaalt zich al bij het 128e veld, wat volgens de zojuist vastgestelde theorema's pas bij het 256e veld zou mogen gebeuren. De reden hiervoor is een probleem met de aanlevering van de samples via de I²S-bus – een onderwerp waar we nu niet nader op in zullen gaan. Onder [6] vindt u een aangepaste versie die het probleem verzacht. De nadruk ligt hier op 'verzacht' omdat bij FFT's bij volledig reële invoerwaarden altijd een spiegeling optreedt. De waarden van nul tot N/2 komen daarbij overeen met die van N/2+1 tot N. De reden hiervoor is dat de complexe onderdelen van de FFT elkaar moeten opheffen. Overigens zijn er nog veel gebieden om verder te spelen met de FFT, bijvoorbeeld de introductie van een vensterfunctie om de invoer te 'besnijden'. Het achterliggende idee is dat delen buiten het Nyquist-venster aan beide zijden naar binnen worden gespiegeld, terwijl een vensterfunctie dit zou onderdrukken (zie **figuur 8**). Aanbevolen lectuur hierover is "Some Windows with Very Good Sidelobe Behavior" [7] van Albert H. Nuttall (1981).

Conclusie

De hardware-FFT-engine maakt laagdrempelige experimenten met FFT-algoritmen en -toepassingen mogelijk. Deze competentie is steeds weer nodig in zowel de civiele als de militaire sector. Als u zo'n board koopt en er een beetje mee speelt, zult u zeker geen spijt hebben van het geld en de tijd die u daarin hebt geïnvesteerd. ◀

200297-04



Gerelateerde producten

> **Sipeed MAix BiT Kit for RISC-V AI+IoT**
www.elektor.nl/sipeed-maix-bit-kit-for-risc-v-ai-iot

WEBLINKS

- [1] Kendryte K210: <https://bit.ly/3fptbHu>
- [2] Standalone-SDK: <https://github.com/kendryte/kendryte-standalone-sdk>
- [3] ArduinoCore-k210: <https://github.com/Seeed-Studio/ArduinoCore-k210>
- [4] fft_lcd: https://github.com/MrJBSwe/fft_lcd/blob/master/main.c
- [5] Teukolsky, Press, Vetterling und Flannery: Numerical Recipes in C: www.numerical.recipes/
- [6] stft_lcd: https://github.com/jpiat/stft_lcd
- [7] Albert H. Nuttall: Some Windows with Very Good Sidelobe Behavior: <https://bit.ly/3fTkCWb>

Listing 1. De volledige Maixduino-listing.

```
#include <Sipeed_ST7789.h>
#include "i2s.h"
#include "fft.h"
SPIClass spi_(SPI0); // MUST be SPI0 for Maix series on board LCD
Sipeed_ST7789 lcd(320, 240, spi_);
uint32_t rx_buf[1024];
#define FFT_N          512U
#define FFT_FORWARD_SHIFT 0x0U
#define SAMPLE_RATE    38640
int16_t real[FFT_N];
int16_t imag[FFT_N];
int hard_power[FFT_N];
uint64_t fft_out_data[FFT_N / 2];
uint64_t buffer_input[FFT_N];
uint64_t buffer_output[FFT_N];
fft_data_t *output_data;
fft_data_t *input_data;
complex_hard_t data_hard[FFT_N] = ;
#define FRAME_LEN      512

void setup()
{
    lcd.begin(15000000, COLOR_RED);
```

vervolg op volgende pagina

```

    fpioa_set_function(20, FUNC_I2S0_IN_D0);
    fpioa_set_function(18, FUNC_I2S0_SCLK);
    fpioa_set_function(19, FUNC_I2S0_WS);

    i2s_init(I2S_DEVICE_0, I2S_RECEIVER, 0x3);
    i2s_set_sample_rate(I2S_DEVICE_0, SAMPLE_RATE );
    i2s_rx_channel_config(I2S_DEVICE_0, I2S_CHANNEL_0,
                        RESOLUTION_16_BIT, SCLK_CYCLES_32,
                        TRIGGER_LEVEL_4, STANDARD_MODE);

    dmac_init();
    sysctl_enable_irq();
}

void loop()
{
    int i, sampleID;

    i2s_receive_data_dma(I2S_DEVICE_0, &rx_buf[0], FRAME_LEN * 2, DMAC_CHANNEL3);
    dmac_wait_idle(DMAC_CHANNEL3); //wait to finish recv

    for ( i = 0; i < FFT_N / 2; ++i)
    {
        input_data = (fft_data_t *)&buffer_input[i];
        input_data->R1 = rx_buf[2*i]; // data_hard[2 * i].real;
        input_data->I1 = 0;           // data_hard[2 * i].imag;
        input_data->R2 = rx_buf[2*i+1]; // data_hard[2 * i + 1].real;
        input_data->I2 = 0;           // data_hard[2 * i + 1].imag;
    }

    fft_complex_uint16_dma(DMAC_CHANNEL0, DMAC_CHANNEL1, FFT_FORWARD_SHIFT, FFT_DIR_FORWARD,
                          buffer_input, FFT_N, buffer_output);

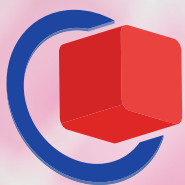
    for ( i = 0; i < FFT_N / 2; i++)
    {
        output_data = (fft_data_t *)&buffer_output[i];
        data_hard[2 * i].imag = output_data->I1 ;
        data_hard[2 * i].real = output_data->R1 ;
        data_hard[2 * i + 1].imag = output_data->I2 ;
        data_hard[2 * i + 1].real = output_data->R2 ;
    }

    float pmax=10;
    for (i = 0; i < FFT_N; i++)
    {
        hard_power[i] = sqrt(data_hard[i].real * data_hard[i].real +
                             data_hard[i].imag * data_hard[i].imag);

        //Convert to dBFS
        hard_power[i] = 20*log10(2*hard_power[i]/FFT_N);
        if( hard_power[i]>pmax && i>5)
            pmax = hard_power[i];
    }

    lcd.setRotation(0);
    lcd.fillScreen(COLOR_WHITE);
    for (int i=0;i<256;i++)
    {
        int dval = 240 - 240*(hard_power[i]/pmax);
        if (dval<2)dval=1;
        if (dval>238)dval=238;
        lcd.drawLine( i, 240, i, dval , COLOR_RED);
    }
    delay(50);
}

```

embeddedworld2023

Exhibition & Conference

... it's a smarter world



JOIN THE EMBEDDED
COMMUNITY

14–16.3.2023



Get your
free ticket now!

embedded-world.com/voucher

Use the voucher code **GG4ew23**

Media partners

Markt & Technik
Das unabhängige Wochenblatt für Elektronik

Elektronik

SmarterWorld
Solutions for a Smarter World

DESIGN &
ELEKTRONIK
KNOW-HOW FÜR ENTWICKLER

Elektronik
automotive

•medical-design

computer &
automation

elektroniknet.de

NÜRNBERG MESSE



Uit het leven gegrepen

ontwerp(on)logica

Ilse Joostens (België)

In mijn tienerjaren heb ik veel elektronische schakelingen als een kip zonder kop nagebouwd. De exacte werking van een schakeling interesseerde me maar matig en vooral een zo snel mogelijk resultaat was mijn belangrijkste drijfveer.

Ook de eindafwerking van een project liet nogal eens te wensen over; ik was al lang blij als mijn bouwsels min of meer werkten. Gaandeweg heb ik de stap naar het zelf ontwerpen van schakelingen gezet, eerst gewoon als hobby maar uiteindelijk ook commercieel. Vooral in dat laatste geval moest ik afstappen van het idee om steeds de best mogelijke schakeling te ontwerpen. Ook financiële overwegingen, de beschikbaarheid van componenten in de gewenste aantallen en de eigen voorraadstatus spelen in dat geval mee en soms neem je daardoor als ontwerper beslissingen die op het eerste zicht niet altijd even logisch lijken.

*Amphenol 36-way connector, selected in 1970
for printers. Foto: Shutterstock / KPixMining*

Centronics-fiasco

Voor een van mijn vroegere projecten had ik op een gegeven ogenblik een groot aantal heldere RGB-LED's nodig die ik goedkoop besteld had bij een Duitse leverancier. De LED's waren dik in orde en de verkoper was zelfs zo lief geweest om er een berg weerstanden bij te leveren met de bedoeling deze als voorschakelweerstand te gebruiken. Ik had deze weerstanden eigenlijk niet nodig; de meeste hadden dan ook nog een wat minder gebruikelijke waarde van 510 Ω . Ik wou ze dus kwijt en uiteindelijk heb ik ze maar gebruikt in het ontwerp van mijn VFD-shield voor Arduino (R1 in het schema) [1]. Nu is de waarde van die weerstand niet kritisch en al gauw werd ik slachtoffer van mijn eigen succes. De verkoop van de kitjes liep als een trein en voor ik het wist was ik door mijn voorraad 510 Ω -weerstanden heen en moest ik ze zelfs bijbestellen. Niet alleen moest ik een meer permanente stocklocatie voor deze weerstanden vinden maar ze waren toen ook nog een stuk duurder dan pak 'm beet een 470- of 560 Ω -weerstand. Over in eigen voet schieten gesproken! Het kitje was zelfs zo mateloos populair dat er

aangepaste versies op het internet verschenen, waaronder ook een ontwerp met zes buisjes. De grap is wel dat iedereen als een kip zonder kop die 510Ω-weerstand heeft overgenomen.

Onwillekeurig moet ik nu terugdenken aan de Centronics-interface [2]. In 1970 had Centronics een goedkope dotmatrix-printer (model 101 [3]) ontwikkeld. Toevallig had het toenmalige Amerikaanse moederbedrijf Wang Laboratories nog een surplusvoorraad van 20.000 stuks 36-polige Amphenol flatcable-connectoren liggen die oorspronkelijk gebruikt werden voor een van hun vroege rekenmachines. Al gauw werd beslist deze connectoren dan maar te gebruiken voor de parallelle interface van de printer. De naar mijn gevoel lompe connector groeide uit tot een de facto industriestandaard en ik wil écht niet weten hoeveel Centronics er naderhand nog heeft moeten bijbestellen.

Kostenefficiëntie en 'Muntzing'

Enige tijd geleden ben ik op een Duits internetforum op kritiek op mijn ontwerp van de VFD-klok met ESP32 [4] gestoten. Alles begon met een persoon die een montagefout had gemaakt bij het bouwen en dan maar de schuld in mijn schoenen als ontwerper wilde schuiven. Een ander forumlid merkte vervolgens op dat het gebruik van een 47μH-spoel in de step-up converter aanleiding gaf tot grote piekstromen en dat het geheel weinig efficiënt was aangezien de gate van de MOSFET rechtstreeks door de uitgang van de 7555 aangestuurd werd. De kritiek is ergens terecht, maar u moet weten dat ik omwille van andere

projecten zowat verzuip in 47μH-spoelen en dus ook geneigd ben ze te gebruiken waar mogelijk. De klok verbruikt in totaal ongeveer 2,25 W waarvan de step-up converter iets meer dan 550 mW voor zijn rekening neemt. Bij dergelijk grote vermogens lig ik niet wakker van efficiëntie en ga ik zeker geen dure state-of-the-art step-up converter ontwerpen. Als efficiëntie echt zo belangrijk is, neem dan een LC-display in plaats van VFD-buizen. De uitstraling is helaas niet dezelfde. Hoewel ik me soms durf bezondigen aan enige overengineering, moet het financiële plaatje wel blijven kloppen. Niet dat ik zo ver wil gaan als de flamboyante zakenman Earl "Madman" Muntz [5][6] die naar verluidt steeds met een zijknip-tang in zijn borstzakje rondliep om op tijd en stond 'overbodige' onderdelen uit de ontwerpen van zijn ingenieurs te knippen om zo de kosten te drukken. Ondanks alles dan toch nog van hebzucht beschuldigd worden met een hint naar 'inhalige zakenbankiers' op dezelfde fora omdat de prijzen van je kits zogezegd aan de hoge kant liggen is wel wat laf. Waarschijnlijk hebben die mensen nog nooit van 'belastingen' en 'loonkosten' gehoord denk ik dan maar.

Intuïtie versus ratio

Op school kwam het vooral aan op berekenen, beredeneren en zo nodig ook bewijzen

met de nadruk op theorie en toch wel een zekere minachting voor de praktijk en al helemaal voor empirische methodes. Je baas zal het echter zeker niet weten te appreciëren als je urenlang zit te rekenen aan een simpele schakeling en ook

voor mij als zelfstandige geldt het aforisme 'time is money.' Uiteraard bezondig ik me af en toe aan berekeningen maar ik durf ook intuïtief te ontwerpen, zeker voor de meer eenvoudige dingen; sommige schakelingen vereisen nu eenmaal een empirische aanpak, vooral als je met obscure vintage-onderdelen werkt. Durf ook vooral bestaande ontwerpen te hergebruiken in plaats van telkens het warm water opnieuw uit te willen vinden.

Ook Guglielmo Marconi is voor de ontwikkeling en verfijning van de magnetische radio-detector, bijgenaamd 'Maggie' [7] empirisch te werk gegaan, en dit op basis van het werk van Ernest Rutherford en Harry Shoemaker. De magnetische radiodetector heeft goede diensten bewezen tot de opkomst van de detectors met vacuümbuizen en een dergelijk toestel was ook aan boord van de Titanic. Pas twintig jaar later waren wetenschappers in staat om adequaat te verklaren hoe het toestel werkte. Een ode aan aan Marconi's bekwaamheid als praktisch ingenieur dus, met de nadruk op 'praktisch'. ◀

220671-03

Disclaimer: alle overbodige tekst is zorgvuldig uit deze column verwijderd met de delete-knop die ik constant bij me draag in mijn handtas.



Guglielmo Marconi. Foto: Shutterstock / Morphart Creation.

WEBLINKS

- [1] Ilse Joostens, "VFD-shield voor Arduino", ElektorMag 9-10/2015: <https://www.elektormagazine.nl/magazine/elektor-201509/28039>
- [2] Parallelle poort: https://nl.wikipedia.org/wiki/Parallelle_poort
- [3] Manuals Library – Centronics 101 Operator's Manual: <https://www.manualslib.com/manual/1438231/Centronics-101.html>
- [4] "VFD-klok met ESP32", ElektorMag 5-6/2018: <https://www.elektormagazine.nl/magazine/elektor-201805/41518>
- [5] Bob Pease – What's All This Muntzing Stuff, Anyhow?: <https://www.electronicdesign.com/technologies/boards/article/21771148/whats-all-this-muntzing-stuff-anyhow>
- [6] Documentaire – Madman Muntz: <https://www.youtube.com/watch?v=deFIB2G0mH8>
- [7] Magnetische detector: https://en.wikipedia.org/wiki/Magnetic_detector

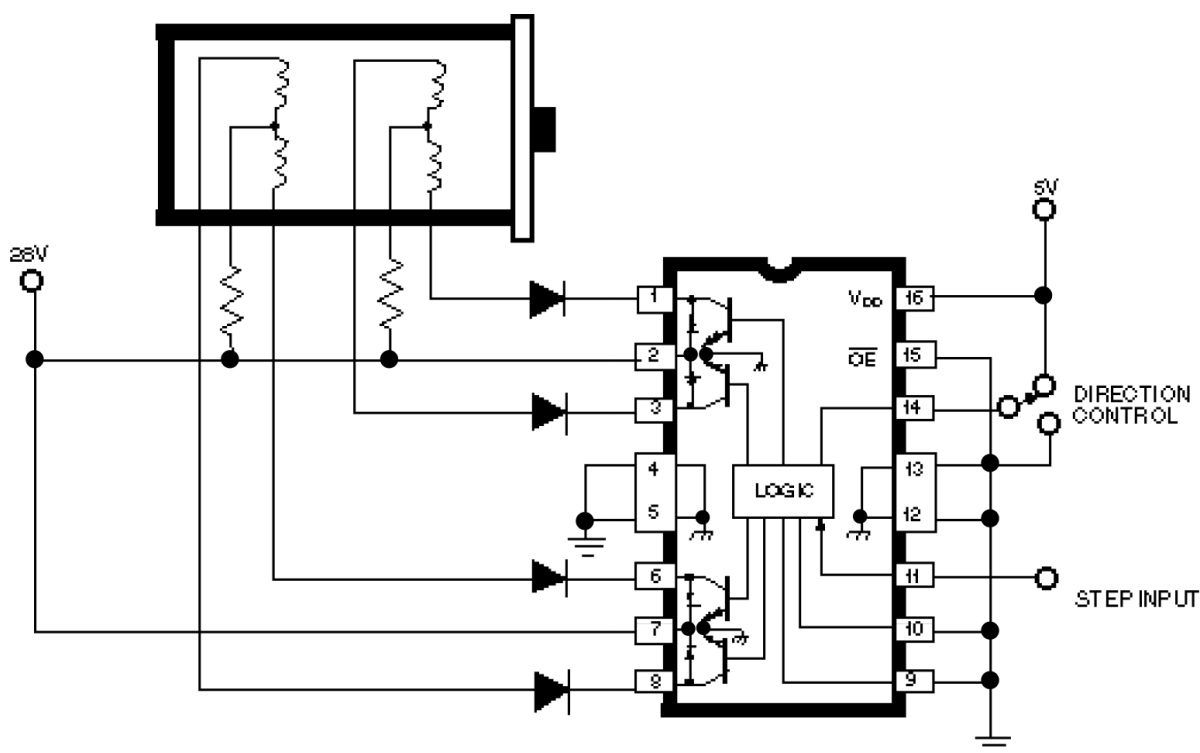
UCN5804 stappenmotordriver

vreemde onderdelen

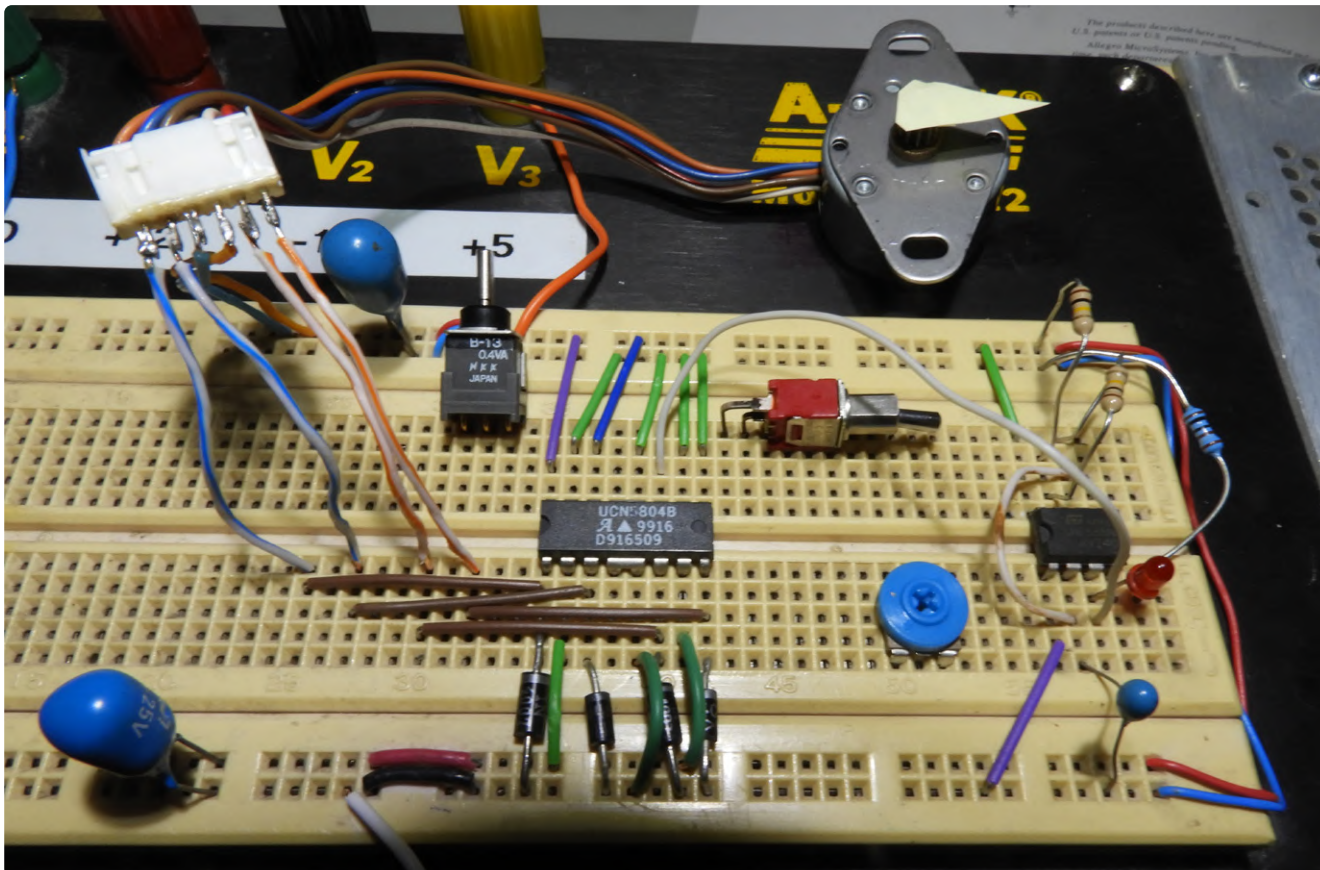
David Ashton (Australië)

Een stappenmotordriver? Wat is daar vreemd aan? Er zijn er waarschijnlijk honderden! Nu, de bijzonderheid van deze ligt in zijn eenvoud. Het is gewoon een IC met step- en direction-ingangen om direct een stappenmotor aan te sturen – andere logica of IC's zijn niet nodig.

De UCN5804 is een stappenmotordriver-IC gemaakt door Allegro Microsystems. Het wordt geleverd in een 16-pins (DIP of SOIC) behuizing, waarbij de middelste pinnen aan elke zijde (4/5 en 12/13) met elkaar en met massa verbonden zijn. Deze zorgen voor warmteafvoer en de massaverbinding. Het is een klein maar geweldig IC dat vier wikkelingen van een vierfase unipolaire stappenmotor kan aansturen met alleen de ingangen *step* en *direction*. Zijn uitgangsvermogen (1,5 A bij maximaal 35 V) is ook indrukwekkend, zodat hij de meeste kleine stappenmotoren rechtstreeks kan aansturen. Er zijn ook enkelfase- en half-step-ingangen voor wat meer veelzijdigheid. Lekker recht-toe-rechtaan, en makkelijk toe te passen.



Figuur 1: UCN5804-schakeling. Bron: Allegro Microsystems datasheet.



Figuur 2: Een breadboard-schakeling met een teruggevonden UCN5804, een 555 voor de stappentiming en enkele schakelaars voor de aansturing.

Helaas wordt het niet meer geproduceerd, hoewel op de Allegro-web-site [1] de datasheet nog steeds te zien is, en dat is lovenswaardig – veel bedrijven verwijderen de datasheets van componenten die niet meer geproduceerd worden.

Een typische schakeling is weergegeven in **figuur 1**. Zoals te zien, is alles wat je nodig hebt om een stappenmotor aan te drijven een handvol diodes. De richtingsregeling is gerealiseerd met een schakelaar, maar dat kan net zo goed een signaal van een microcontroller of een andere logische schakeling zijn.

Waarom is Allegro dan gestopt met die kleine maar geweldige stappenmotordriver? Waarschijnlijk omdat het een ouder bipolair IC is, geen MOS, en geen van de energiebesparende eigenschappen heeft die fabrikanten tegenwoordig nodig hebben om hun systeem elektrisch efficiënt te maken. De discontinuïteitsmededeling op de datasheet beveelt de A3967 en A3977 van Allegro aan, die volledig anders zijn, want deze zijn gemaakt voor het aansturen van bipolaire stappenmotoren (die tegenwoordig inderdaad vaker voorkomen) met H-brug drivers. Ze hebben ook veel meer toeters en bellen, zoals stroomdetectie, PWM-aansturing, en halve, kwart en achtste stappen. Een andere belangrijke omissie, vooral voor hobbyisten zoals ik die nog steeds met gaatjesprint knutselen, is dat deze twee IC's alleen verkrijgbaar zijn in respectievelijk 24- en 28-pins SOIC-behuizingen.

Er zijn verschillende andere stappenmotordrivers, maar ze doen allemaal in een of ander opzicht onder voor de UCN5804. De bekende L297/298-combinatie is behoorlijk veelzijdig, maar dat is een oplossing met twee componenten. Andere IC's die pretenderen stappenmotordrivers te zijn, zijn slechts halve H-brug schakelingen en zonder de mooie en eenvoudige *step*-en *direction*-ingangen. Dan moet je zelf als onverschrokken ontwikkelaar de stuursignalen genereren met een ander IC of een microcontroller.

Het is erg jammer dat Allegro met dit IC is gestopt, want het is ideaal voor hobbyisten (ja, ik weet het, hobbyisten zijn een kleine markt). In **figuur 2** zie je een breadboard-schakeling met de UCN5804 samen met een 555-timer en wat schakelaars. Gelukkig heb ik er nog een paar over van afgedankte tape back-up drives. Maar als je goed rondkijkt, kun je ze nog steeds op het internet vinden voor ongeveer vijf tot tien dollar per stuk. ◀

220530-03 (vertaling: Willem den Hollander)

Vragen of opmerkingen?

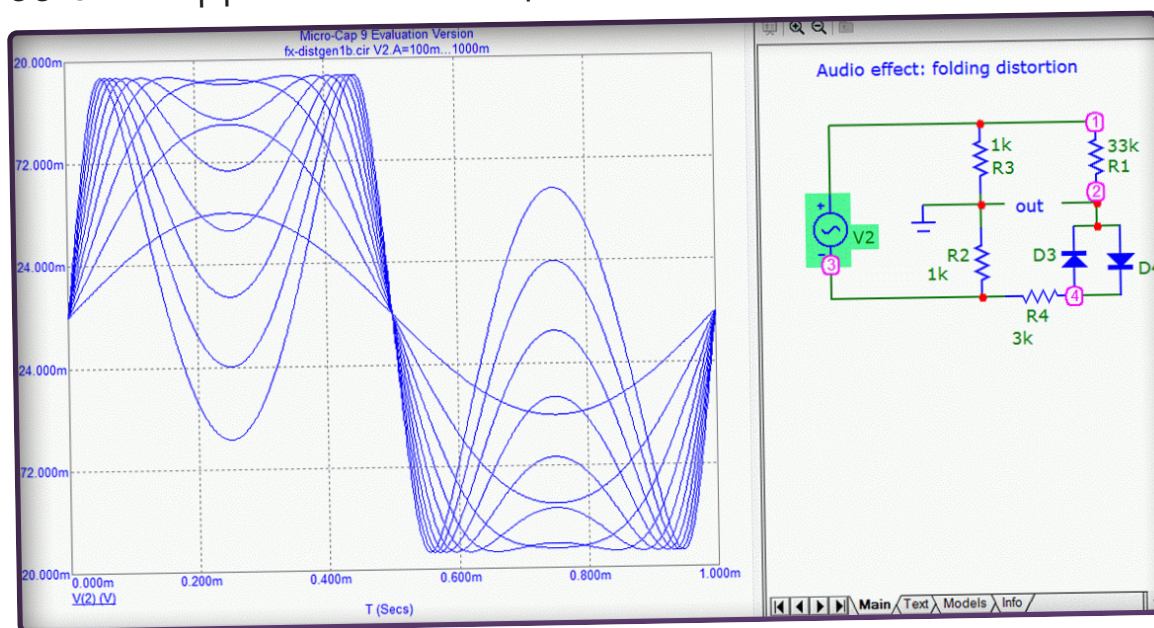
Heeft u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de redactie van Elektor via redactie@elektor.com.

WEBLINKS

[1] Allegro Microsystems - UCN5804 datasheet: <https://elektor.link/ucn5804>

Simuleren met Micro-Cap

eerste stappen in een complexe wereld



Raymond Schouten (Nederland)

Een simulator hoort thuis in de gereedschapskist van elke elektronicus, net als een voeding, een multimeter en een oscilloscoop. En net als elk ander gereedschap moet je, voordat je er goed mee uit de voeten kan, er eerst mee leren omgaan. In dit artikel laten we zien hoe je aan de slag kunt met de freeware-simulator Micro-Cap.

Waarom zou je simuleren?

Het gebruik van een simulator kan veel tijd besparen bij het testen van ideeën of het dimensioneren van componenten. Het is ook een goed leer- en debug-gereedschap dat aanschouwelijk maakt wat er in een schakeling gebeurt.

Simulatiesoftware zoals LTspice (Analog Devices), Tina (Texas Instruments) en Micro-Cap gebruiken de tekstgebaseerde SPICE-

code die in 1973 is ontwikkeld aan de Universiteit van Californië, Berkeley. Moderne software voegt daar voornamelijk een grafische interface aan toe. Een belangrijk deel van dit artikel geldt daarom voor alle drie genoemde simulatoren, vooral als het gaat om het vermijden van simulatiefouten.

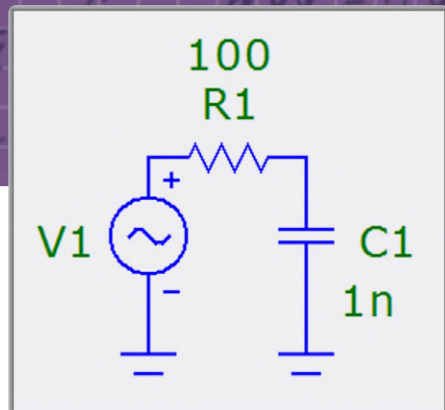
De simulator die in dit artikel wordt geïntroduceerd is Micro-Cap [1]. Het is een simulator van professionele kwaliteit (\$ 4500) die als

freeware werd uitgebracht nadat de product-ontwikkeling ervan was gestopt. Vergeleken met andere gratis simulatoren heeft het vrij geavanceerde functies. Het nadeel is het risico van veroudering omdat er niet meer aan gewerkt wordt. Voorlopig kun je deze software gratis vinden op [1].

Met Micro-Cap kun je schakelingen tekenen en de signalen erin bekijken terwijl je aan 'knoppen' draait om parameters te veranderen. Het wordt geleverd met een grote bibliotheek met componentmodellen (inclusief vacuümbuizen!). Ontwerptools voor actieve en passieve filters zijn ingebouwd. Een leuk detail voor audiotoeepassingen is dat je de golfvorm van je gesimuleerde schakeling kunt beluisteren.

Hoe simuleer je?

Het simuleren van een schakeling is als het bouwen ervan op breadboard. Je plaatst de componenten, verbindt ze met elkaar om er een schakeling mee te maken, voegt



Figuur 1. Mijn eerste schakeling: een eenvoudig RC-laagdoorlaatfilter.

voedingsspanningen toe en sluit (optioneel) een signaalbron aan. Vervolgens kun je de signalen meten met een oscilloscoop (in het tijd domein) of met een spectrumanalyzer (in het frequentiedomein). Meer details hierover zijn te vinden in het kader **Verschillende analyses**.

Je eerste schakeling tekenen

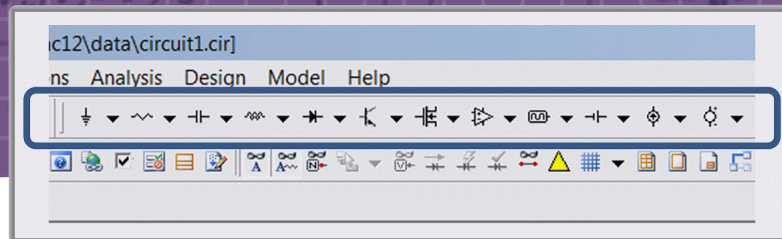
Na het opstarten van Micro-Cap beginnen we met het tekenen van een eenvoudig RC-laagdoorlaatfilter en simuleren we de frequentierespons ervan zoals getoond in **figuur 1**. Er zijn verschillende manieren en snelkoppelingen om componenten aan de schakeling toe te voegen (zie het kader **Waar zijn de componenten?**). Voorlopig gebruiken we de componentenbalk aan de bovenkant van het scherm (**figuur 2**).

Klik kort met de linker muisknop op het weerstandssymbool en verplaats de muiswijzer naar het lege werkblad (klik eventueel op het pijltje of driehoekje rechts van de symboolknop voor een andere oriëntatie). Er verschijnt een weerstandssymbool. Klik nogmaals met de linker muisknop om de weerstand op de gewenste positie in het werkblad te plaatsen. Voer in het venster dat nu wordt geopend een waarde in van **100** in het veld **Value** bovenaan. Sluit het venster met de knop **OK** of door resoluut op de Enter-toets te drukken.

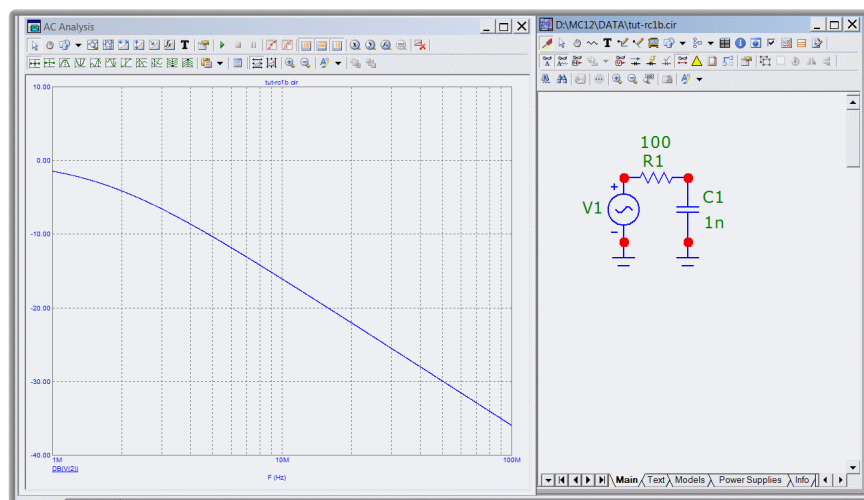
Herhaal deze procedure voor een condensator; vul voor diens waarde **1n** in. Als je de condensator zo plaatst dat een van zijn aansluitingen een van de aansluitingen van de weerstand raakt, worden ze automatisch met elkaar verbonden.

Vervolgens voegen we een sinusgenerator of -bron toe door op **Component** in het hoofdmenu te klikken en vervolgens **Analog Primitives Waveform Sources Sine Source** te selecteren. Kies **GENERAL** in de lijst rechts van het eigenschappen-venster van de bron en klik op **OK**.

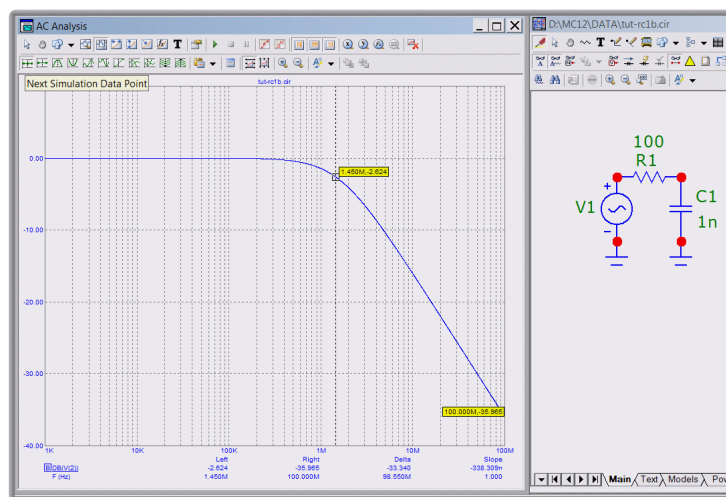
Maak de schakeling af door de massasymbolen uit de componentenbalk toe te voegen aan de condensator en de signaalbron.



Figuur 2. Micro-Cap wordt geleverd met een onderdelenbalk.



Figuur 3. Als je op het RC-knooppunt klikt, verschijnt de overdrachtcurve van het filter.



Figuur 4. Pas de grenzen van de grafiek aan om te zien wat je wilt zien.

Als je de componenten verder uit elkaar hebt geplaatst, moet je ze nog met elkaar verbinden. Dat kan met de knop **Wire Mode** of door op **Ctrl-W** te drukken. Om een draad te tekenen, klik je op het beginpunt van de draad en sleep je, terwijl je de linker muisknop ingedrukt houdt, de muiswijzer naar het eindpunt van de draad.

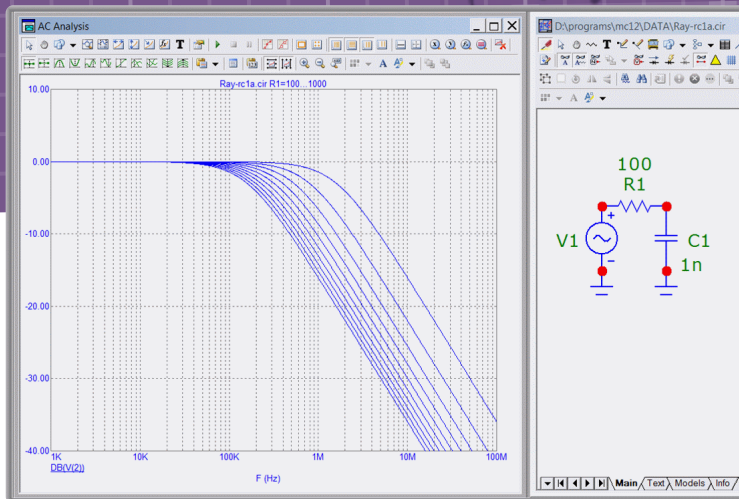
Simuleren maar!

We zijn nu klaar om een simulatie uit te voeren. In het menu **Analysis** kies je **Probe AC...** wat je naar de analysemodus brengt, klaar om te 'meten'. In deze modus kun je het getekende schema niet meer veranderen, alleen de componentwaarden. De rode

stippen in de schakeling zijn de knooppunten. Door op een knooppunt te klikken, verschijnt een grafiek in het venster **AC Analysis**, zoals in **figuur 3**.

Dit is een grafiek met de standaard frequentie-instellingen, maar we willen een grafiek die begint bij een lagere frequentie. Om dit te bereiken, kies je **Limits...** uit het **Probe**-menu. Verander de **Frequency Range** in '100Meg, 1k' (het formaat is 'hoog, laag') en klik op **Close** om af te sluiten. De grafiek verandert in die van **figuur 4**. We zien nu zowel de doorlaatband van het filter (DC tot 1 MHz) als de stopband (alles boven 1 MHz).

Je kunt de datapunten in de grafiek aflezen met behulp van een cursor. Klik linksboven op



Figuur 5. Door de waarde een parameter (hier de waarde van R1) stapsgewijs te veranderen, kan de invloed ervan op de overdrachtsfunctie van de schakeling zichtbaar worden gemaakt.

de knop *Next Simulation Data Point* boven de grafiek en er verschijnt een cursor die je kunt verplaatsen.

Simulatie versus werkelijkheid

Volgens de grafieken in de figuren 3 en 4 dempt ons filter kennelijk alles boven 1 MHz. Dat zou betekenen dat dit filter alle frequenties tot oneindig onderdrukt. We weten dat dit in werkelijkheid niet waar is. Waarom is het dan wel zo in de simulatie?

De reden is dat een simulator ideale componenten veronderstelt. Als je het werkelijke gedrag van een schakeling wilt benaderen, moet je de onvolkomenheden van zowel de gebruikte componenten als van de echte wereld toevoegen door verdwaalde parasitaire elementen 'in te bouwen'. Zie het kader **Beperkingen van componentmodellen** voor details over hoe je je simulatie meer op een echte schakeling kunt laten lijken.

Een componentwaarde wijzigen in de analysemodus

In de analysemodus kun je de waarden van de componenten nog wijzigen. Daartoe moet je overschakelen naar de selectiemodus door op de knop met de muiswijzer linksboven in het schakelingvenster te klikken (je kunt ook Ctrl-E intoetsen). Nu kun je op een component(-waarde) klikken en deze op dezelfde manier wijzigen als voorheen. De simulatieresultaten worden automatisch bijgewerkt. Als je ergens anders in de schakeling wilt meten, moet je terugkeren naar de *Probe*-modus door op de *Probe*-knop te klikken.

Een parameterwaarde stapsgewijs veranderen

Een krachtige eigenschap van een simulator is dat de waarde van een of meer parameters stapsgewijs kunt veranderen en de resultaten kan weergeven in de vorm van een reeks

Hoeveel is 'M'?

In SPICE-taal staat M voor milli (10^{-3}), dus als je een weerstandswaarde instelt op 1 Mohm, krijg je 0,001 Ω ... Je kunt MEG gebruiken voor mega of 10^6 .

grafieken. Laten we bijvoorbeeld de waarde van weerstand R1 stapsgewijs veranderen alsof het een potentiometer is, om te zien wat het afstembereik is voor een bepaalde sweep. We zullen de afsnijfrequentie van het filter 'sweepen' door R1 in stapjes van 100 Ω te veranderen van 100 Ω naar 1000 Ω . Selecteer daartoe *Stepping...* in het *Probe*-menu. Selecteer R1 in het *Step What*-vak. Voer waarden in voor de velden *From*, *To* en *Step Value* en vergeet niet *Step It* op *Yes* te zetten. Klik op *OK* en vervolgens op de knop *Run* (dat is die met de groene pijl of driehoek; of druk op F2) in het venster AC Analysis om de grafiek weer te geven als functie van de weerstandswaarde (figuur 5).

In dit voorbeeld zijn de stappen even groot (lineair), maar je kunt ook exponentieel stappen. Het is mogelijk om door meerdere componentwaarden tegelijk of in geneste sweeps te stappen. Je kunt ook door de modelwaarden van componenten stappen, bijvoorbeeld de versterkingsfactor van een transistor.

Transiëntanalyse

Je moet de juiste analysemodus kiezen om niet-lineair gedrag te zien (zie het kader **Soorten analyse**). Om niet-lineair gedrag te demonstreren, gebruiken we een simpel vervormingseffect: een diodeclipper zoals in figuur 6. We kunnen dan transiëntanalyse gebruiken om vervorming te zien alsof we een oscilloscoop gebruiken.

Stel de frequentie van de sinusbron, V1, in op 1 kHz (in het vak 'F' rechtsonder in het eigenschappen-venster) en kies *Probe Transient...* uit het menu *Analysis*. Kies in het *Probe*-menu *Limits...* en stel *Maximum Run Time* in op 2m en *Maximum Time Step* op 2u.

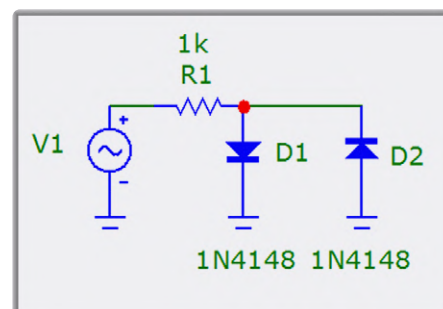
Verschillende analyses

Na het tekenen van de schakeling wil je gaan meten. Hiervoor moet je een analysemodus kiezen. In SPICE-taal heet meten met een oscilloscoop 'transiëntanalyse', terwijl meten met een spectrumanalyzer 'AC-analyse' wordt genoemd. De verbindingpunten van de componenten heten knooppunten. In de analysemodus kun je de spanningen op deze knooppunten en de stromen erdoorheen meten.

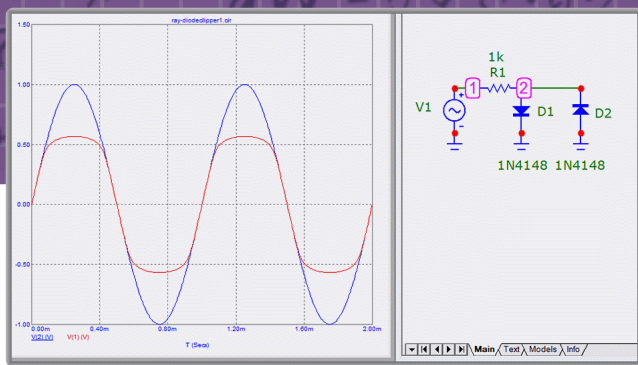
Voor het visualiseren van niet-lineair gedrag (zoals vervorming en clipping) moet je de transiëntanalyse gebruiken. De AC-analyse gaat altijd uit van lineair gedrag en geeft de kleinsignaal-frequentierespons van schakelingen zoals filters en versterkers.

In beide analysemodi kun je ook de DC-spanningen en -stromen zien. Er zijn nog meer modi mogelijk, zoals ruis- en impedantieanalyse.

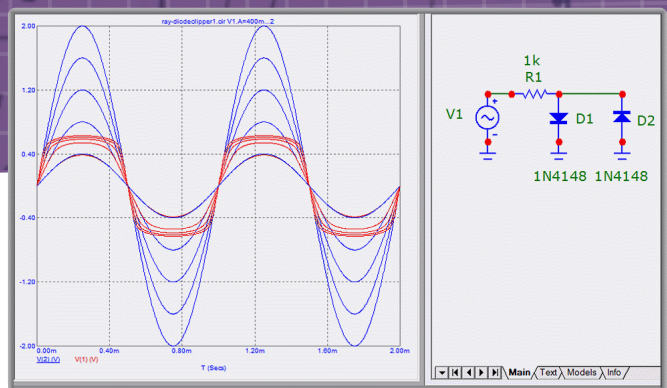
Wanneer de schakeling correct is getekend (zonder 'zwevende' knooppunten) wordt een analysetype geselecteerd. Vervolgens wordt een berekening uitgevoerd over een door de gebruiker gedefinieerde periode of frequentiebereik met een geselecteerde resolutie. Het instellen van deze resolutie is een afweging tussen de gladheid van de curve en de rekentijd. Als de berekening klaar is, kun je in het circuit gaan 'meten'.



Figuur 6. Deze eenvoudige schakeling vertoont niet-lineair gedrag.



Figuur 7. Een transientanalyse van de schakeling van figuur 6 levert deze signalen op.



Figuur 8. De uitgangsverzadiging neemt geleidelijk toe naarmate de amplitude van het ingangssignaal toeneemt.

Het is een goede gewoonte om de stapgrootte in te stellen op 1/1000 van de looptijd. Meet nu weer in de schakeling, zowel aan de ingang (de sinus), als aan de uitgang (de afgeknotte golfvorm).

Amplitude-sweep

Zoals eerder uitgelegd, kun je door een reeks componentwaarden of modelparameters stappen. Laten we dat eens doen met de amplitude van de sinusbron. Kies **Stepping...** uit het **Probe**-menu. Kies de V1-amplitude 'A' en ga van **400m** naar **2V** in stapjes van 400 mV (vergeet niet **Step It** op **Yes** te zetten).

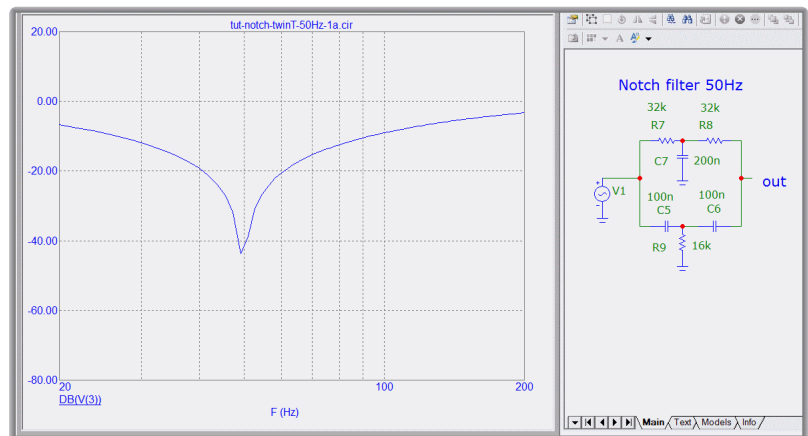
Figuur 8 toont de resultaten.

Aangezien de schakeling van **figuur 6** bedoeld is voor een geluidseffect, zou het nuttig zijn om te horen hoe het klinkt. Micro-Cap biedt de mogelijkheid de golfvorm af te spelen via je geluidskaart of op te slaan als WAV- of CSV-bestand.

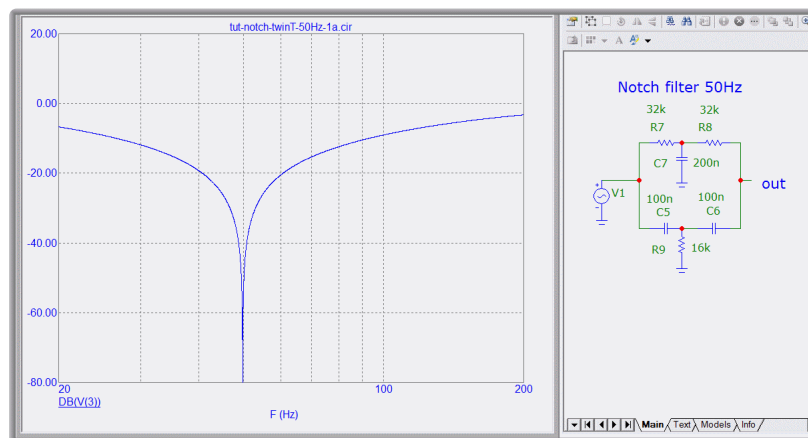
De duur van het geluid wordt bepaald door de probe-grenzen. Verander **Maximum Run Time** in **500m** (en **Maximum Time Step** in **50u**) voor een halve seconde geluid. Dubbelklik vervolgens op het trace-venster om de eigenschappen ervan te openen en selecteer het tabblad **Save Curves**. Selecteer de golf die je wilt horen in de **Curves**-lijst. Indien je stepping hebt ingeschakeld (zoals het stapsgewijs veranderen van de amplitude van het ingangssignaal), kun je elke stap beluisteren door deze te selecteren in het vak **What To Save** en vervolgens op **Play** te drukken.

Waar zijn de componenten?

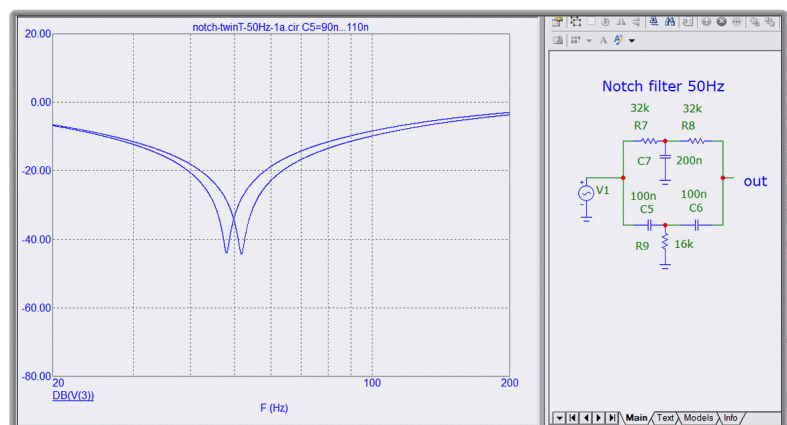
Alle componenten zijn te vinden via het tabblad **Components** en in de uitklapbare bibliotheeklijst (het 'Panel'), maar er zijn ook enkele sneltoetsen. Bovenaan het scherm is de componentenbalk beschikbaar, maar nog sneller zijn toetsen als 'R' voor een weerstand en 'C' voor een condensator. Bovendien slaat het programma de recent gebruikte componenten op in het startvenster van het tabblad **Components**.



Figuur 9. De simulatieresolutie is van belang. Hier is **Probe Limits** ingesteld op 100 punten.



Figuur 10. Dezelfde simulatie als in figuur 9, maar nu met **Probe Limits** ingesteld op 10.000 punten.



Figuur 11. Dezelfde schakeling als in figuur 9, maar nu is C5 niet identiek aan C6.

Resolutie van de simulatie

Verkeerde resolutie-instellingen kunnen details in het resultaat van een simulatie beïnvloeden of zelfs onzichtbaar maken. Een notchfilter heeft bijvoorbeeld een scherpe dip bij de frequentie waarvoor het bedoeld is. Bij een te geringe frequentieresolutie lijkt de dip minder diep dan hij in werkelijkheid is. **Figuur 9** en **figuur 10** tonen de resultaten van hetzelfde twin-T notchfilter, dat tweemaal is gesimuleerd met verschillende resolutie-instellingen.

Ideale componenten

Dit filter heeft met de gegeven onderdelenwaarden een oneindig diepe dip (notch) bij 50 Hz. Maar voordat je dit 'perfecte' filter bouwt, moet je even nadenken over de componentwaarden. In een simulator zullen de twee condensatoren van 100 nF zowel identiek als exact 100 nF zijn, en krijg je een oneindig diep 50 Hz notchfilter zoals hier getoond.

Verandering van de waarde van bijvoorbeeld C5 met plus of min 10% heeft een grote invloed (**figuur 11**). De dip is nu slechts 40 dB diep en het filter is bovendien ontstemd, wat resulteert in slechts 35 dB damping bij 50 Hz. Dit laat zien hoe een simulator je kan helpen te voorspellen hoe de toleranties van componenten in de praktijk uitpakken in je schakeling – voordat je die opbouwt.

Voor nog realistischer simulaties moet je daarnaast rekening houden met andere parasitaire parameters, zoals de lengte van aansluitdraden en printsporen (zie het kader).

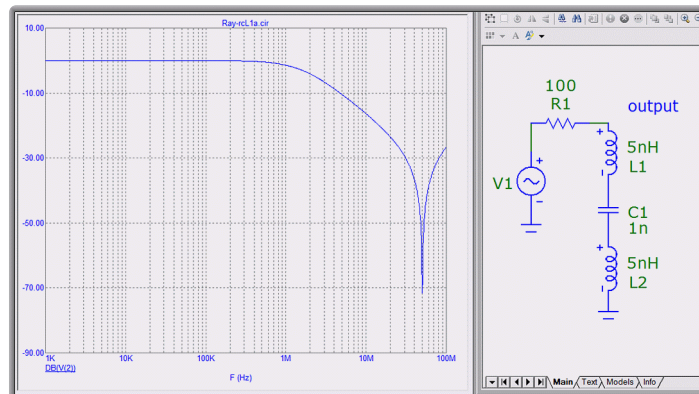
Hiermee houdt het nog lang niet op

Dit artikel heeft slechts een tipje opgelicht van de sluier die over deze complexe materie ligt. Zoals al opgemerkt, moet een simulator deel uitmaken van je gereedschapskist, net als een multimeter en een oscilloscoop. Houd echter altijd in gedachten dat een simulator een vereenvoudigde versie van de werkelijkheid toont. Daarom moet je voor correcte resultaten met een simulator (Micro-Cap of welke andere dan ook) enig idee hebben van de praktische aspecten van het bouwen van een schakeling. Weten hoe de simulator moet worden ingesteld en wat de beperkingen ervan zijn, is een eerste vereiste. Simulatie combineren met het bouwen van een echt prototype is nog steeds een goed idee.

Beperkingen van componentmodellen

Een simulatie is slechts zo goed als de modellen die worden gebruikt. Alle componenten in ons filter zijn ideaal. In de praktijk is dat natuurlijk niet het geval. Voor ons eenvoudige laagdoorlaatfilter is de belangrijkste beperking de serie-inductie van de condensator aansluitingen. Elke millimeter draad voegt een inductie van ongeveer 1 nH toe, waardoor de impedantie van de condensator boven een bepaalde frequentie gaat toenemen en het filtereffect afneemt.

Laten we aannemen dat je condensator aansluitingen van 5 mm heeft, wat we simuleren door twee spoelen van 5 nH in serie met de condensator aansluitingen op te nemen (wat hetzelfde is als het toevoegen van één serie-inductie van 10 nH).



We zien dat het filter een scherpe dip heeft rond 50 MHz en dan vanaf 50 MHz begint te 'leken'. Dit wordt veroorzaakt door de resonante LC-schakeling. Als je RF-storingen tot 1 GHz wilt filteren is dit een probleem, maar in speciale gevallen kun je gebruik maken van dit effect. Als je bijvoorbeeld een bemonsterings-klokfrequentie bij 50 MHz wilt filteren, dan is dit een verbetering, maar moet je wel een beetje met de waarde van de inductie 'spelen'.

Het bovenstaande is een voorbeeld van hoe een simulator je kan helpen een schakeling op de juiste manier op te bouwen.

Vraag: als draden inductie toevoegen, levert het gebruik van SMD-condensatoren dan een filter zonder parasitaire inductie op? Nee, maar het komt in de buurt. De condensator heeft nog steeds een eindige lengte (2 mm is 2 nH) en zijn printspoor naar massa voegt ook zelfinductie toe. Via's kunnen 0,3...1 nH toevoegen, afhankelijk van hoe je ze definieert en plaatst.

Gooi je breadboard met zijn slechte contacten dus nog niet weg (hoe modelleren je die eigenlijk?).

220336-03

Vragen of opmerkingen?

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



Gerelateerde producten

> Joy-IT ScopeMega50 USB Oscilloscope (SKU 18277)
www.elektor.nl/18277

> Gilles Brocard, *The LTspice XVII Simulator* (SKU 19741)
www.elektor.nl/19741

WEB LINK

[1] Micro-Cap website: <https://www.spectrum-soft.com/download/download.shtm>



PAUL

Award 2022

jonge technische talenten en hun creatieve ontwerpen

Florian Schäffer (voor Elektor)

Het bevorderen van jong elektronicatalent is een belangrijke missie, en dat geldt ook voor het Duitse Fachverband Elektronikdesign und -fertigung (FED). Als onderdeel van de FED-conferentie in het Duitse Potsdam werd in 2022 opnieuw de PAUL Award ter waarde van 6000 euro uitgereikt. Iedereen tussen de 15 en 25 jaar mocht een creatief elektronica-ontwerp indienen. Er werden drie winnende teams gekozen, die we hier presenteren.



Groepsfoto met de genomineerden en winnaars van de PAUL Award 2022 in Potsdam (bron: FED e.V.).

Stekkerdoos met brandbeveiliging

Het winnende team *Fisego* ontwikkelde een stekkerdoos met ingebouwde temperatuurbeveiliging met een microcontroller om brand door overbelasting te voorkomen. De jury beloonde het idee, de uitvoering en de documentatie met de eerste prijs van 3000 euro. Daarnaast namen Sophia Reiter (een 22-jarige studente elektrotechniek) en Fabian Goedert (een 25-jarige student civiele techniek) de mediaprijs in ontvangst, die onafhankelijk door Elektor werd toegekend en tijdens de ceremonie in Potsdam aan hen werd overhandigd.



Jurylid Jürgen Deutschmann overhandigt de eerste prijs voor de brandveilige multifunctionele stekkerdoos aan Sophia Reiter en Johannes Steube (bron: FED e.V.).

In het prototype van de 3D-geprinte stekkerdoos bewaakt een schakeling met een ATmega328, een ACS712 hall-sensor en een DS18B20-temperatuursensor het stroomverbruik en de temperatuur in de doos. Op een OLED-display worden de parameters weergegeven met drie waarschuwingsniveaus. Wanneer een grenswaarde wordt overschreden, wordt eerst een waarschuwing gegeven met mogelijke tegenmaatregelen, voordat de stekkerdoos automatisch uitschakelt bij het bereiken van de volgende grenswaarde. In tegenstelling tot stekkerdozen met thermische zekeringen of automatische stroomonderbrekers, kan de gebruiker bij deze contactdoos ingrijpen voordat de doos uitschakelt door bijvoorbeeld een van de verbruikers uit te schakelen.

Volgens Sophia Reiter was het zelfgestelde doel van de beide studenten een stekkerdoos voor het Internet of Things (IoT) met een ingebouwd brandbeveiligingssysteem, die niet alleen in de particuliere sector kan

worden gebruikt, maar ook van nut kan zijn voor gebruikers in de industrie bij het bewaken van machines of fabrieken. Het onlangs opgerichte bedrijf heeft al een verdere stap in de richting van dit doel gezet door prototypes van de stekkerdoos in spuitgiettechniek te produceren.

FED erkent gebrek aan jong talent en stimuleert nieuwe ideeën

De PAUL Award werd voor het eerst georganiseerd in 2020, maar moest toen vanwege COVID een pauze inlassen, zodat de tweede prijsuitreiking pas op 28 september 2022 kon plaatsvinden. Zoals Jürgen Braunsteiner, nieuw bestuurslid van de FED, in zijn inleidende toespraak voor ongeveer 100 gasten benadrukte, heeft de sector te maken met een generatiewisseling, waarbij vooral kleine en middelgrote bedrijven worstelen met problemen bij het aantrekken van jong talent.

Het stimuleren van jonge mensen met

creatieve ideeën is daarom een belangrijk doel: ze moeten via de PAUL Award worden gemotiveerd om hun projecten te ontwikkelen, en moeten een kans krijgen om deze te presenteren en contacten te leggen met de industrie.

Als positief voorbeeld noemde Braunsteiner de jonge Australische Cynthia Sin Nga Lam. Haar H2prO-project reinigt afvalwater met behulp van fotokatalyse en wekt tegelijkertijd elektriciteit op. Dit apparaat dat het potentieel heeft om de wereld te veranderen, zorgde in 2014 voor heel wat opschudding in de media. Het project toonde aan dat nieuwe ideeën overal te vinden zijn, als je er maar zelf aan durft te werken om ze met eenvoudige middelen te realiseren.

Energy harvesting met Peltier-element

Een kleine deelnemersgroep van vijf kandidaten is de uitdaging van energy harvesting aangegaan: de projecten moesten zonder externe energiebron functioneren – een



Team Fisego met de speciale prijs van Elektor (bron: FED e.V.).



De tweede plaats ging naar Manuel Wilke voor zijn accu-management-systeem (bron: FED e.V.).

taak die werd opgelost door het team *Drink Safe*: ze ontwikkelden een onderzetter die sterk doet denken aan gadgets die kopjes via USB warm houden, maar die een heel andere taak heeft: de onderzetter meet de temperatuur van het kopje en signaleert via een LED wanneer het drankje voldoende is afgekoeld om er zonder verbrandingsgevaar van te kunnen genieten. Gezien waarschuwingen als “Let op, hete drank” op wegwerpbekers, is dit geen bijzonder vergezocht idee.

Voor de werking maakt het project gebruik van een Peltier-element dat een stroom opwekt wanneer er een temperatuurverschil is tussen zijn twee zijden. Omdat de bij dit proces opgewekte stroom zeer klein is, is een geïntegreerde step-up regelaar nodig om de LED aan te sturen. Op deze manier is er geen aparte stroombron zoals een batterij nodig. Het idee en de uitvoering leverden het team de derde plaats op.

Weerstation en accu-managementsysteem

Het idee van een ander team om een waterrad in een regenpijp te installeren en daarmee een zichzelf voedend IoT-weerstation te construeren, kon de jury helaas niet overtuigen, wat misschien ook te maken had met het feit dat het project nog met kinderziektes worstelde. Na afloop konden de twee studenten niet zeggen of ze zich hierdoor zouden laten weerhouden om hun werk voort te zetten of dat ze een nieuw idee zouden gaan ontwikkelen.

De tweede prijs (2000 euro) werd toegekend aan Manuel Wilke, vijfdejaars student aan de Berlijnse Universiteit voor Toegepaste Wetenschappen (BHT), die het *nandomBMS* accu-managementsysteem (BMS) ontwikkelde. Het ontwerp is gebaseerd op een XC4000 Cortex-M4 microcontroller en een

LTC6813 load balancer met BMS van Analog Devices en kan tot 18 in serie geschakelde cellen monitoren. De microcontroller kan met de buitenwereld communiceren via CAN. De print zal voornamelijk worden gebruikt in een eigen E-Buggy en is voorzien van galvanische scheiding tussen hoog- en laagspanning.

De FED PAUL Award in 2023

Dit jaar zouden de organisatoren vooral meer deelnemers willen zien, wat onder andere bereikt moet worden door meer PR-inspanning. Daarnaast zullen er geen extra technische eisen meer worden gesteld. Alleen de kosten moeten binnen de zakgeldgrenzen blijven, waardoor de focus waarschijnlijk meer op schoolgroepen en de maker-scene komt te liggen. Wie individueel of als team wil deelnemen, kan zijn aanvraag nu indienen en er dan tot 30 september 2023 aan werken. Alle informatie is te vinden op de PAUL Award-website [1]. ◀

Vertaling: W. den Hollander – 220599-03



De winnende PAUL Award-projecten werden tentoongesteld op de FED-conferentie: een nieuw prototype van de multifunctionele stekkerdoos, de kale print van het accu-managementsysteem en het Drink Safe-apparaat.



Team BMK, dat een prijs won voor zijn project Drink Safe, eindigde op de derde plaats (bron: FED e.V.).

WEB LINKS

[1] PAUL Award: <https://www.paul-award.de>



Mijn eerste Software Defined Radio

in minder dan 15 minuten klaar

Clemens Valens (Elektor Labs)

Wist je dat je een FM radio-ontvanger kunt bouwen door gewoon een paar blokken en lijnen te tekenen op een virtueel doek? Na het lezen van dit artikel wordt tekenen en schilderen gerarandeerd je nieuwe hobby!



Figuur 1. Het in dit artikel gebruikte SDR-front-end is de HackRF One van Great Scott Gadgets.

Software-gedefinieerde radio (Software Defined Radio, SDR) gaat over het moduleren en demoduleren van radio-signalen in software. In plaats van speciale elektronische schakelingen, zoals mixers of demodulators, gebruikt SDR software. Natuurlijk is er wat hardware nodig om analoge radiosignalen om te zetten in digitale en omgekeerd, maar dat is dan ook alles wat je nodig hebt. Om het eenvoudig te houden, beperken we ons in dit artikel tot de ontvangst van radiosignalen, maar het meeste van het gepresenteerde materiaal geldt ook voor zenden. SDR kan in beide richtingen werken.

SDR kan alles

SDR biedt vele voordelen ten opzichte van analoge radio. Ten eerste begrijpt SDR elke modulatietechniek die je maar kunt bedenken, en meer. Dit betekent dat je met SDR niet alleen AM- en FM-radio kunt beluisteren, maar ook WiFi of Bluetooth kunt gebruiken, of digitale TV en DAB+ kunt decoderen, of met HAM-radio kan spelen – en nog veel meer.

SDR kan dit omdat het de modulatie- of demodulatietechniek in software implementeert, en software is gemakkelijk te veranderen. Omdat alles in software gebeurt, kan SDR ook dingen doen die met alleen elektronische componenten zeer moeilijk of zelfs onmogelijk zijn.

Past altijd

Een tweede voordeel van SDR is dat je voor al dit lekkers slechts één stuk hardware nodig hebt. Je hebt geen aparte FM-ontvanger en WiFi-ontvanger en digitale TV-ontvan-

ger en wat al niet meer nodig. SDR gebruikt hetzelfde front-end voor alles.

Dit front-end is een analoog/digitaal-converter (ADC) met een antenne die analoge radiosignalen omzet in digitale signalen die met een computer kunnen worden verwerkt. Voor het zenden is een digitaal/analoog-converter (DAC) nodig. In werkelijkheid is dit front-end méér dan alleen een ADC en/of DAC. Het voert ook enige menging uit om het gewenste RF-sigitaal in het bereik te brengen dat kan worden bemonsterd (en omgekeerd bij zenden). De basisfunctie blijft echter hetzelfde.

SDR-front-ends zijn er in soorten en maten; je moet er een kiezen die bij je budget en wensen past. Hoe duurder de converter, hoe meer je ermee kunt doen. Hier gebruiken we de HackRF One van Great Scott Gadgets [1] (figuur 1). Dit is niet het goedkoopste SDR front-end, maar het ondersteunt ontvangen en zenden, en werkt tot 6 GHz. Ook en niet onbelangrijk wordt het ondersteund door vele SDR-softwareplatforms zoals GnuRadio [2], SDR Sharp (SDR#) [3] en SDR++ [4]. Voor het hieronder beschreven project zullen andere (goedkopere) SDR-front-ends waarschijnlijk even goed werken, zolang ze in de FM-radio band (87,5...108 MHz) kunnen functioneren.

Open source en hardware

Een ander leuk aspect van SDR is dat veel ervan open source en open hardware is, gemaakt door een grote

gemeenschap van enthousiastelingen. Zij ontwerpen en publiceren schema's voor front-ends en ontwikkelen softwaretools en algoritmen voor SDR-toepassingen. HackRF One is ook open source, en de ontwerpbestanden zijn beschikbaar op GitHub [5].

Vereisten

Zoals al opgemerkt, heb ik voor dit SDR-project HackRF One gebruikt als front-end, in combinatie met een telescoopantenne van 88 cm. We zullen GNU Radio Companion gebruiken voor het softwaredeel. Eigenlijk gebruiken we GNU Radio Companion oftewel GRC, de grafische interface voor GNU Radio. GNU Radio is beschikbaar voor Linux, Windows en macOS, dus de kans is groot dat jouw computer het ook kan draaien. Het draait zelfs op een Raspberry Pi.

Programmeren = slepen en neerzetten!

GRC is een grafisch programmeertool waarmee je een radiotoepassing kunt maken zonder echt te programmeren. Je sleept gewoon blokken op een 'doek' die je onderling verbindt met virtuele draden. De programmering blijft beperkt tot het configureren van de blokken. GRC wordt geleverd met een grote bibliotheek met blokken voor het bouwen van allerlei modulatie- en demodulatieschema's. We beginnen met een leeg doek. Nou ja – niet helemaal leeg, want het bevat al twee blokken: een *Options*-blok en een *Variable*-blok. Dubbelklikken met de linker muisknop opent de eigenschappen van een blok, zodat je de parameters ervan kunt wijzigen. Met het *Options*-blok kun je enkele low-level opties definiëren, en je kunt een naam voor het doek instellen – zie **figuur 2**. We laten overal de standaardwaarden staan.

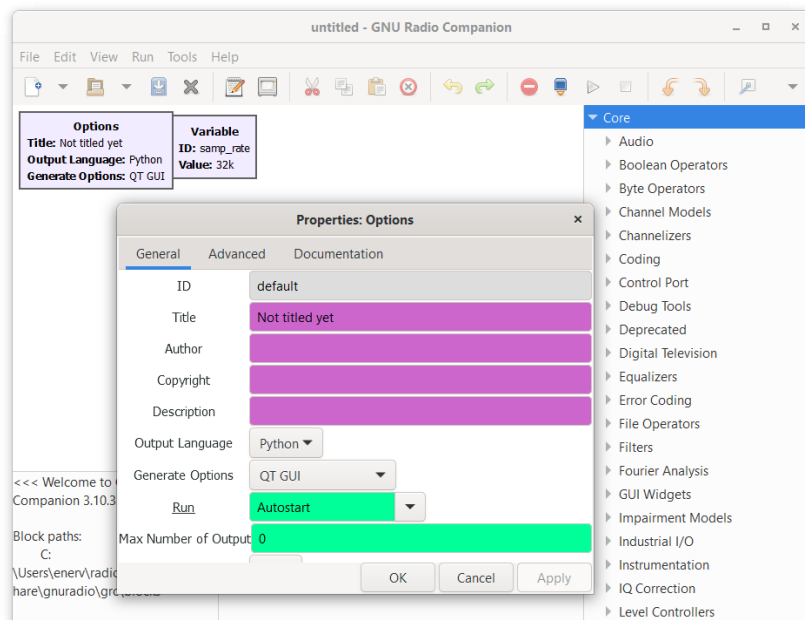
Globale bemonsteringsfrequentie

Het blok *Variable* definieert een globale sample rate-variabele die `samp_rate` heet. Deze wordt gebruikt in andere blokken. Strikt genomen is het niet nodig, maar het is praktisch, en dus laten we het zo. De waarde is echter te laag voor het bemonsteren van FM-signalen; daarom veranderen we hem in 10 miljoen. De maximale waarde voor HackRF One is 20 miljoen. Als je een te hoge waarde neemt, kan je computer het moeilijk bijbenen.

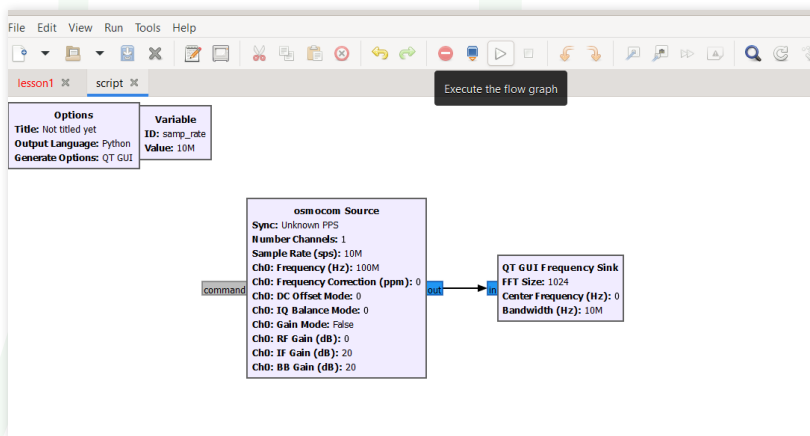
Voeg een RF-source toe

Nu hebben we RF-invoer nodig. In GRC wordt dit een *source* (bron) genoemd. Er is geen speciaal source-blok voor de HackRF One. De *Soapy*-sectie heeft een Soapy HackRF Source, maar die werkte bij mij niet. De *osmocom*-source werkte wel. Het OsmoSDR front-end bestaat niet meer, maar het blok ondersteunt ook andere SDR-hardware, waaronder HackRF One.

Men een dubbelklik open je het eigenschappen-venster, en we zien dat de sample rate is ingesteld op `samp_rate`, de naam van het variable-blok dat GRC ons gaf. We hoeven hier maar één parameter te veranderen, en dat



Figuur 2. Een nieuw project in GNU Radio Companion (GRC) wordt gestart met twee blokken. Dubbelklikken op een blok opent de eigenschappen ervan.



is de RF-versterking van kanaal 0, die te hoog is voor FM-radiosignalen die nogal sterk zijn. We kunnen deze op nul zetten.

Spectrumanalyzer

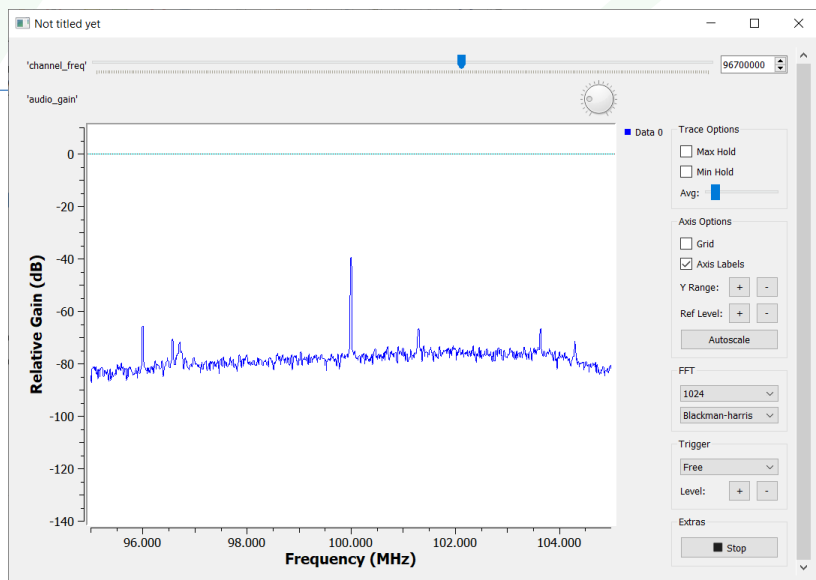
Om snel te kijken of alles werkt, kunnen we een spectrum-analyzer toevoegen aan het doek en die aansluiten op de uitgang van het blok RF-source. Het gaat om de QT GUI Frequency Sink en die is te vinden in de *Instrumentation*-sectie van GRC. De nomenclatuur van deze blokken kan een beetje verwarrend zijn, en dat maakt ze soms moeilijk te vinden.

Om de frequentie-ingang te verbinden met de RF-source, klik je op de ingang en vervolgens op de uitgang van de source. Andersom gaat ook – net als tekenen met ingedrukte linkermuisknop. Merk op dat je alleen in- en uitgangen van dezelfde kleur met elkaar kunt verbinden, blauw in dit geval.

Eerste poging

Je kunt nu op de knop *Play* (die met het driehoekje erop) klikken om het flowdiagram uit te voeren (**figuur 3**). Je moet natuurlijk, als je dat nog niet hebt gedaan, je ontwerp opslaan voordat je verder kunt gaan. Als het goed is,

Figuur 3. Klik op de play-knop om je eerste flowdiagram uit te voeren.



▲
Figuur 4. Na activeren van het Control Panel van de QT GUI Frequency Sink heb je toegang tot verschillende regelaars.

wordt een frequentieplot zichtbaar. Een beetje 'spelen' met de antenne zou iets andere resultaten moeten opleveren, maar dat is misschien moeilijk te zien. Voor een duidelijker beeld moeten we de frequentieplot configureren. Klik op de **Stop**-knop (die met het vierkantje) om de flow-grafiek te stoppen. Dubbelklik op het frequentieblok om de eigenschappen ervan te openen en stel **averaging** (middeling) in op gemiddeld. Stel de middenfrequentie in op dezelfde waarde als de frequentie van Channel 0 van de osmocom-source, in ons geval 100 MHz. Klik vervolgens op het tabblad **Config** en zet **Control Panel** op **Yes**. Sluit het eigenschappenvenster en klik weer op **Play**. Nu heeft de frequentieplot een bedieningspaneel en kun je het vakje **Max Hold** aanvinken (figuur 4). Het spelen met de antenne zou nu beter zichtbaar moeten worden en als dat zo is, weet je dat je RF-signalen kunt ontvangen.

Het is allemaal plug&play!

Het leuke is dat we niets hoefden te doen om het flowdiagram met HackRF One te laten communiceren. Er hoeven geen drivers te worden geïnstalleerd, geen poorten geselecteerd, het is allemaal plug&play.

Variabelen zijn handig

We hoefden alleen de middenfrequentie van de frequentiesink in te stellen op dezelfde waarde als die van de RF-bron, wat betekent dat deze op ten minste twee plaatsen nodig is. Als we deze vervangen door een

globale variabele, zoals GRC deed voor de sample rate, hoeven we hem maar één maal in te stellen.

Kopieer het **sample rate** variabeleblok, open de eigenschappen ervan, en verander de ID in **center_freq**. Stel de waarde in op 100 MHz, stel de Channel 0-frequentie van de osmocom-source in op **center_freq**, en stel de middenfrequentie van de frequency sink in op **center_freq**. De twee parameters zijn nu met elkaar verbonden. Als we de waarde van de variabele **center_freq** veranderen, zal deze ook veranderen in alle andere blokken die er gebruik van maken.

Mengen

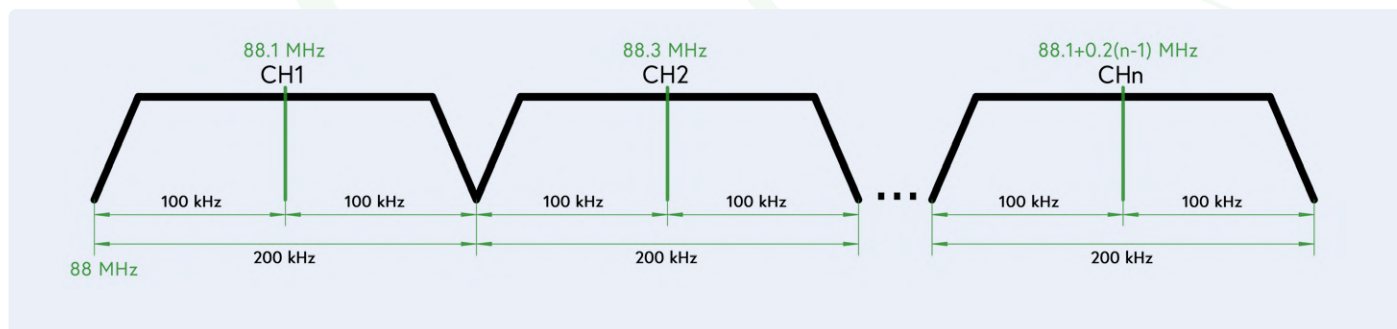
Om iets te bereiken met SDR moet je natuurlijk wel enig idee hebben hoe radio's werken. De algoritmen in SDR doen dezelfde dingen als de elektronische schakelingen in een analoge radio. Om een FM-radiozender te ontvangen moeten we er dus op afstemmen, het signaal demoduleren en filteren en het hoorbaar maken.

Afstemming kan plaatsvinden met een zogenaamde mixer of frequentievermenigvuldiger. Wanneer dit gebeurt met kwadratuur- of complexe signalen, kunnen hiermee frequenties naar boven en beneden worden verschoven. In GRC komen de blauwe in- en uitgangen overeen met complexe floating-point signalen die geschikt zijn om te worden gemengd.

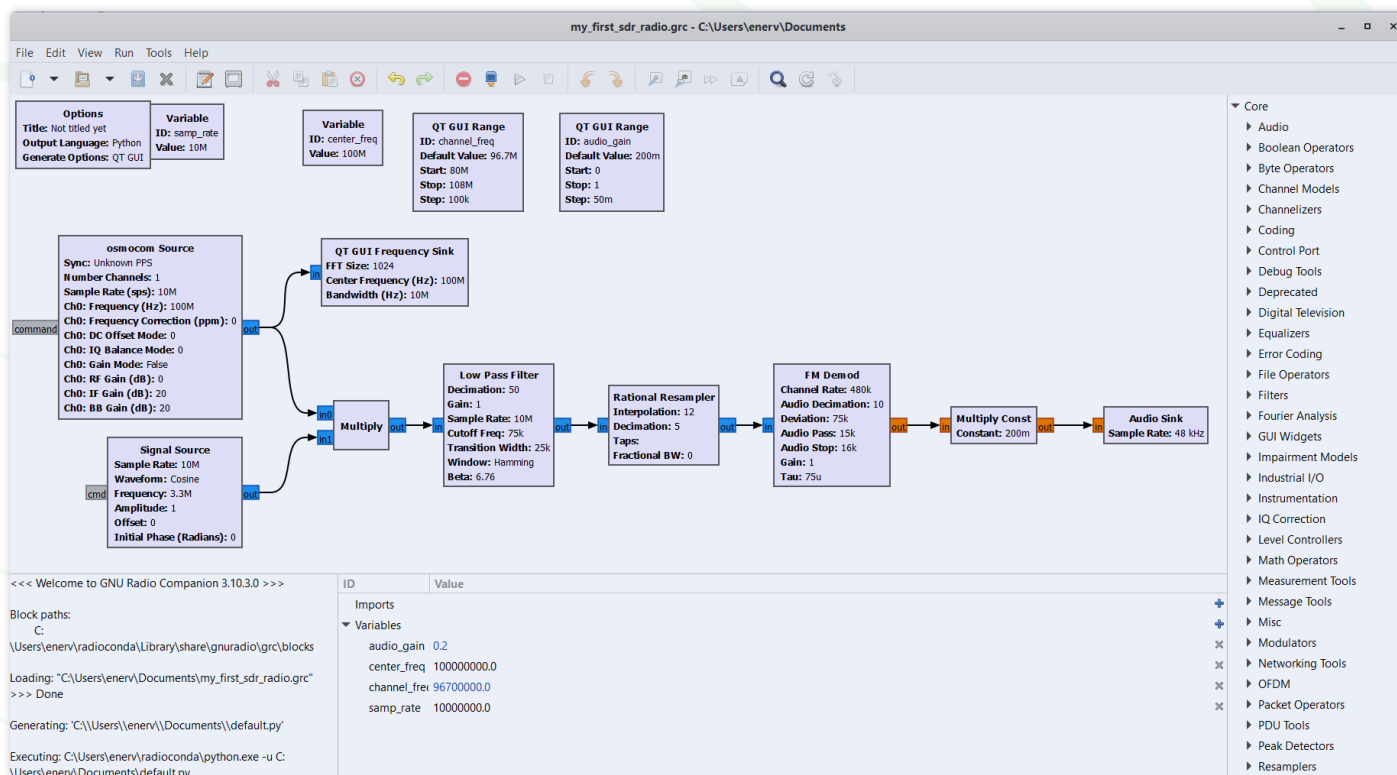
Het doel is om de gewenste frequentie te verschuiven naar het midden van de frequentieplot, waar we hem gemakkelijker uit kunnen halen. Om af te stemmen op een radiostation hebben we dus een vermenigvuldiger nodig en een ander signaal dat de mate van frequentieverschuiving bepaalt. Een **Multiply**-blok is beschikbaar in de sectie **Math Operators**. Voor het afstemsignaal kunnen we het blok **Signal Source** uit de sectie **Waveform Generators** gebruiken.

Afstemmen op een zender

FM-radiozenders hebben een bekende frequentie, die we de kanaalfrequentie zullen noemen. Omdat we deze frequentie gemakkelijk willen kunnen veranderen, maken we er een globale variabele voor aan die we **channel_freq** noemen. Waar ik woon zendt een radiostation op



Figuur 5. De FM-frequentieband is verdeeld in kanalen met een breedte van 200 kHz.



96,7 MHz uit, dus zet ik die waarde in `channel_freq`.

De mixer moet de kanaalfrequentie verschuiven naar de middenfrequentie, dus moet de frequentie van de signaalbron het verschil van de twee zijn. Het verschil tussen 100 MHz en 96,7 MHz is 3,3 MHz en dat is inderdaad de frequentie in het signaalbron-blok.

Omlaag converteren

Met het station op de centrale frequentie moeten we het signaal extraheren en omlaag mengen zodat het kan worden gedemoduleerd. Extraheren betekent filteren en omlaag converteren komt in SDR overeen met het reduceren of decimeren van de sample rate. Het **Low Pass Filter** uit de sectie **Filters** combineert beide bewerkingen in één blok.

De FM-band is verdeeld in kanalen. De breedte verschilt per land, maar ligt meestal rond de 200 kHz. Een radiostation zit in het midden van een kanaal, wat betekent dat het aan weerszijden 100 kHz bandbreedte heeft (figuur 5). De minimale bemonsteringsfrequentie die nodig is voor een signaal van 100 kHz is 200 kHz (Nyquist-Shannon theorema). Onze bemonsteringsfrequentie is 10 MHz, en daarom stellen we de deelfactor van het filterblok in op 10 MHz / 200 kHz oftewel 50. Om het gewenste signaal te extraheren, hebben we geen supersteil filter nodig. Een afsnijfrequentie van 75 kHz met een overgangsbreedte van 25 kHz is prima voor een totale bandbreedte van 100 kHz.

Demoduleren

We hebben nu een FM-gemoduleerd kwadratuursignaal met een sample rate van 200 kHz dat we willen demoduleren. Hiervoor kunnen we het blok **FM Demod** uit de sectie **Modulators** gebruiken. Op de uitgang van de demodulator staat een audiosignaal dat we naar een audio-sink kunnen sturen, in ons geval de geluidskaart van de computer.

Problemen met de sample rate

Het probleem waar we nu tegenaan lopen is sample-rate matching. De audio-sink ondersteunt verschillende bemonsteringsfrequenties die afhankelijk zijn van je geluidskaart. Op mijn computer is de verhouding tussen de mogelijke audiosamplesnelheden en de 200 kHz ingangs-bemonsteringsfrequentie in alle gevallen een niet-geheel getal. Maar de FM demodulator kan alleen delen door een geheel getal. Hoe kunnen we dit probleem oplossen?

De oplossing is om een sample rate-converter in te bouwen die fractionele conversie kan uitvoeren. GRC heeft hiervoor twee opties: een fractionele resampler en een rationele resampler. Aangezien onze in- en uitgangsfrequenties gehele waarden zijn, kunnen we de rationele resampler uit de sectie **Resamplers** gebruiken. Met een fractionele resampler gaat het ook, maar dat kost veel meer rekenkracht.

Voor de beste audiokwaliteit stellen we de audio-bemonsteringsfrequentie in op 48 kHz. Als we dan de demodulator laten werken met bijvoorbeeld 10 keer die frequentie, 480 kHz, hebben we een sample rate-conversie nodig van 200 : 480. Dit kan worden vereenvoudigd met behulp van de grootste gemene deler van beide waarden, die 40 is. 200 : 40 = 5 en 480 : 40 = 12, dus als we eerst omhoog converteren of interpoleren met een factor 12 en dan omlaag converteren met een factor 5, wordt een frequentie van 200 kHz omgezet in een frequentie van 480 kHz. We moeten deze snelheid in het **FM Demod**-blok instellen, samen met een deelfactor van tien, om ervoor te zorgen dat de uitgangsfrequentie overeenkomt met de ingangsfrequentie van de audio-sink.

Bedieningselementen toevoegen

Als je dit flowdiagram uitvoert met de kanaalfrequentie vast ingesteld op een bestaand FM-radiostation in de



Figuur 6. Het flowdiagram van de FM-ontvanger met afstemming en volumeregeling. Als je het uitvoert, ziet het eruit als in figuur 4.

buurt, zou je nu het radioprogramma moeten horen. Is dat niet cool?

We kunnen onze radio flink verbeteren door een afstemmogelijkheid toe te voegen, zodat je gemakkelijk verschillende stations kunt ontvangen. Hiervoor kunnen we een *QT GUI Range* blok gebruiken uit de *GUI Widgets*-sectie. Stel zijn ID in op `channel_freq`. Vul de parameters in en zorg ervoor dat ze allemaal binnen het bereik blijven. Merk op dat het geen zin heeft om een (zeer) kleine stapwaarde in te stellen, aangezien de meeste FM-stations op MHz-frequenties met slechts één decimaal zitten; 100 kHz is waarschijnlijk prima.

Je kunt een heleboel vormen voor de bediening kiezen, maar een teller plus schuifregelaar is het meest praktisch, omdat je dan de frequentie als een numerieke waarde kunt zien. Je kunt ook direct een frequentie intypen. Hernoem of verwijder het oude `channel_freq` variabeleblok, omdat het niet langer nodig is en conflicten met de schuifregelaar veroorzaakt. Je kunt de radio nu afstemmen zoals elke andere radio.

Signaalkleuren

Een andere verbetering is de toevoeging van een volumeregelaar. Die is vergelijkbaar met de afstemknop, en je kunt hem kopiëren en de parameters ervan aanpassen. Hernoem de ID naar `audio_gain`. Kies een *Multiply Const* uit de *Math Operators*-sectie en voeg deze in tussen het *FM Demod*-blok en de audio-sink. Merk op dat de verbindingen rood zijn. Dat komt omdat de input en output van de vermenigvuldiger blauw zijn in plaats van oranje. Ze zijn niet van hetzelfde type.

Open de eigenschappen van de vermenigvuldiger en stel *IO Type* in op `float`. Stel ook het veld *Constant* in op `audio_gain` om het te verbinden met de schuifregelaar. Sluit het blok en zie hoe de draden zwart zijn geworden, aangezien alle audio in- en uitgangen nu oranje zijn.

Uitgangsopties

Standaard is de uitvoer van GRC een Python-script met GUI zoals gedefinieerd door de *Generate Options* in het *Options*-blok. Maar er zijn andere mogelijkheden, zoals C++. Als je alle GUI-blokken uit het flowdiagram verwijdert en *Generate Options* instelt op *No GUI*, kan het script buiten GRC worden uitgevoerd. Zo kunt je stand-alone SDR-toepassingen bouwen.

Dat is het. Voer het flowdiagram uit en geniet van je softwaregedefinieerde FM-ontvanger! Om dieper in te gaan op de fascinerende wereld van SDR, raad ik ten eerste aan de uitstekende videocursus van Michael Ossmann [6] te bekijken, die de inspiratie vormde voor dit artikel. 

220659-03

Vragen of opmerkingen?

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



Gerelateerde producten

- > Great Scott Gadgets HackRF One Software Defined Radio (1 MHz to 6 GHz) (SKU 18306) www.elektor.nl/18306
- > Great Scott Gadgets ANT500 Telescopic Antenna (75 MHz to 1 GHz) (SKU 18481) www.elektor.nl/18481
- > Elektor SDR Hands-on Kit (SKU 19041) www.elektor.nl/19041


SDR-gebaseerde
FM-radio-ontvanger op
Elektor.TV



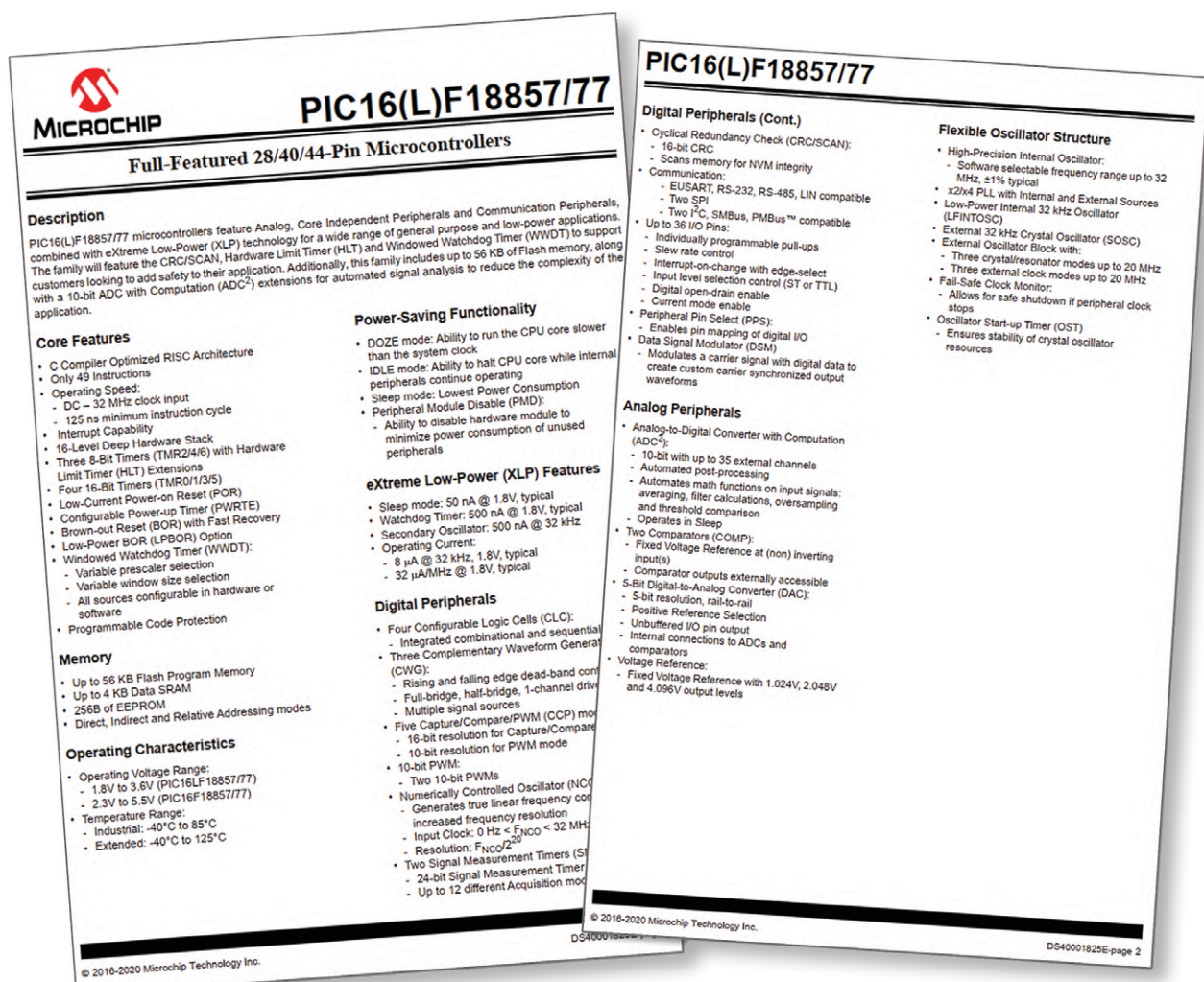
WEB LINKS

- [1] HackRF One bij Great Scott Gadgets: <https://greatscottgadgets.com/hackrf/>
- [2] GNU Radio en GRC: <https://www.gnuradio.org/>
- [3] SDR# ofwel SDR Sharp: <https://airspace.com/download/>
- [4] SDR++: <https://www.sdrpp.org/>
- [5] HackRF One op GitHub: <https://github.com/greatscottgadgets/hackrf>
- [6] Videocursus over SDR door Michael Ossmann: <https://greatscottgadgets.com/sdr/>



Microcontroller-documentatie verklaard

(deel 1) de opzet van een datasheet

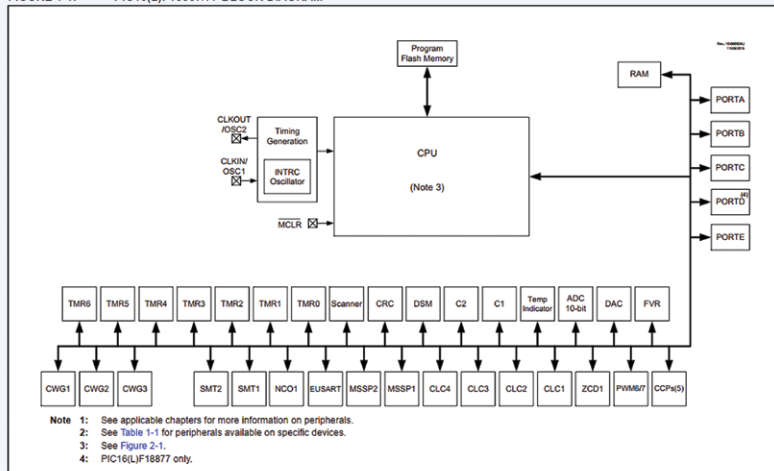


Figuur 1. Op de eerste pagina's zien we een overzicht van de functies die deze microcontroller te bieden heeft (bron: Microchip Technology).

Stuart Cording (Elektor)

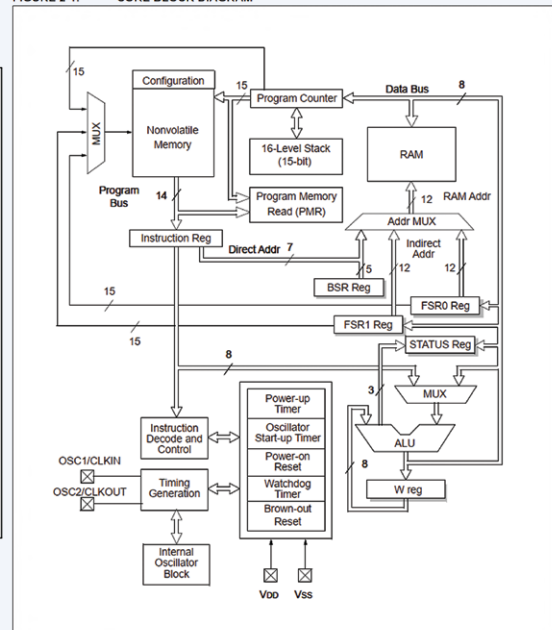
Documentatie... je kunt er van houden of niet, maar je kunt er niet omheen. Bij microcontrollers hoort altijd een heleboel documentatie omdat ze in vergelijking met andere, eenvoudiger chips echt complex zijn. In deze reeks artikelen bekijken we microcontroller-documentatie: wat staat er in de datasheet, wat staat er niet in en waar is de ontbrekende informatie dan wel te vinden?

FIGURE 1-1: PIC16(L)F18857/77 BLOCK DIAGRAM



Figuur 2. Het blokschema is een goed startpunt om de mogelijkheden van de microcontroller te leren kennen (links). Soms is er ook een blokschema van de processorcore (rechts) (bron: Microchip Technology).

FIGURE 2-1: CORE BLOCK DIAGRAM



De datasheet van een microcontroller kan tegenwoordig gemakkelijk meer dan 600 pagina's tellen. Gelukkig heeft Elektor allerlei hulpmiddelen in de aanbieding, zoals boeken en artikelen voor beginners, die u op gang kunnen helpen. Maar toch zult u zich, vroeger of later, in de echte microcontroller-documentatie moeten verdiepen. En hoewel de datasheet van een microcontroller behoorlijk dik kan zijn, staat er waarschijnlijk toch nog niet alle informatie in die u nodig hebt.

Naast de datasheet hebt u ook documentatie nodig over de tools die de broncode omzetten in firmware, tools voor het ontwikkelen en debuggen van code en tools voor programmeren bij massaproductie. Als de wereld van microcontrollers nieuw voor u is, kan deze artikelreeks u helpen te begrijpen waar verschillende zaken gedocumenteerd zijn, hoe u de hiërogliefen in de documentatie kunt lezen en waar u kunt vinden of er ook fouten in de documentatie zitten.

Een voorbeeld van microcontroller-documentatie: PIC16F18877

We beginnen dit eerste artikel van een serie van drie over microcontroller-documentatie met een eenvoudige 8-bit microcontroller: de PIC16F18877 van Microchip Technology. Welke documentatie wordt daarvoor aangeboden? Open

de link [1] en klik dan op "Documents". U ziet een datasheet, errata, wat "Supporting Collateral" over toepassingsspecifieke onderwerpen en evaluatieboards, de programmeerspecificaties en een lange lijst van application notes. Verderop op de pagina vindt u enige broncode bij enkele van de application notes, verkoopbrochures en een white paper over analoog/digitaal-converterers (ADC's).

Wat staat er in de datasheet van een microcontroller?

Om te beginnen downloaden we de datasheet van de PIC16F18877 [2]. Datasheets voor microcontrollers kunnen behoorlijk overweldigend zijn en toch staat er niet alles in wat u erover zou moeten weten. In **figuur 1** ziet u wat u er minimaal in zou moeten aantreffen.

➤ **Gedetailleerde informatie over één of meer microcontrollers.** Vaak worden op basis van één siliciumchip verschillende microcontrollers ontwikkeld. Die hebben dan bijvoorbeeld een verschillende behuizing of een verschillend aantal aansluitingen (pinnen of pootjes). De fabrikant maakt dan geen individuele documentatie voor elke variant, maar beschrijft liever verschillende types tegelijk in één datasheet. Dat is ook hier het geval.

Deze datasheet betreft twee varianten: de PIC16F18857 en de PIC16F18877.

➤ Blokschema van de microcontroller.

Meestal zien we daar de processorcore (het detailniveau kan verschillen: hoe ingewikkelder de core, des te eenvoudiger het schema), het geheugen, de bussen en de interfaces. Dit kan u een snel overzicht geven van wat de microcontroller allemaal kan. In dit geval vinden we het blokschema van de microcontroller (**figuur 2**) op pagina 18 en een blokschema van de processorcore op pagina 33.

➤ **Behuizingsvarianten.** Die kunnen variëren van traditionele through-hole uitvoeringen (voor zover nog verkrijgbaar) tot allerlei SMD-varianten. In dit voorbeeld begint dat op pagina 4.

➤ **Geheugenvarianten.** Vaak kunt u kiezen voor een verschillende omvang van RAM-, flash-, EEPROM- en andere geheugentypen zoals caches. In ons voorbeeld zien we een globaal overzicht op pagina 3. De tabel geeft de verschillende geheugengroottes en de geïmplementeerde periferie (**figuur 3**).

➤ **Blokschema's van de on-chip interfaces.** De functionaliteit van geïntegreerde interfaces is gemakkelijker te begrijpen met behulp van een schema. Blokschema's verduidelijken de pinning, waar de kloksignalen vandaan

PIC16(L)F188XX Family Types

Device	Data Sheet Index	Program Flash Memory (Words)	Program Flash Memory (KB)	EEPROM (bytes)	Data SRAM (bytes)	I/O Pins ⁽¹⁾	10-Bit ADC ² (ch)	5-Bit DAC	Comparator	8-Bit (with HLT)/16-Bit Timers	SMT	Windowed Watchdog Timer	CRC and Memory Scan	CCP10-Bit PWM	Zero-Cross Detect	CWG	NCO	CLC	DSM	EUSART/I ² C/SPI	Peripheral Pin Select	Peripheral Module Disable
PIC16(L)F18854	(1)	4096	7	256	512	25	24	1	2	3/4	2	Y	Y	5/2	Y	3	1	4	1	1/2	Y	Y
PIC16(L)F18855	(2)	8192	14	256	1024	25	24	1	2	3/4	2	Y	Y	5/2	Y	3	1	4	1	1/2	Y	Y
PIC16(L)F18856	(3)	16384	28	256	2048	25	24	1	2	3/4	2	Y	Y	5/2	Y	3	1	4	1	1/2	Y	Y
PIC16(L)F18857	(4)	32768	56	256	4096	25	24	1	2	3/4	2	Y	Y	5/2	Y	3	1	4	1	1/2	Y	Y
PIC16(L)F18875	(2)	8192	14	256	1024	36	35	1	2	3/4	2	Y	Y	5/2	Y	3	1	4	1	1/2	Y	Y
PIC16(L)F18876	(3)	16384	28	256	2048	36	35	1	2	3/4	2	Y	Y	5/2	Y	3	1	4	1	1/2	Y	Y
PIC16(L)F18877	(4)	32768	56	256	4096	36	35	1	2	3/4	2	Y	Y	5/2	Y	3	1	4	1	1/2	Y	Y

Figuur 3. Een overzicht van de geheugenopties voor de verschillende PIC16F18x7-varianten en de geïntegreerde interfaces (bron: Microchip Technology).

Vragen of opmerkingen?

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

(zoals de ontvangst van een databyte). Hoe dit alles ontcijferd wordt, komt in deel 2 van deze reeks aan de orde.

► **Elektrische specificaties.** Hier vindt de gebruiker informatie over de toelaatbare spanningen en stromen op de aansluitingen van de microcontroller. Die worden meestal tweemaal gespecificeerd: eenmaal als absolute maximumwaarden, en eenmaal als minimale, typische, en maximale waarden tijdens normaal gebruik. Er worden ook nog timingeigenschappen genoemd, zoals in **figuur 4**. In deze datasheet zijn die te vinden vanaf **pagina 592**.

► **Aanbevolen tools.** U hebt tools nodig voor het omzetten van de broncode naar assembler en voor het debuggen van de resultaten. De meeste datasheets geven aanbevelingen voor de daarvoor te gebruiken tools. In dit geval is daar iets over te vinden op **pagina 638**.

37.1 Absolute Maximum Ratings⁽¹⁾

Ambient temperature under bias		-40°C to +125°C
Storage temperature		-65°C to +150°C
Voltage on pins with respect to V _{SS} on V _{DD} pin		-0.3V to +6.5V
PIC16F18857/77		-0.3V to +6.5V
PIC16F18857/77		-0.3V to +6.5V
on NC pins		-0.3V to +6.5V
on all other pins		-0.3V to (V _{DD} + 0.3V)
Maximum current on V _{SS} pin ⁽¹⁾		350 mA
-40°C ≤ T _A ≤ +85°C		120 mA
85°C < T _A ≤ +125°C		120 mA
on V _{DD} pin for 28-Pin devices ⁽¹⁾		250 mA
-40°C ≤ T _A ≤ +85°C		85 mA
85°C < T _A ≤ +125°C		85 mA
on V _{DD} pin for 40-Pin devices ⁽¹⁾		350 mA
-40°C ≤ T _A ≤ +85°C		120 mA
85°C < T _A ≤ +125°C		120 mA
on any standard I/O pin		±50 mA
Clamp current, I _C (V _{DS} < 0 or V _{DS} > V _{DD})		±20 mA
Total power dissipation ⁽²⁾		600 mW

37.3 DC Characteristics

TABLE 37-1: SUPPLY VOLTAGE

Standard Operating Conditions (unless otherwise stated)					
Param. No.	Sym.	Characteristic	Min.	Typ [†]	Max.
Supply Voltage					
D002	V _{DD}		1.8	—	3.6
			2.5	—	3.6
			2.5	—	5.5
			2.5	—	5.5
RAM Data Retention ⁽¹⁾					
D003	V _{DD}		1.5	—	—
D003	V _{DD}		1.5	—	—
Power-on Reset Release Voltage ⁽²⁾					
D004	V _{DD}		—	1.6	—
D004	V _{DD}		—	1.6	—

Power-on Reset Release Voltage⁽²⁾

TABLE 37-13: ANALOG-TO-DIGITAL CONVERTER (ADC) CONVERSION TIMING SPECIFICATIONS					
Standard Operating Conditions (unless otherwise stated)					
Param. No.	Sym.	Characteristic	Min.	Typ [†]	Max.
A000	T _{AD}	ADC Clock Period	1	—	9
A021	T _{CONV}	Conversion Time	—	11	12
A022	T _{ACQ}	Acquisition Time	—	2	—
A023	T _{SHD}	Sample and Hold Capacitor Discharge Time	—	—	—

Figuur 4. Elektrische specificaties worden gegeven als absolute maxima, maar ook als minimale, typische, en maximale waarden voor de spanningen en stromen bij normaal bedrijf. Daarnaast zijn er gegevens over de timing (bron: Microchip Technology).

komen, en waar diverse klokkuitgangen van de oscillator en (indien voorhanden) PLL (phase locked loop) te vinden zijn. In deel twee van deze reeks gaan we daar een aantal voorbeelden van bekijken.

► Beschrijving van de registers.

Elke interface kan op een specifieke

manier worden geconfigureerd. Bij een UART (Universal Asynchronous Receiver/Transmitter) gaat het daarbij om de baudrate, het aantal databits enzovoort. De registerbeschrijving verklaart hoe de interface geconfigureerd moet worden en hoe we de status kunnen opvragen na bepaalde events

Volgende keer: meer over blokschema's

Nu we de basisstructuur van de datasheet van een microcontroller hebben gezien, gaan we in het volgende artikel bekijken hoe de registers worden beschreven en hoe we de blokschema's moeten lezen. We gaan dan ook in detail in op twee belangrijke blokken in de microcontroller: de klokgenerator (oscillator) en de implementatie van het resetcircuit. ◀

200721-04

WEB LINKS

- [1] PIC16F18877-productpagina: <http://bit.ly/2KS1s8C>
- [2] PIC16F18877-datasheet: <https://bit.ly/3nNwPjn>



Gerelateerde producten

- **T. Hanna, Microcontroller Basics with PIC, Elektor 2020**
www.elektor.nl/microcontroller-basics-with-pic
- **A. Pratt, Programming the Finite State Machine, Elektor 2020**
www.elektor.nl/programming-the-finite-state-machine



Hoe verder met AI en embedded systems?

tools, platforms – en worden schrijvers obsoleet?

Stuart Cording (Elektor)

AI wordt vaak gezien als een universeel probleemoplossend instrument dat elke uitdaging aankan. Maar ook wordt beweerd dat het het einde van de beschaving zal betekenen. Als technici weten we dat geen van beide gevallen waar is. Maar hoe kunnen we beter gebruik maken van AI, of eigenlijk machine learning, in de toepassingen die we ontwikkelen? En zullen de meest recente ontwikkelingen op het gebied van AI ons echt overbodig maken?

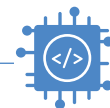
Het noemen van kunstmatige intelligentie (AI) trekt gegarandeerd de aandacht van de reguliere pers. Dankzij internet en cloud services, gecombineerd met soms twijfelachtige informatiebronnen, lijken overal nieuwe AI-gestuurde services op te duiken. Met een paar muisklikken kunnen je woorden in de mond van Morgan Freeman worden gelegd of kan je foto worden geïntegreerd in een nieuw kunstwerk. En hoewel dit amusant en tot nadenken stemmend kan zijn, is het niet gemakkelijk om een link terug te vinden naar de wereld van embedded systems. Maar dat is precies wat er gaande is binnen de industrie. De meeste embedded systemen gebruiken op regels gebaseerde programmeermethoden om hun functionaliteit te verwezenlijken. Op basis van de input bepaalt een reeks if/else- of switch-statements hoe er gereageerd moet worden. Dit werkt goed voor een beperkt aantal ingangen. Maar op een gegeven

moment maakt het aantal ingangen of de complexiteit van hun relatie het moeilijk om duidelijke programmeerregels te definiëren.

Stel je bijvoorbeeld een motor voor in een fabriek die 24 uur per dag, zeven dagen per week draait. De ervaring leert dat na verloop van tijd slijtage optreedt en het lagervet dikker wordt. Mettertijd veranderen hierdoor de opstarttijd, het geproduceerde geluid, de trillingen, de bedrijfstemperatuur van de motor en de opgenomen stroom. Regelmatig onderhoud is de manier waarop deze problematiek tegenwoordig wordt aangepakt, maar het resulteert in regelmatige onderbrekingen die de productie stilleggen. Of dit te vaak of niet vaak genoeg gebeurt is vaak moeilijk te zeggen. Bovendien is het onwaarschijnlijk dat een potentiële storing als gevolg van een haarscheurtje in de as, lagers, behuizing of bevestigingen wordt opgemerkt. Mits goed getraind kunnen AI-methoden potentiële storingen vaststellen aan de hand van een complexe mix van informatiebronnen. Als dergelijke intelligentie kan worden toegepast in een systeem op basis van een microcontroller, krijg je een betaalbaar controlesysteem dat tijd en geld bespaart, en verspilling door onnodige smering en vervanging van onderdelen vermindert.

Microcontrollers intelligent maken

Op het niveau van de microcontroller hebben we het eerder over machine learning (ML) dan over AI. Dit houdt in dat een machine geprogrammeerd wordt om regels te gebruiken die ontwikkeld zijn om op basis van de analyse van beschikbare gegevens beslissingen te nemen. Hoewel microcontrollers meer dan krachtig genoeg zijn om dergelijke ML-algoritmen uit te voeren, blijft het leren van trainingsgegevens buiten hun bereik en is daarvoor minstens een



desktopcomputer nodig, en misschien zelfs een cloudserver. Edge Impulse, opgericht in 2019, heeft een platform ontwikkeld dat gewijd is aan ML in embedded systems, en heeft met succes samengewerkt met halfgeleiderleveranciers wereldwijd [1] om brede ondersteuning uit te rollen.

Het uitgangspunt voor elke ML-toepassing is data. Terwijl sommige toepassingen, zoals autonoom rijden, terabytes aan trainingsdata nodig hebben, kunnen eenvoudige microcontroller-gebaseerde systemen leren van slechts een paar kilobytes aan data. De gegevens uit het board in de Edge Impulse-omgeving te krijgen, is dus de eerste uitdaging. De eerste gedachte zou zijn om de seriële interface van een Arduino te gebruiken om de gegevens naar de PC te sturen en ze van daaruit als tekstbestand door te sturen. Hun platform is echter ingesteld om de gegevens rechtstreeks te verwerken.

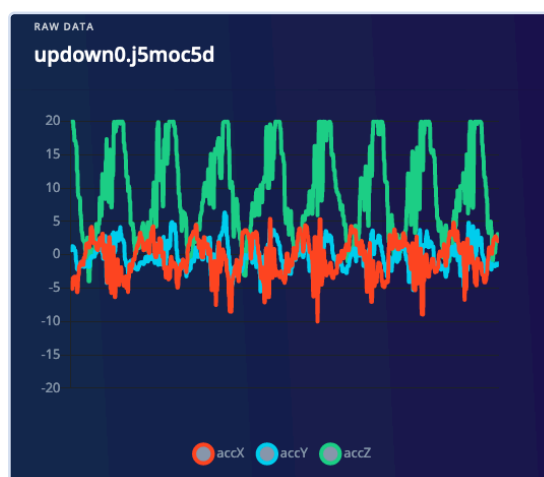
Data - voer voor AI

Eén hulpmiddel is de Data Forwarder [2], een commandoregel-toepassing (CLI) die gegevens van een ontwikkelbord rechtstreeks naar de Edge Impulse-omgeving stuurt. Met je gebruikersnaam en wachtwoord wordt een verbinding tot stand gebracht tussen de seriële poort op je PC en de server. Aan de kant van de microcontroller is uitvoer van data via de seriële interface in een komma- of tabgescheiden formaat alles wat nodig is. Zolang de bemonsteringsfrequentie relatief laag is, is dit een ideale manier om rechtstreeks representatieve gegevens van je sensoren te verzamelen (**figuur 1**). Krachtiger embedded systems, zoals de Raspberry Pi of de NVIDIA Jetson Nano, kunnen gebruik maken van de meegeleverde software development kit (SDK) [3]. Die ondersteunt ook sensoren zoals microfoons en camera's die grotere hoeveelheden data genereren.

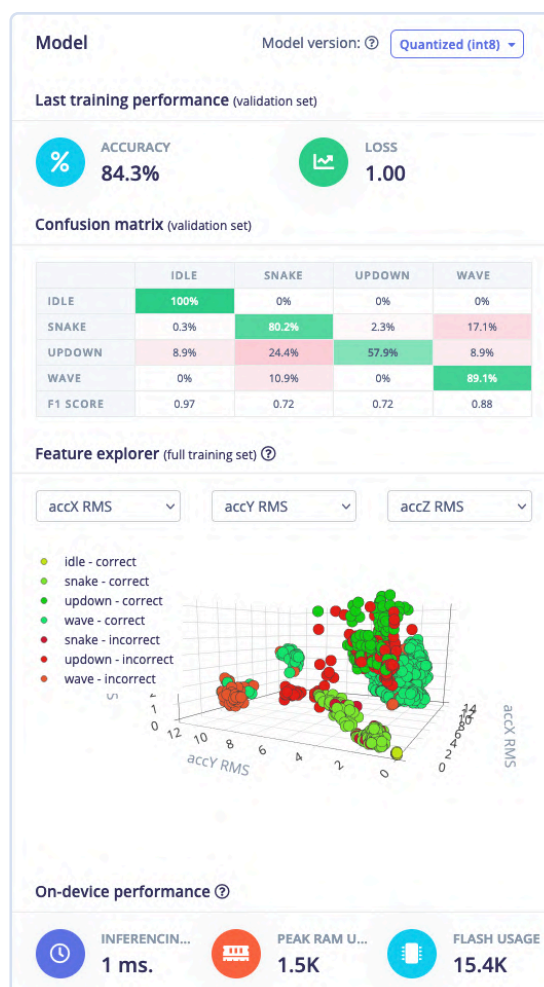
Met de data die naar Edge Impulse zijn verstuurd, is de volgende stap het definiëren van een 'impuls'. Deze bestaat uit twee blokken. De eerste hakt de data in kleinere brokken en gebruikt signaalverwerkingstechnieken om er informatie uit te halen. Dit zorgt ervoor dat de beschikbare sensorgegevens worden omgezet in consistente informatie voor de tweede verwerkingsfase. Het tweede blok is waar het leren en classificeren plaatsvindt (**figuur 2**). In een voorbeeldproject met de naam "Continuous Motion Recognition" wordt goed uitgelegd hoe deze blokken worden geconfigureerd om accelerometerdata te analyseren en deze input te classificeren als een van vier gebaren [4].

Dit is de meest kritische stap in elke ML-ontwikkeling, waarbij vaak creatief moet worden nagedacht over de beste aanpak. Soms is het negeren van de invoer van sommige sensoren de beste oplossing, terwijl in andere gevallen juist meer gegevens nodig zijn. Misschien merk je dat je teveel leert, of dat het gekozen neurale netwerkmodel slecht past bij de classificatie die je

probeert te maken. Een andere cruciale stap is de classificatie van afwijkingen. In het voorbeeldproject zijn er vier gedefinieerde gebaren. Andere bewegingen die op de geleerde gebaren lijken, moeten echter worden uitgesloten. Goede anomaliedetectie levert een betrouwbaarder ML-resultaat op.

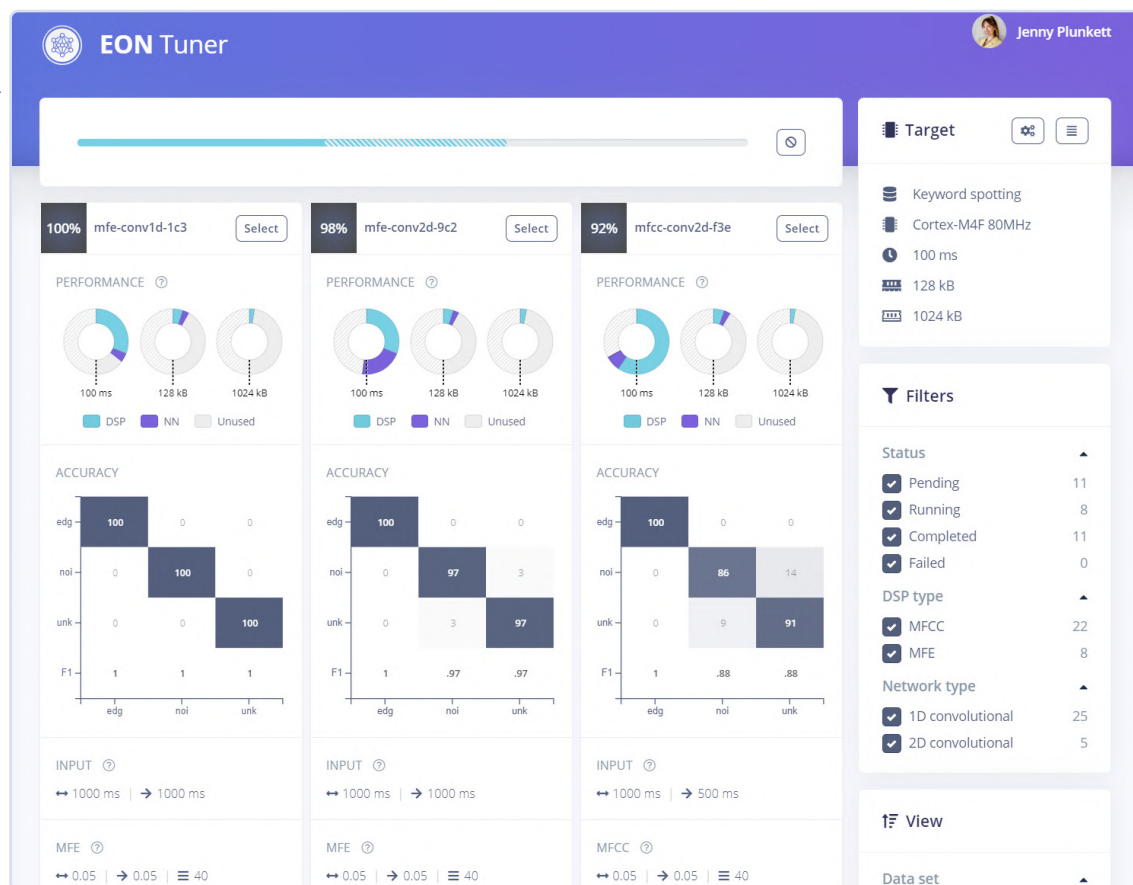


Figuur 1. Een drie-assige accelerometer levert bewegingsdata om een toepassing te ontwikkelen voor ML-gebarenherkenning (bron: Edge Impulse).



Figuur 2. De Edge Impulse-omgeving geeft feedback over nauwkeurigheid, snelheid en geheugengebruik na eerste trainingscyclus (bron: Edge Impulse).

► **Figuur 3.** ML-algoritmen worden verder geoptimaliseerd voor het doelsysteem met behulp van EON Tuner, waardoor de inferentieresponstijden verbeteren en het geheugengebruik afneemt (bron: Edge Impulse).



▼ **Figuur 4.** SlateSafety gebruikte Edge Impulse om edge ML fysiologische monitoring en de BAND V2 veiligheidsmonitoring te verbeteren (bron: SafetySlate).

De laatste stap is de implementatie. Via de webinterface wordt firmware voor het gekozen apparaat gedownload voor integratie in je toepassing. Voor Arduino wordt een bibliotheek gegenereerd, terwijl je voor andere microcontrollers een C++ bestand kunt genereren. Natuurlijk variëren de prestaties van microcontrollers enorm. Om het best mogelijke resultaat te garanderen, biedt Edge Impulse de EON Tuner [5] aan. Dit tool kan de detectie nauwkeurigheid verder verbeteren, de inferentie versnellen en de geheugeneisen verminderen door informatie te gebruiken

zoals doelapparaat, grootte van het geheugen en latentie (**figuur 3**).

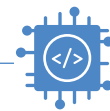
ML in echte toepassingen

Echte toepassingen gebruiken deze aanpak om ML-functionaliteit in te bouwen. BAND van SlateSafety [6] integreert een reeks biometrische sensoren om werknemers te controleren die onder moeilijke omstandigheden werken (**figuur 4**). Deze variëren van eerstehulpverleners tot werknemers in de industrie die zware persoonlijke beschermingsuitrusting dragen, zoals brandweerlieden. Die uitrusting stuurt gewoonlijk gegevens naar de cloud, zodat collega's de vitale functies van een gebruiker kunnen controleren. Maar vooral bij rampen kunnen verbindingen haperen of compleet uitvallen. Het ontwikkelingsteam gebruikte Edge Impulse om edge ML te integreren in de bestaande uitrusting, getraind op historische biometrische gegevens. Met behulp van de EON Tuner werd het algoritme optimaal afgestemd op de hardware en vervolgens ingezet via een over-the-air update. Nu kan BAND de drager een waarschuwing geven wanneer er een risico op een hitecollaps bestaat, zelfs wanneer er geen draadloze verbinding is.

Productontwikkeling verbeteren met AI

Natuurlijk hoeft AI niet geïntegreerd te zijn in het product. Het kan ook voor de ontwikkeling ervan worden gebruikt. Tegenwoordig worden veel complexe toepassingen op basis van modellen ontwikkeld, waarbij in wezen een beschrijving wordt ontwikkeld van hoe iets zou werken met behulp van software en





wiskundige vergelijkingen die hun oorsprong vinden in de fysica. Maar zelfs deze aanpak heeft zijn grenzen. Dit is waar Monolith en zijn zelflerende AI-platform [6] om de hoek komen kijken.

Het platform is in staat de fysieke eigenschappen van complexe systemen te leren op basis van reeds verzamelde gegevens. Voertuigen doorlopen bijvoorbeeld een serie tests op een testparcours, waarbij meerdere sensoren slinger- en -draaibewegingen monitoren, samen met de wielsnelheid en de acceleratie. Het verzamelen van gegevens voor verschillende stijfheden van de ophanging geeft een goed inzicht in hoe het voertuig reageert op verschillende rij-situaties. Normaliter worden de gegevens geanalyseerd en vervolgens in nieuwe instellingen voor de volgende testrit gebruikt. Monolith kan de gegevens van de eerste reeks testritten evalueren en het resultaat van veranderingen aan de ophanging met grote nauwkeurigheid voorspellen. De resultaten kunnen worden gebruikt om sneller de beste instellingen voor de vering te bepalen, zodat er minder extra fysieke testritten nodig zijn.

Deze benadering kan ook worden toegepast op metrologie. Gasmeters moeten uitzonderlijk nauwkeurig zijn om een correcte facturering te garanderen, maar dat vormt een uitdaging wanneer de meter een reeks verschillende gassen moet meten. Bij een klant kwam bij het simuleren van ultrasone meters een puur wiskundige analyse aan de grens van het mogelijke, zodat voor de kalibratie slechts herhaalde testprocessen overbleven als om de benodigde certificering te behalen. Gelukkig leverde al die tests een rijke verzameling gegevens op voor analyse. Met behulp van zelflerende AI-modellen is de hoeveelheid benodigde tests met maar liefst 70% afgenomen, waardoor de ontwikkeling aanzienlijk is versneld.

Forse verkleining van AI-computers

Wedstrijden zoals de DARPA Grand Challenge [8], waarbij teams autonome voertuigen bouwden die een bochtig parcours moesten kunnen afleggen, veroorzaakten een golf van belangstelling voor zelfrijdende auto's. Nu, bijna twintig jaar later, is er veel geld uitgegeven maar is daar weinig uit voortgekomen. Tesla haalt op dit gebied vaak het nieuws, vooral wanneer overenthousiaste Tesla-bezitters te veel vertrouwen stellen in de mogelijkheden van hun voertuig [9]. Voor het overige lijkt alleen Waymo echte zelfrijdende voertuigen op afroep aan te bieden [10], maar die rijden uitsluitend in Phoenix en San Francisco in de VS.

Een van de problemen is dat een computer die een auto bestuurt een uitzonderlijke uitdaging vormt. Niet alleen moet het voertuig voortdurend de situatie om zich heen beoordelen, maar het moet ook de acties van andere bestuurders en weggebruikers zoals voetgangers en fietsers voorspellen, die zich misschien niet aan de verkeersregels houden.



Wat er gebeurt is dat de elektrische en elektronische (E/E) architectuur van voertuigen verandert om te voldoen aan de toekomstige behoeften van autonome voertuigen. Met een veelheid aan sensoren die enorme hoeveelheden gegevens opleveren, gaat de industrie over op automotive Ethernet. Momenteel vormt deze benadering de basis van geavanceerde rijhulpsystemen (ADAS) die, door de controle van remmen, versnelling en stuurinrichting, kunnen ingrijpen als wij bestuurders een fout maken. Volgens de autonomeniveaus van de Society of Automotive Engineers (SAE) voldoen auto's uit de topklasse momenteel aan niveau 2+, waarbij sommige aan niveau 3 tippen. Volledige autonomie waarbij we achterover kunnen leunen en ons kunnen ontspannen is echter niveau 5, dus daar moet nog veel gebeuren.

Bedrijven als Eurotech ondersteunen de industrie om de ontwikkeling van de benodigde algoritmen te versnellen. Momenteel verzamelt een acht uur durende testrit 120 TB aan gegevens die terug moeten naar het lab voor verwerking en analyse. Verbeteringen van AI-algoritmen kunnen in het laboratorium worden getest met behulp van de verzamelde gegevens, maar er is weinig beschikbaar om het testen en de ontwikkeling van algoritmen in het veld te ondersteunen. Gebruikmakend van zijn ervaring met vloeistofkoeling biedt Eurotech een serie geavanceerde AI-hardware die daarvoor geschikt is – in wezen compacte supercomputers die in de kofferbak van een voertuig passen. Een apparaat zoals de DYNACOR 40-36 is robuust opgebouwd voor gebruik in zowel on- als off-road voertuigen [11]. Voorzien van een 16-core Intel Xeon CPU met 64 GB RAM en maximaal twee NVIDIA GV100 GPU's met 32 GB RAM, levert deze ventilatorloze computer 237 TFLOPS om deep learning-toepassingen aan te kunnen (figuur 5). Verschillende gigabit Ethernet-interfaces ondersteunen het inlezen van de enorme hoeveelheden sensorgegevens, van radar en camera's tot lidar, en sluizen die naar een 32 TB solid-state geheugen door. Wanneer tijdens testritten meer

▲
Figuur 5. De ontwikkeling van AI-algoritmen wordt versneld met behulp van krachtige in het voertuig geplaatste computers, zoals deze vloeistofgekoelde DYNACOR 40-36 (bron: Eurotech).



► *Figuur 6: Het door DALL-E 2 AI van OpenAI gegenereerde beeld van een zelfrijdend voertuig onder besturing van de vertrouwde Commodore 64. Een werk in uitvoering.*

inferentie- en versterkingstests kunnen worden uitgevoerd, zou de weg naar niveau5-autonomie aanzienlijk versneld kunnen worden.

Brengt AI mijn baan in gevaar?

Een voortdurende discussie op sociale media is of de vooruitgang in AI banen in creatieve bedrijfstakken in gevaar brengt. De lancering van DALL-E 2 door OpenAI [12] zet gesproken opdrachten om in afbeeldingen (**figuur 6**). Maar misschien nog indrukwekkender is zijn vermogen om bestaande beelden realistisch te bewerken. Het kan bijvoorbeeld objecten in de voor- of achtergrond verwijderen. En op basis van een werk van de Nederlandse schilder Vermeer, kan de AI zijn "Meisje met de parel" uitbreiden met een geloofwaardige impressie van de kamer waarin ze zat toen ze werd geschilderd.

Schrijvers, zoals de redacteurs van Elektor en andere gerenommeerde uitgeverijen, zijn echter geschokt door de lancering van ChatGPT [13]. Deze AI kan met gebruikers in spreektaal communiceren in een veelheid van talen. Tot nu toe hebben discussies over de voordelen van siliciumcarbide MOSFET's (SiC) en galliumnitride transistoren (GaN) ten opzichte van silicium MOSFET's een zeer nauwkeurige respons opgeleverd. Dus zelfs niche-onderwerpen lijken goed te worden bestreken.

Hoewel uitzonderlijk slim, kent het hulpmiddel alleen antwoorden op onderwerpen die zich voordeden voordat het werd getraind. Omdat het niet voortdurend bijleert, zal het niet op de hoogte zijn van actuele zaken of de nieuwste drama's rond K-pop bands (jammer). Een ander kritiekpunt is dat de antwoorden na een tijdje ietwat gekunsteld en formeel lijken. Maar wie op zoek is naar inspiratie of een verjaardagsrijm in de stijl van een beroemd dichter, zal niet teleurgesteld worden. Om te bepalen of de toekomst van Elektor afhankelijk is van wezens van vlees en bloed of van computers, geef ik je een samenvatting van het onderwerp van dit artikel geschreven door ChatGPT. Tot de volgende keer... of misschien toch niet!

Samengevat zijn *embedded systems* en AI twee technologieën die steeds vaker samen worden gebruikt om intelligente, autonome apparaten en systemen te maken. *Embedded systems* leveren het hardware- en softwareplatform waarop AI-algoritmen kunnen draaien, terwijl AI-algoritmen deze systemen in staat stellen hun omgeving op een intelligentere en menselijkere manier waar te nemen, te analyseren en erop te reageren. Naarmate de mogelijkheden van zowel *embedded systems* als AI verder verbeteren, kunnen we een breed scala aan spannende nieuwe toepassingen verwachten op gebieden als robotica, gezondheidszorg, vervoer en meer. ◀

220673-03

Vragen of opmerkingen?

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



Elektor Engineering Insights

Elektor Industry Insights: bekijk het live!

Elektor Industry Insights is de informatiebron bij uitstek voor engineers met weinig tijd of maker-professionals, die op de hoogte willen blijven van de wereld der elektronica. Tijdens elke episode bespreekt Stuart Cording (redacteur, Elektor) reële engineering-uitdagingen en oplossingen met experts op het gebied van industriële elektronica.

Bezoek www.elektormagazine.com/eei voor info over komende en voorbije afleveringen.

WEBLINKS

- [1] Edge Impulse partners: <http://bit.ly/3vbqRhG>
- [2] Edge Impulse CLI installatie: <http://bit.ly/3BQQt71>
- [3] Edge Impulse Ingestion SDK: <http://bit.ly/3jplhp3>
- [4] Continuous motion recognition voorbeeldproject: <http://bit.ly/3WAV9G5>
- [5] EON Tuner: <http://bit.ly/3PQhFsr>
- [6] SlateSafety BAND: <http://bit.ly/3YVpe5x>
- [7] Monolith: <http://bit.ly/3BWZlmh>
- [8] DARPA Grand Challenge, Wikipedia: <http://bit.ly/3VnqWJI>
- [9] J. Stilgoe, "Tesla crash report blames human error – this is a missed chance," Guardian, januari 2017: <http://bit.ly/3Vjp7gI>
- [10] DYNACOR 40-36: <http://bit.ly/3GdmgBS>
- [11] Waymo One: <http://bit.ly/3FNG8Ku>
- [12] DALL-E 2: <http://bit.ly/3PNPWsD>
- [13] ChatGPT: <http://bit.ly/3PLjZRo>

Digitalisering

van verticale landbouw

Een bijdrage van Würth Elektronik eiSos

Verticale landbouw is een veelbelovende aanpak om een groeiende wereldbevolking te voeden. Geoptimaliseerde verlichting, irrigatie, bemesting en klimaatbeheersing zorgen voor de hoogste opbrengsten en de beste kwaliteit. Dit artikel presenteert een IoT-connected prototype.

Wat is digitale landbouw?

“Digitale landbouw is de naadloze integratie van digitale technologieën in gewassen veeteeltbeheer en andere processen in de landbouw. Voor boeren biedt digitale landbouw de mogelijkheid om de productie te verhogen, op lange termijn kosten te besparen en risico's uit te sluiten.” [3]

Deze aanpak kan niet alleen de opbrengst van de momenteel beschikbare landbouwgrond verhogen, maar kan ook met succes worden toegepast in dichtbevolkte stedelijke gebieden waar ruimte schaars is. De belangrijkste voordelen van indoor farming zijn dat je daarmee je eigen gewassen kunt kweken, milieufactoren kunt controleren, de kwaliteit kunt garanderen en koolstofuitstoot kunt minimaliseren. Het meest voor de hand liggende voordeel van indoor vertical farms is de ruimte. Verticale boerderijen zijn in staat om de productiviteit drastisch te verhogen door meer gewassen onder te brengen op eenzelfde oppervlak.

Er zijn meerdere uitdagingen bij digitale vertical farming. Het grootste nadeel van vertical farming zijn de hoge elektriciteitskosten die nodig zijn om het grote aantal LED's van stroom te voorzien. In plaats van gewoon wit volspectrumlicht kunnen LED's worden gebruikt die alleen de noodzakelijke golflengtes leveren. Het is echter een enorme uitdaging om op grote schaal voor optimale lichtomstandigheden te zorgen en het energieverbruik handmatig binnen de perken te houden.

Bovendien veranderen de milieumomstandigheden die nodig zijn voor een optimale groei, afhankelijk van de keuze van de plant en de groeifase. Het is uiterst moeilijk om

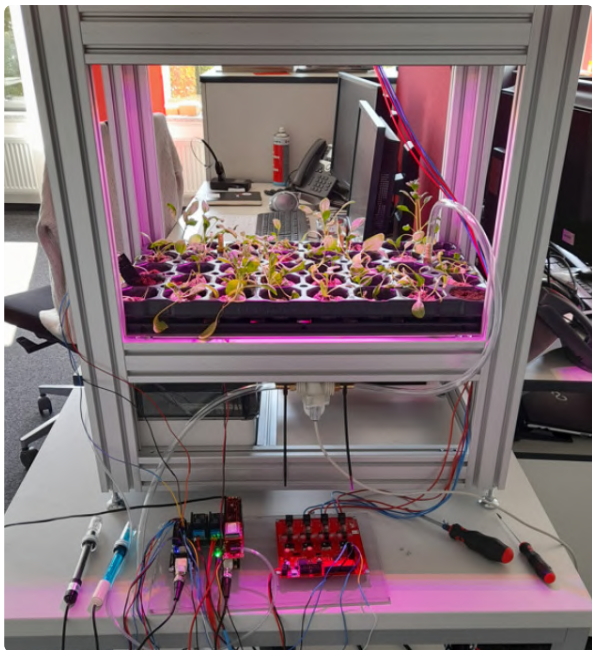
Een verwachte wereldbevolking van bijna 10 miljard zal tegen 2050 een toename van de wereldwijde landbouwproductie met 50 procent vereisen, volgens de Voedsel- en Landbouworganisatie van de VN (FAO). Toch zijn landbouwgrond en andere agrarische activiteiten beperkt. De hoeveelheid landbouwgrond is de afgelopen decennia zelfs afgenomen, van bijna 40% van het landoppervlak in 1991 tot slechts 37% in 2018 [1].

Wereldwijd beslaat de landbouwgrond in 2020 naar schatting 4,7 miljard hectare, ongeveer een derde van het totale landoppervlak. Van dit totale landbouw-oppervlak bestaat ruwweg een derde uit akkerland (ongeveer 1,6 miljard hectare) en de overige twee derde uit permanent gras- en weideland (ongeveer 3,2 miljard hectare) [2]. 1,6 miljard hectare is 12% van het landoppervlak van de aarde. Deze mondiale aandelen zijn sinds 2000 niet noemenswaardig veranderd, maar aan de andere kant is de wereldbevolking toegenomen van 6,15 miljard tot 7,91 miljard in 2021, dat is bijna 30% toename en de groei blijft

doorgaan.

Innovatieve landbouwtechnieken, zoals gecontroleerde omgevings-landbouw (ook wel verticale landbouw genoemd), zijn veelbelovend als middel om de landbouwproductie te verhogen en daarmee te voldoen aan lokale of regionale voedselbehoeften en om te helpen miljarden mensen van versere en voedzamere producten te voorzien, terwijl de milieueffecten van conventionele landbouw tot een minimum worden beperkt. Digitale landbouwbedrijven kunnen alleen slagen als verschillende sleuteltechnologieën, waaronder geavanceerde verlichtingssystemen, controle en toezicht, alsmede regionale acceptatie en passende regelgeving, effectief worden toegepast.

Het doel van dit artikel is te bespreken hoe kunstlicht, monitoring en besturing op afstand belangrijk zijn voor de groei van planten in verticale kweekinstallaties en hoe je je eigen verticale kweekinstallatie op een snelle en kosteneffectieve manier kunt bouwen met behulp van *rapid prototyping tools*.



Figuur 1. Prototype van de indoor farming box (bron: Würth Elektronik eiSos).

verschillende omgevingsfactoren zoals temperatuur, lucht- en bodemvochtigheid te controleren en dynamisch te optimaliseren.

IoT om verticale landbouw te automatiseren

In dit artikel wordt een IoT-oplossing voorgesteld om bovenstaande uitdagingen te overwinnen en tot volledige automatisering te komen. Het Internet of Things (IoT) kan ruim worden gedefinieerd als een overkoepelende term voor een reeks technologieën waarmee apparaten met elkaar kunnen worden verbonden en op elkaar kunnen reageren. Onderling verbonden

apparaten die gegevens genereren, maken een reeks nieuwe toepassingen mogelijk. Industriële automatisering, gezondheidszorg, slimme woningen, slimme steden, slimme netwerken en slimme landbouw zijn enkele van de gebieden waarop de IoT-technologie aanzienlijke voordelen biedt. Het industriële IoT (IIoT), ook wel de 'vierde industriële revolutie' of Industry 4.0 genoemd, is de digitalisering van activa en processen die producten, machines, diensten, locaties/sites verbindt met werknemers, managers, leveranciers en partners. Het nauwer verbinden van de digitale wereld met de fysieke wereld

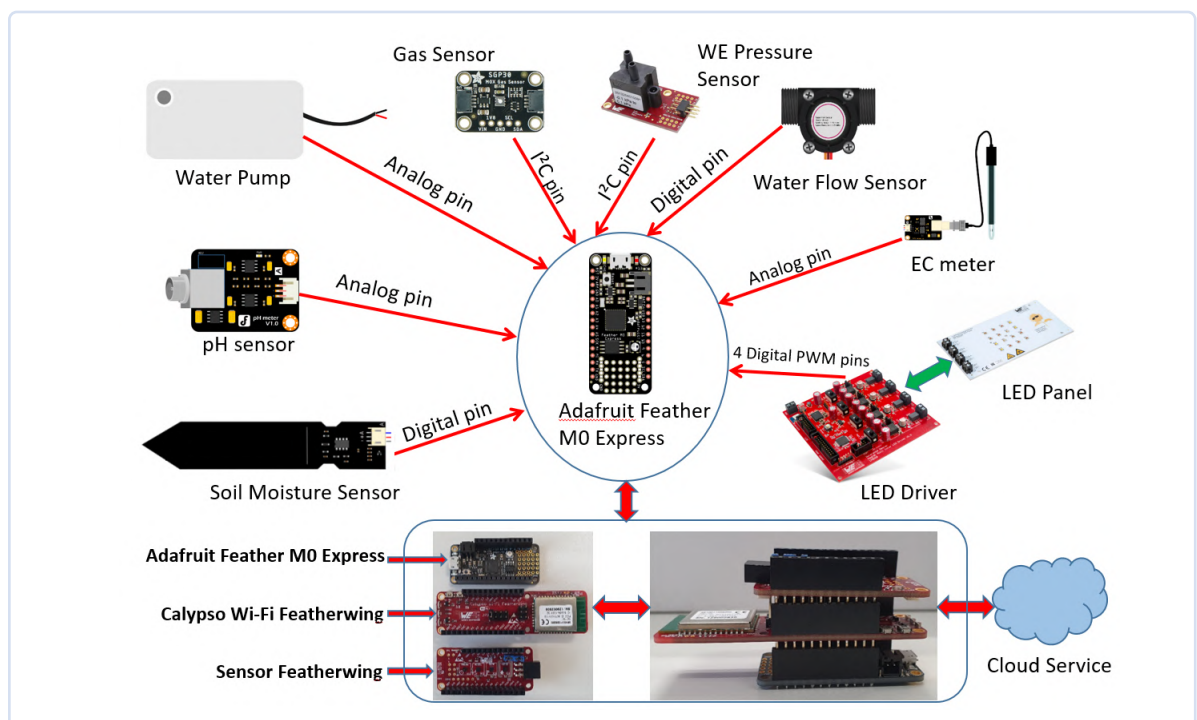
van machines draagt bij tot een hogere productiviteit, veiligheid, efficiëntie en duurzaamheid.

Het IIoT creëert een universum van sensoren dat een versnelde deep learning van bestaande operaties faciliteert en die op zijn beurt snelle contextualisering en automatische patroon- en trenddetectie mogelijk maakt. Dit leidt tot echte kwantitatieve vastlegging van kwalitatieve operaties, wat resulteert in betere kwaliteit, efficiëntie, meer veiligheid, lagere kosten en diverse andere voordelen.

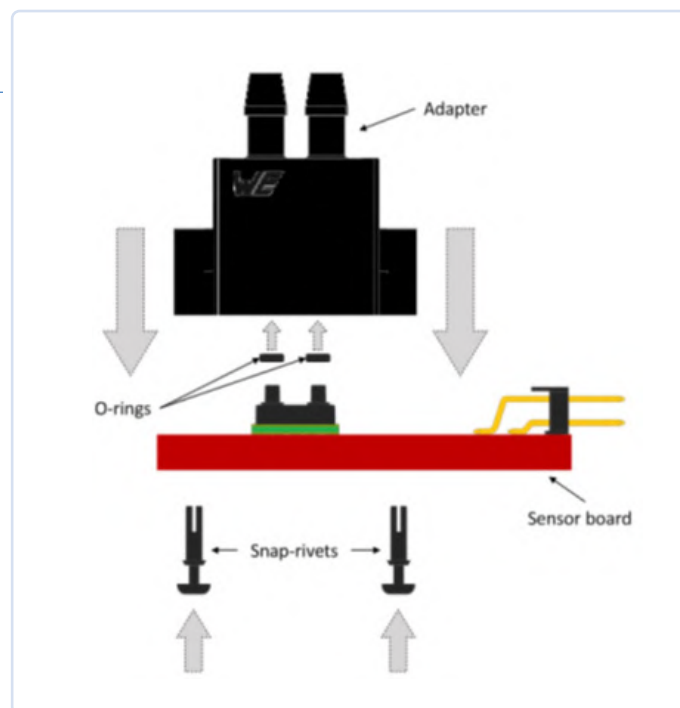
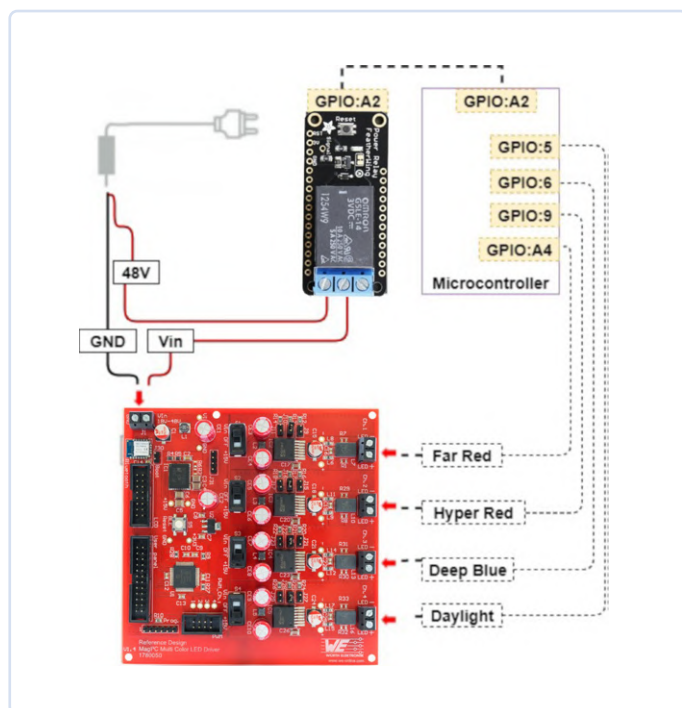
Hier wordt een op het IoT gebaseerde oplossing gepresenteerd om verlichting, irrigatie en andere omgevingsfactoren van een verticale boerderij volledig te automatiseren en te controleren. Verder wordt dit concept in de praktijk gebracht met rapid prototyping tools in de vorm van een volledig geautomatiseerd, verbonden systeem met data-analyse en cloud-connectiviteit met verschillende oplossingen voor verschillende omgevingen. **Figuur 1** toont een prototype van een indoor farming box.

"Feather"-gebaseerde implementatie

De kerntaak van elke IoT-oplossing is gegevens uit het veld naar de cloud te brengen, waar de analyse ervan de gewenste



Figuur 2. IoT-systeemprototype op basis van de Feather-vormfactor voor vertical farming (bron: Würth Elektronik eiSos).



meerwaarde voor de toepassing oplevert. Voor het realiseren van een dergelijke toepassing is gebruik gemaakt van open source hardware- en software-ecosystemen en is een complete IoT-oplossing gebouwd. Voor dat doel zijn Würth Elektronik FeatherWings gecombineerd met het Adafruit M0 Express Feather Board, in combinatie met onze LED's en LED-drivers, en diverse andere sensoren en componenten (**figuur 2**).

Het basisidee was om een digitaal verticaal landbouwsysteem te implementeren om daarmee enerzijds de groei en anderzijds het stroom- en -waterverbruik te optimaliseren.

Gecontroleerde irrigatie en verlichting

Een verticaal landbouwsysteem bestaat uit een op aarde gebaseerde verticale kweekopstelling, met een vierkanaals LED-driver en een RGBW LED design kit. Dit biedt een eenvoudige oplossing om RGBW-kleuren te mengen voor verschillende verlichtings-situaties, de groei van planten te versterken met het tuinbouwpaneel – en het maakt zelfs binnenverlichting mogelijk op basis van een Human Centric Lighting-concept (HCL). Het druppel-irrigatiesysteem en de watertank maken gebruik van gerecycleerd water, waarvan de pH- en EC-waarden worden gemeten, gecontroleerd en verbonden met de cloud met behulp van verschillende connectiviteitsmodules van Würth Elektronik.

Een bodemvochtigheidssensor controleert de bodemvochtigheid en indien nodig wordt het water met een kleine pomp in het systeem gepompt. De rest van het water wordt opgevangen, gefilterd en teruggeleid naar een tank. De kern van het systeem is een Adafruit Feather M0 Express-board en het Adafruit vermogensrelais. FeatherWings worden gebruikt als schakelaars. Gegevens worden naar de cloud gestuurd waar de informatie wordt verwerkt, geanalyseerd en gebruikt om het systeem te besturen.

Adaptief verlichtingssysteem

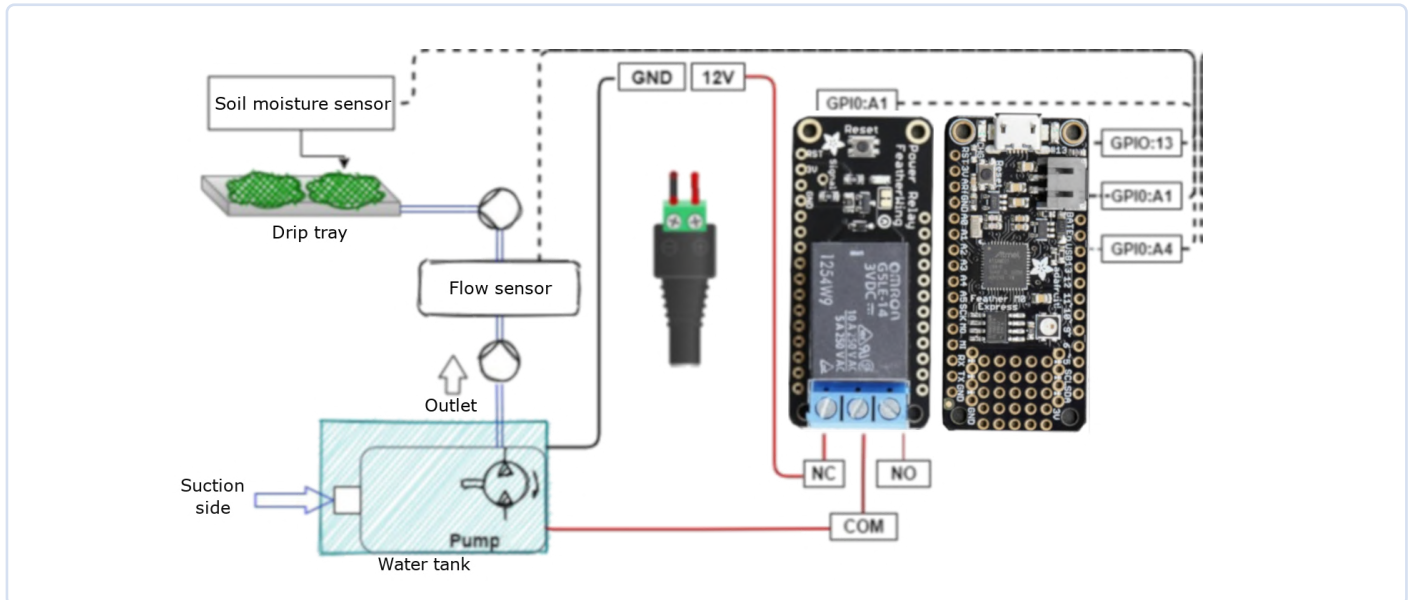
Het verlichtingssysteem (**figuur 3**) bestaat uit twee verschillende onderdelen, een Horticulture LED Panel met vier afzonderlijke kanalen met speciale monocolor LED's voor planten en de MagI3C multi-color LED-driver, beide deel uitmakend van de WE lighting development kit [4]. De kit biedt een eenvoudige oplossing om de groei van planten te stimuleren. Met een MagI3C LED Step Down High Current Module is het mogelijk de intensiteit en de kleur van elk van de vier LED-strings afzonderlijk in te stellen om te voldoen aan de toepassingsbehoeften en de LED-waarden in te stellen op het plantenprofiel. Het tuinbouwpaneel bestaat uit 6 hyperrode, 4 verre rode, 2 diepblauwe en 4 witte keramische LED's. Het systeem kan worden bestuurd met Bluetooth of via de cloud met behulp van WiFi of cellulaire connectiviteit. Hier bespreken we de universele cloud-oplossing.

Irrigatiesysteem

Het watervolume in de tank wordt gemeten met de drukverschilsensor WSEN-PDUS. Dit is een zeer nauwkeurige, MEMS-gebaseerde, piëzoresistieve sensor (**figuur 4**). Bovendien pompt een 12V-pomp het voedselrijke water via een stromingssensor in de druppelbak. De waterkwaliteit wordt gecontroleerd met een Gravity: Analog pH Sensor/Meter Kit en een Gravity: Analog Electrical Conductivity Sensor van DFRobot. Het meeste natuurlijke water heeft een pH-waarde tussen 5 en 8. De algemeen aanvaarde pH-waarde voor irrigatiewater ligt tussen 5,5 en 7,5, maar de beste resultaten zijn behaald met water met een pH-waarde tussen 5,5 en 7. Water in dit bereik houdt de voedingsstoffen in evenwicht, zorgt voor effectieve chemische desinfectie en voorkomt kalkaanslag in irrigatieapparatuur [5].

'Fertigatie' is samengesteld uit de woorden fertilisatie en irrigatie. Door het kleine substraatoppervlak vereist de substraat-teelt veel bewatering en bemesting, die aan het irrigatiewater wordt toegevoegd. Een EC-meter wordt gebruikt om een te lage of te hoge toevoer van meststoffen in het irrigatiewater op te sporen. De elektrische geleidbaarheid (EC) is een maat voor de concentratie van ionen in water. Hoe zuiverder het water, hoe geringer het geleidingsvermogen.

Voor de bodemvochtmeting werd de capacitieve Adafruit STEMMA Soil Sensor gebruikt. Capacitieve metingen gebruiken



Figuur 5. Irrigatiesysteem met sensoren, waterreservoir plus pomp en besturingsboards (bron: Würth Elektronik eiSos).

slechts één sonde, hebben geen metalen delen die kunnen oxideren, en introduceren geen gelijkstroom in de planten. De pomp wordt automatisch en rechtstreeks vanuit de cloud aangestuurd. De werkingstijd wordt berekend op basis van het aantal planten, de vereiste bodemvochtigheid en het debiet van de pomp (figuur 5).

Milieudetectie

Sensoren meten de toestand van het milieu en interpreteren die als analoge of digitale gegevens. Anderzijds activeren actuatoren een fysieke verandering in de gemeten omgeving. Dankzij de vooruitgang op het gebied van de elektronica in het algemeen en halfgeleiders in het bijzonder is er een breed scala van uiterst efficiënte en toch zeer compacte sensoren en actuatoren beschikbaar.

Kooldioxide wordt op grote schaal gebruikt in verrijgings- en extractieprocessen. Dit proces versnelt niet alleen de groei van planten, maar kan in de meeste gevallen

ook worden gebruikt voor fumigatie. Een CO₂-niveau van 800...1500 ppm wordt gewoonlijk gebruikt bij verrijking om de groei met 20...30% te doen toenemen. Het verhogen van het algemene metabolisme helpt de planten de effecten van hitte te weerstaan. Grotere, gezondere en robuustere planten zullen extremen in het milieu beter verdragen. Een hoger metabolisme betekent echter ook extra eisen. De planten zullen niet alleen meer water en voedingsstoffen nodig hebben, maar ook extra ventilatie.

CO₂-systeemniveaus worden gemeten en online gemonitord met de Adafruit SPG30 luchtkwaliteitssensor, aangesloten op een Würth Elektronik Sensor FeatherWing. Merk op dat de sensor het equivalente CO₂-niveau meet. De eCO₂ wordt berekend op basis van de H₂-concentratie, het is geen 'echte' CO₂-sensor voor laboratoriumgebruik. De volgende stap is het toepassen van de verrijking.

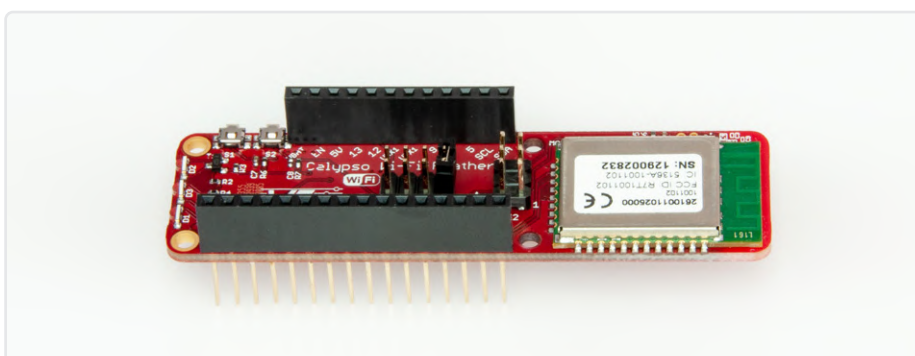
Planten hebben meerdere groeistadia;

in elk van die stadia heeft een plant verschillende licht- en warmteomstandigheden nodig. De meeste planten kunnen normale temperatuurschommelingen verdragen. De optimale thermische omstandigheden voor plantengroei kunnen niet alleen variëren tussen de stadia, maar ook gedurende de dag. De optimalisatie kan gebeuren door de temperatuur te meten en de dagelijkse cycli te optimaliseren, en een paar graden lagere temperatuur wanneer de lichten uit zijn. De luchtvochtigheid wordt uitgedrukt als het vochtigheidspercentage van de lucht. Verschillende plantstadia hebben verschillende vochtigheidsniveaus nodig. Een te vochtige omgeving kan de kans op verspreiding van ziekten vergroten. Zowel vochtigheid als temperatuur worden gecontroleerd met behulp van de WE Sensor FeatherWing.

IoT-connectiviteit

Sensoren en actuatoren worden doorgaans geïnstalleerd in apparaten met beperkte toegang tot de digitale wereld. Draadloze connectiviteit biedt – naast veel andere voordelen – de bereikbaarheid die voor dergelijke toepassingen noodzakelijk is. Er is tegenwoordig een grote verscheidenheid aan standaarden en merkgebonden oplossingen voor draadloze connectiviteit beschikbaar. Een aantal factoren zoals bereik, doorvoercapaciteit, spectrumvoorschriften, lokale wetgeving en stroomverbruik bepalen de keuze van de draadloze connectiviteitsoplossing.

De connectiviteit is geïmplementeerd met behulp van twee verschillende benaderingen: aan de ene kant met een Calypso WiFi FeatherWing (figuur 6) voor omgevingen met een bruikbaar WiFi-netwerk en aan de



Figuur 6. De Calypso FeatherWing biedt connectiviteit in omgevingen waar een WiFi-netwerk beschikbaar is (bron: Würth Elektronik eiSos).

andere kant met een Adrastea-I Feather-Wing (**figuur 7**) voor omgevingen zonder WiFi. Beide zijn verbonden met de rest van het systeem via het Adafruit M0 Express Feather-board.

De Calypso is een compacte WiFi-radio-module op basis van IEEE 802.11 b/g/n (2,4 GHz). Een on-board TCP/IP-stack en MQTT-protocol zijn out-of-the-box geïmplementeerd. De Adrastea-I module is een compacte LTE-M/NB-IoT cellulaire module met geïntegreerde GNSS en een ARM Cortex-M4 processor, geschikt voor elke IoT-toepassing.

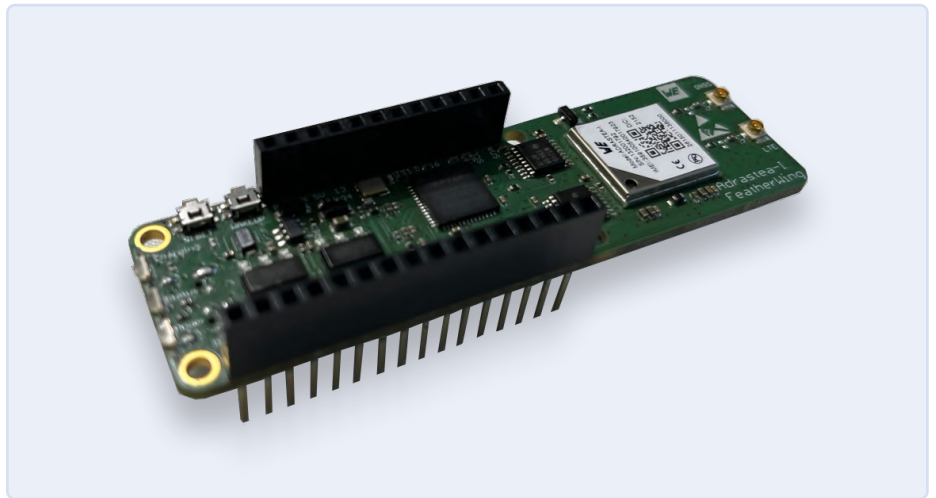
Beide modules kunnen eenvoudig en veilig met elke cloud worden verbonden. In dit geval is gekozen voor een externe M0-processor en Microsoft IoT Central, een Platform-as-a-service, vanwege de eenvoud en het gebruiksgemak. IoT Central heeft een gebruiksklare UI en API gebouwd om IoT-apparaten aan te sluiten, te beheren en te bedienen. Met zijn telemetrie, eigenschappen en opdrachten werd het gebruikt om alle aspecten van het vertical farming-systeem te monitoren en te besturen.

AI-ondersteunde benadering

Verder kan AI worden gebruikt om patronen te detecteren en plantprofielen in de cloud aan te maken, waarmee de LED-waarden, temperatuur, vochtigheid en andere parameters automatisch kunnen worden ingesteld om optimale omstandigheden te realiseren.

Verticale landbouw alleen zal de wereld niet voeden, maar het zal een groter aantal mensen van meer verse producten voorzien. Met deze aanpak werd een miniatuurversie van een verticaal landbouwsysteem gedemonstreerd wat mensen in hun keuken kunnen opstellen. Er was ooit een tijd dat zulke dingen bloembakken werden genoemd... 

230077-03



Figuur 7. De Adrastea-I module is een compacte LTE-M/NB-IoT cellulaire module met geïntegreerde GNSS en een ARM Cortex-M4 processor (bron: Würth Elektronik eiSos).

Over de auteurs



Miroslav Adamov studeerde natuurkunde en informatica aan de Universiteit van Belgrado, Servië. Daarna zette hij zijn wetenschappelijk werk voort aan de TU-Berlijn, WIAS Berlijn, FAU Erlangen/Nürnberg en Center of Private Equity Research in München. Na een paar jaar in kwantitatieve financiën kwam hij in 2015 bij Würth Elektronik als Senior Business Analyst. Zijn functie als Senior IoT Solution Architect nam hij over in 2017, met primaire focus op conceptualisatie en implementatie van industriële IoT-oplossingen.



Adithya Madanahalli studeerde af aan de Technische Universiteit München met een MSc. in Communication's Engineering. Daarna werkte hij enkele jaren als software engineer op het gebied van draadloze connectiviteit en sensoren. Sinds 2022 is Adithya IoT Engineer bij Würth Elektronik eiSos in de business unit voor draadloze connectiviteit en sensoren. Daar is hij gespecialiseerd in het ontwerp en de ontwikkeling van IoT-oplossingen die zich richten op hardware, embedded software en end-to-end beveiliging.

WEBLINKS

- [1] "The future of food and agriculture: Trends and challenges," a report by the Food and Agriculture Organization of the United Nations, 2017 [PDF]: <https://fao.org/3/a-i6583e.pdf>
- [2] "Land statistics and indicators – Global, regional and country trends, 2000-2020," Faostat Analytical Brief 48 [PDF]: <https://fao.org/3/cc0963en/cc0963en.pdf>
- [3] E. Ambrose, "Digital agriculture: Why the future is now." : <https://ag.purdue.edu/stories/digital-agriculture-why-the-future-is-now/>
- [4] Würth Elektronik Lighting Development Kit: https://we-online.com/en/components/products/LIGHTING_DEVELOPMENT_KIT
- [5] V. Brunton, "Irrigation water quality" [PDF]: https://dpi.nsw.gov.au/__data/assets/pdf_file/0005/433643/Irrigation-water-quality.pdf

Ondanks de talloze uitdagingen die de COVID-19 pandemie en de recente geopolitieke gebeurtenissen met zich meebrengen, is er een breed scala aan interessante zakelijke mogelijkheden voor leidinggevend, ondernemers, professionele technici en studenten die zich richten op zowel embedded technologieën als kunstmatige intelligentie-gerelateerde toepassingen. Laten we eens kijken naar een paar feiten en cijfers.

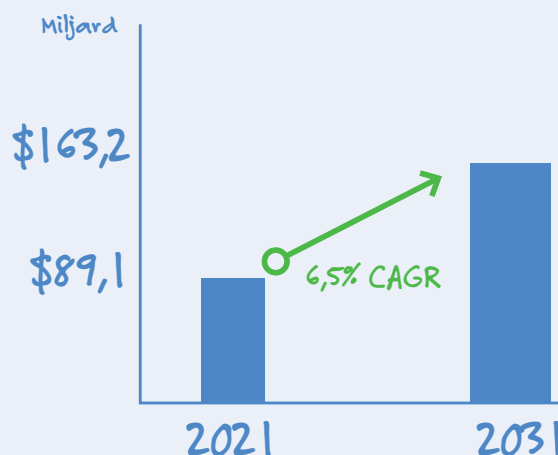
Embedded systemen: een vooruitblik

Benieuwd naar de toekomst van de embedded systems-industrie? Er lijkt groei in het verschiet te liggen. Volgens een recent rapport van Allied Market Research is de industrie op weg om in 2031 uit te groeien tot meer dan 163 miljard dollar, met de regio Azië/Stille Oceaan als koploper [1].

Bedrijven om in de gaten te houden

- Advantech Corp.
- Analog Devices
- Infineon Technologies
- Intel Corp.
- Microchip Technology
- NXP Semiconductors
- Qualcomm
- Renesas Electronics
- STMicroelectronics
- Texas Instruments

(Bron: Allied Market Research)



Kunstmatige intelligentie in beeld

20%

Ongeveer 1 op de 5 afgestudeerden in computer-wetenschappen in 2020 specialiseerde zich in AI/ML.

\$ 93,5 miljard

De particuliere investeringen in AI liepen in 2021 op ongeveer 93,5 miljard dollar, meer dan het dubbele van het bedrag in 2020.

2x

Het aantal Engelstalige AI-gerelateerde publicaties (zoals tijdschriften, boeken, scripties en conferenties) verdubbelde (wereldwijd) van 162.444 in 2010 tot 334.497 in 2021.

Noord-Amerika

Van 2010 tot 2021 werd 56,96% van de toegekende AI-octrooien aangevraagd in Noord-Amerika. 31,09% kwam uit Azië/Stille Oceaan. 11,27% kwam uit Europa en Centraal-Azië.

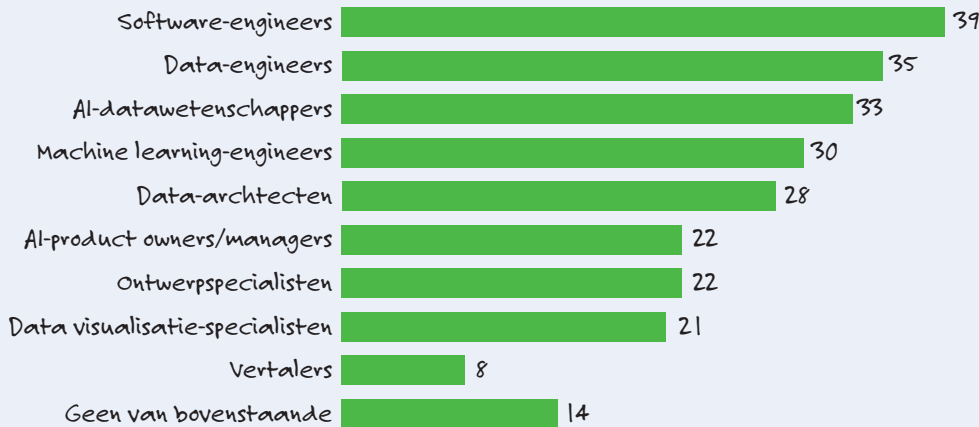
30x

Het aantal ingediende AI-octrooien was in 2021 meer dan 30 keer hoger dan in 2015. Dat is een jaarlijkse groei van 76,9%.

Als het wereldwijde marktonderzoeksbureau IDC gelijk heeft, zal de wereldwijde markt voor artificial intelligence (AI) – inclusief software, hardware en services – in de periode 2022...2026 een samengestelde jaarlijkse groei van 18,6% vertonen om in 2026 een omvang van 900 miljard dollar te bereiken [2]. Kansen voor bedrijven, professionele engineers, ondernemers en zelfs makers zijn er in overvloed. Bekijk deze belangrijke gegevens uit een rapport van 2022 van de Stanford University [3]. De groei van AI is duidelijk te zien.

Waar zijn de banen in AI?

AI-gerelateerde jobs die vorig jaar ingevuld werden bij de respondenten

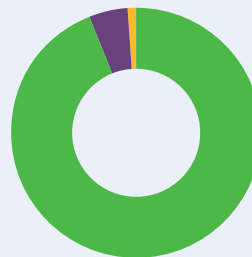


(Bron: www.mckinsey.com)

De toekomst is rooskleurig voor software-engineers! Volgens het recente rapport "State of AI" van McKinsey stonden software-engineers de afgelopen maanden vooraan bij de zich op AI richtende werkzoekenden die werden aangenomen. Data-engineers en AI-datawetenschappers kwamen op de tweede en derde plaats [4]. Dit is een ander duidelijk teken dat veel organisaties grotendeels zijn overgestapt van experimenteren met AI naar het actief integreren ervan in bedrijfsapplicaties.

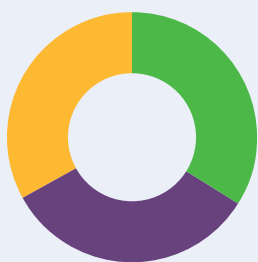
Wat denken zakelijke leidinggevenden over AI?

We weten inmiddels allemaal dat artificial intelligence big business is. Maar ben je benieuwd hoe leidinggevenden in het bedrijfsleven denken over AI? Hoewel de belangstelling afhankelijk van de bedrijfstak verschilt, zijn AI-oplossingen zeker een gespreksonderwerp onder leidinggevenden bij conferenties, tijdens bestuursvergaderingen en op bedrijfsuitjes. De volgende gegevens van Deloitte bieden enig inzicht [5].



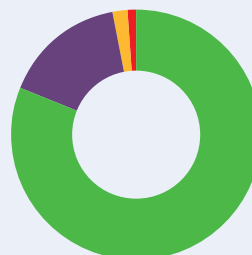
Belang van AI voor het succes van een organisatie

- 94% Zeer belangrijk of belangrijk
- 5% Enigszins belangrijk
- 1% Niet belangrijk



Wat zijn de drie grootste uitdagingen bij het opstarten en realiseren van projecten?

- 30% Gebrek aan financiering voor AI-technologie
- 29% Selectie van de juiste AI-technologieën
- 29% Onvoldoende technische vaardigheden



Zal het gebruik van AI-technologie de arbeidsprestaties en tevredenheid verbeteren?

- 82% Mee eens
- 16% Noch mee eens noch oneens
- 2% Oneens
- 1% Weet niet zeker

WEB LINKS

- [1] CISION, "Embedded Systems Market to Reach \$163.2 Billion by 2031, Globally, by 2031 at 6.5% CAGR," Allied Market Research, 2022: <https://bit.ly/embedded-outlook-AMR>
- [2] IDC, "IDC Forecasts 18.6% Compound Annual Growth for the Artificial Intelligence Market in 2022-2026," 2022: <https://bit.ly/IDC-AI-2022-2026>
- [3] D. Zhang, et al, "The AI Index 2022 Annual Report," HAI, Stanford University, March 2022: <https://bit.ly/AI-index-rep-22>
- [4] McKinsey, "The State of AI in 2022 — And a Half Decade in Review," QuantumBlack, Dec 6, 2022: <https://bit.ly/mckinsey-AI-22>
- [5] Deloitte, "Fueling the AI transformation," October 2022: <https://bit.ly/deloitte-ai-state>

Een introductie tot TinyML

Mark Patrick (Mouser Electronics)

Dit artikel onderzoekt concepten van machine learning (machinaal leren) met behulp van eenvoudige, low-power microcontrollers, onder de noemer TinyML. Machine learning blijft zijn stempel drukken op vele aspecten van ons dagelijks leven, thuis, op kantoor of waar dan ook. Hoewel veel applicaties voor machine learning aanzienlijke rekenkracht vereisen om complexe wetenschappelijke of financiële gegevens te kraken, bieden ze slechts beperkte reken- en verbindingsmogelijkheden voor het internet der dingen (IoT) en andere randtoepassingen.

AI en machinaal leren: integraal onderdeel van ons leven

Praten tegen ons polshorloge of een Star Trek communicator-gadget was nog niet zo lang geleden science fiction, maar nu is dat precies wat velen van ons dagelijks doen. We praten met onze smartphone-apps, met het infotainment-systeem van onze auto's en onze slimme "smart speakers" thuis.

Kunstmatige intelligentie (AI) en het creëren van zelfdenkende computers die voelen, herkennen en problemen oplossen, vormen nu de basis van de hedendaagse computer- en datawetenschap. Machine learning (ML) is een toepassingsgebied van AI en richt zich op het gebruik van algoritmen die computers in staat stellen te leren hoe ze een taak beter moeten uitvoeren zonder expliciet daarvoor te zijn geprogrammeerd.

Een groot deel van de dagelijkse realiteit wordt al vormgegeven door machine learning, van weersvoorspellingen en het vinden van de beste route voor het

woon-werkverkeer tot de reclamebanners en advertenties op je favoriete social media-app. Veel gebieden van wetenschappelijk onderzoek zijn nu afhankelijk van ML om petabytes aan gegevens te doorzoeken op trends.

Machine learning – hoe werkt het?

Van de hierboven genoemde voorbeelden van machine learning zien we meestal alleen de resultaten. We hebben vaak geen idee van de complexiteit achter het vinden van een nieuw zwart gat bijvoorbeeld, of het voorspellen van het weer aan de hand van het aantal permutaties van eerdere weersomstandigheden. Veel complexe taken vereisen grote datasets, meerdere algoritmen en een aanzienlijke rekenkracht. Als je je verder verdiept in machine learning, vind je verschillende categorieën van leren, elke geschikt voor een specifieke taak. We gaan er hier niet dieper op in, maar ze worden onderverdeeld in "supervised", "unsupervised", "semi-supervised" en "reinforcement learning". Voor elke ML-toepassing is een neurale netwerk cruciaal. Simpel gezegd, een neurale netwerk is een poging om de functie van de neuronen in het menselijk brein te repliceren in een wiskundig model. Dit model maakt gebruik van algoritmen om de waarschijnlijkheid van een resultaat af te leiden. Bij beeldherkenning bijvoorbeeld is er 95% waarschijnlijkheid dat de afbeelding op de foto een hond is. Er zijn verschillende soorten neurale netwerken en je komt termen tegen als convolutionele neurale netwerken (CNN) en terugkerende neurale netwerken (RNN). Elk soort neurale netwerk heeft een andere set onderling verbonden lagen, waardoor de één beter geschikt is voor een specifieke taak dan de ander.

Bij supervised learning wordt het neurale netwerk gevoed met trainingsgegevens, zodat een algoritme een resultaat kan afleiden. Voor beeldherkenning bijvoorbeeld is een CNN het beste. Om verschillende soorten fruit te herkennen, moet je het neurale netwerk-algoritme voeden met duizenden gelabelde

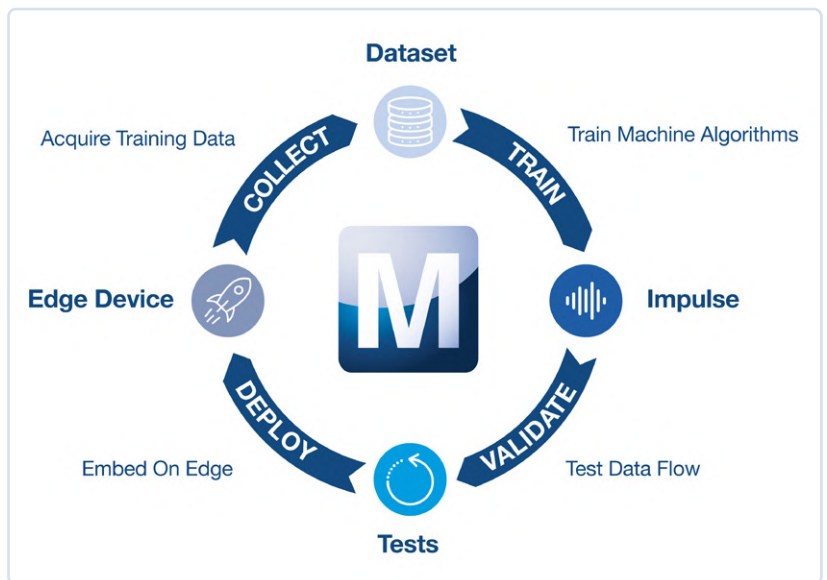
beelden van verschillende soorten fruit met diverse gradaties van rijpheid, gefotografeerd vanuit verschillende hoeken. Het algoritme zoekt dan naar herkenbare kenmerken waarmee het de diverse typen fruit van elkaar kan onderscheiden. De trainingsfase is iteratief en kan inhouden dat het algoritme, wanneer het wordt geconfronteerd met een testdataset van beelden, wordt bijgesteld om de hoogste waarschijnlijkheid te bereiken.

Zodra de testgegevens de beste algoritme-prestaties van het neurale netwerk opleveren, is het model klaar voor gebruik. Bij inzet, ook bekend als inferentie, leidt het model resultaten af op basis van waarschijnlijkheid. Onze dagelijkse interactie met bijvoorbeeld onze slimme speaker omvat meestal een triggerwoord of -zin, zoals "Hé, Google" om hem te wekken, zodat hij begint te luisteren naar wat er gaat komen. Een slimme speaker beschikt niet over de rekencapaciteit van een datacentrum, dus worden er korte audiobestanden opgenomen en naar de cloud gestreamd voor inferentie om de aard van ons verzoek te bepalen. Het detecteren van het triggerwoord of de triggerzin is een uitstekend voorbeeld van eenvoudig machine learning. Welkom bij TinyML!

Machine learning en het industriële IoT

Nu machine learning alomtegenwoordig wordt, groeit de potentiële lijst van applicaties exponentieel. Veel industriële toepassingen zijn bijvoorbeeld niet gericht op "big data", maar op de vraag hoe we productielijnen efficiënter kunnen maken. De kosten van een onverwachte storing in een productieproces kunnen stevig oplopen. Een motorstoring tijdens een gekoeld mengproces van voedingsmiddelen kan de productie stilleggen, waardoor er grondstoffen verloren gaan. Veel organisaties passen nu predictief onderhoud toe door een stilstand te plannen om dergelijke storingen te voorkomen. Toestandsafhankelijk monitoren van productiemiddelen, zoals motoren en aandrijvingen, helpt bij het voorspellen wanneer bijvoorbeeld een motorlager tekenen van overmatige slijtage gaat vertonen. Machine learning-algoritmen in IIoT-edgesensoren kunnen vaststellen wanneer de trillings-signatuur van een motor afwijkt van de norm en zo tijdig waarschuwen.

Alleen al binnen het industriële domein zijn de toepassingen voor machine learning enorm, maar deze gaan ook gepaard met verschillende technische uitdagingen. In tegenstelling tot de "big data"-toepassingen zijn de reken- en geheugenbronnen van een eenvoudige IIoT-edgesensor slechts een fractie van die in een datacenter. Een fabriek kan honderden sensoren hebben, dus schaalvoordeel is een factor, evenals de fysieke grootte, de beschikbaarheid van een geschikte stroomvoorziening en bekabelde of draadloze connectiviteit. In de meeste gevallen is het neurale netwerk-



Figuur 1. Training en inferentie.

algoritme van een trillingssensor minder veeleisend dan bijvoorbeeld sterrenkundig onderzoek, dus zoeken embedded ontwikkelaars naar manieren om netwerkmodellen te draaien op low-power microcontrollers met batterijvoeding.

TinyML: machine learning aan de edge

Het voor de training van een neurale netwerkmodel gebruikte computerplatform hoeft niet hetzelfde te zijn als het platform dat wordt ingezet, waardoor een van de meest hulpmiddel-intensieve processen wegvalt. Er zijn echter nog steeds technische uitdagingen. Kan het algoritme tijdig worden uitgevoerd? Hoe communiceert het apparaat met een hostsysteem om het operationele personeel te informeren? Ook de embedded ontwikkelaars worden met uitdagingen geconfronteerd. De meesten van hen zijn geen datawetenschappers, dus het leren van de concepten van machine learning en het werken met neurale netwerken vormen een steile leercurve. [▶](#)

230062-03



Over de auteur

Als Technical Marketing Manager voor EMEA bij Mouser Electronics is Mark Patrick verantwoordelijk voor het maken en verspreiden van technische content in die regio – content die essentieel is voor de strategie van Mouser om zijn technische publiek te ondersteunen, te informeren en te inspireren. Voordat hij leiding gaf aan het Technical Marketing-team, maakte Patrick deel uit van het EMEA Supplier Marketing-team en speelde hij een cruciale rol bij het aangaan en ontwikkelen van relaties met belangrijke productiepartners. Naast een verscheidenheid aan technische en marketingfuncties, heeft Patrick onder andere acht jaar bij Texas Instruments in Applications Support en Technical Sales gewerkt. Als 'hands-on' engineer in hart en nieren, met een passie voor vintage synthesizers en motorfietsen, draait hij zijn hand er niet voor om die te repareren. Patrick heeft een eerstegraads Honours Degree in Electronics Engineering van Coventry University.



KwickPOS

Tevreden klanten: op de Raspberry Pi gebaseerd kassasysteem is een hit in drukke restaurants en winkels.

Het in Houston gevestigde KwickPOS had zijn eerste klant al voordat het in 2003 actief werd. De eigenaren van een Chinees restaurant die op zoek waren naar een handige manier om bestellingen van klanten te verwerken, benaderden hun bevriende IT-ontwikkelaars Tom Jin en Ming Ye. Jin en Ye, die

20 jaar ervaring hebben in Silicon Valley met het ontwikkelen van restaurantsoftware, gingen snel aan de slag met het maken van een op Linux gebaseerd verkoopsysteem dat bestellingen kon weergeven en verwerken.

Ze begonnen met Chinese restaurants – “de meest uitdagende restaurants om te beheren vanuit een menu-perspectief en vanuit het oogpunt van organisatorische feedback

binnen het restaurant,” aldus Ming Ye – en het raakte al snel bekend bij andere restauranthouders. Er was voldoende vraag om van het concept een zakelijke onderneming te maken.

De uitdaging

Een succesvol en veelgevraagd systeem voor verkooppunten omzetten in een volwaardig bedrijf was voor Jin en Ye een hele uitdaging, gezien hun bestaande carrières in Silicon Valley. Toch besloten ze in 2015, na een aantal van de systemen te hebben gebouwd, om van KwickPOS een ‘echt bedrijf’ te maken, door POS-systemen te ontwikkelen en te verkopen vanuit hun hoofdkantoor in Houston, Texas. Nadat ze backoffice-ondersteuning en servicestrategieën hadden toegevoegd aan hun POS-product, begonnen ze hun bedrijf te laten groeien via kanaalverkoop, distributeurs, handelsdiensten en andere organisaties die verkoopverwerkingsmogelijkheden nodig hebben. Dure Windows-gebaseerde servers en kiosken hadden hun beperkingen, niet in het minst vanwege de kosten en de waarde van de hardware voor dieven.





Het bedrijf heeft nu meer dan 2000 klanten met KwickPOS-systemen in hun restaurants

Ze nemen ook nogal wat ruimte in beslag. “Sommige point-of-sale-systemen voor restaurants hebben drie of vier terminals, waarvan één de eigenlijke server is,” legt Ye uit. Meer dan eens is de laptop of desktopcomputer ten prooi gevallen aan een opportunistische dief, waardoor het betrokken bedrijf geen mogelijkheid had om bestellingen op te nemen en uit te voeren.

Een ander probleem voor winkels en restaurants is wat er gebeurt als de internetverbinding uitvalt, een terechte zorg in delen van de VS waar stroomuitval niet ongewoon is. Nu steeds meer bestellingen via e-mail of online binnenkomen, zijn bedrijfscontinuïteit en uptime belangrijker dan ooit, evenals de mogelijkheid om betalingen te verwerken.

De oplossing

Bij de lancering van de Raspberry Pi zag Jin onmiddellijk het potentieel om KwickPOS op dit platform te laten draaien en het te gebruiken als server en als een onopvallende maar krachtige terminal in vaak krappe klantenkiosken en restaurants, waar de eetruimte en het aantal couverts gemaximaliseerd moet worden. Aangezien de bestaande backend-software van het bedrijf sinds 2013 op Linux was gebaseerd, was de overstap naar de Raspberry Pi commercieel gezien een goede zaak, vooral omdat de Compute Module slechts het tiende van een Windows-omgeving kost. KwickPOS kon volledig gebruik maken van de Raspberry Pi voor zijn cloud-based server.

In tegenstelling tot de meeste POS-systemen is KwickPOS browser-gebaseerd, met een server op locatie – iets wat mogelijk is door het compacte formaat van de Compute Module – en de weergave van de POS-applicatie op de terminals wordt gerepliceerd naar de cloud. Een Raspberry Pi-server in elke terminal verwerkt de betalingsgegevens. Deze opzet heeft een cruciaal voordeel: de offline modus. “Als het internet uitvalt, is dat prima”, aldus Ye. “De manager kan vanaf de terminal inloggen, deze in offline modus zetten en het restaurant blijven bedienen.” Als het internet terugkomt, zetten ze hem in online modus en worden alle reeds verwerkte transacties gerepliceerd en afgerond voor het restaurant. “Uptime is een groot concurrentievoordeel.”

Waarom de Raspberry Pi?

Jin waardeert het feit dat de Raspberry Pi Compute Module zo stabiel is. Hij is gefrustreerd door het voortdurende idee dat het gebruik van Windows ‘als een wet’ is voor bedrijven, met veel klanten die het nog steeds gebruiken, ondanks dat de server ervan stelsel-

matig het zwakke punt in hun opstelling blijkt te zijn. KwickPOS-diensten gebruiken zowel Compute Module 3 als 4.

Vanuit het oogpunt van de klant is het gebruik van de Raspberry Pi ook een goede keuze. “Klanten zijn erg blij met dit apparaat”, aldus Ye. Om te beginnen neemt het niet veel ruimte in beslag in een onvermijdelijk krappe restaurantruimte en is het discreet genoeg om niet de aandacht te trekken van opportunistische dieven. In het zeldzame geval dat een apparaat wordt gestolen, kan het bedrijf gewoon doorgaan, omdat de Raspberry Pi webserver de taak overneemt. Het systeem werkt nog steeds alsof het is aangesloten en de klant kan een vervangend apparaat plaatsen zonder gegevens te verliezen.

KwickPOS heeft zijn klantenbestand ook uitgebreid met boetieks. Net als bij zelfstandige restaurants is de noodzaak om snel en efficiënt bestellingen op te nemen en te verwerken van het grootste belang. Belangrijk is dat KwickPOS platformafhankelijk is. Nu portable betaalterminals gemeengoed zijn geworden, kan KwickPOS worden geïntegreerd in de bestaande opzet van een winkel.

De resultaten

Sinds de overstap naar de Raspberry Pi in 2018 is KwickPOS uitgegroeid tot een organisatie die klanten heeft in 45 Amerikaanse staten, maar ook in Canada en Mexico. Er is zelfs een Chinees restaurant in Londen dat het systeem gebruikt. Het bedrijf heeft nu meer dan 2000 klanten met KwickPOS-systemen in hun restaurants, variërend van zelfstandige vestigingen tot ketens. ◀

WEBLINKS

Succesverhaal: magpi.cc/success



High-performance in elke klasse

computer-on-module standaarden



▲ Een bijdrage van Congatec

Figuur 1. De verschillende modulegroottes van fabrikant-onafhankelijke computer-on-module standaarden.

Er gebeurt veel op de computers-on-module markt. Met COM Express 3.1 is nu een nieuwe versie van de meest succesvolle computer-on-module standaard beschikbaar. En COM-HPC, de standaard voor krachtige embedded computers, roept ook veel nieuwe vragen op. Wat moeten OEM's en systeemontwerpers dus weten?

Volgens cijfers van IHS Markit zijn computer-on-modules het meest gebruikte embedded ontwerp principe – zelfs vóór klassieke embedded boards zoals Mini-ITX of 3,5-inch single board computers (SBC's). De grote populariteit van dergelijke embedded systeemont-

werpen vloeit voort uit de succesvolle combinatie van flexibele, klantspecifieke carrier board-ontwerpen met kant-en-klare, eenvoudig te integreren modules die alle noodzakelijke drivers en firmware bevatten. Deze supercomponenten omvatten alle belangrijke bouwstenen zoals CPU, RAM, snelle interfaces en vaak ook de grafische eenheid in één enkel functionerend pakket. Een ander voordeel is dat computers-on-modules van dezelfde standaard vrij uitwisselbaar zijn, zowel tussen processorgeneraties als tussen fabrikanten. Dit geeft OEM's volledige flexibiliteit bij het opschalen en upgraden van hun oplossingen met de nieuwste procestechnologie, zelfs na enkele jaren. Het maakt het ook gemakkelijk om multi-vendor strategieën toe te passen, wat prijsvoordelen biedt en bovenal beschikbaarheid garandeert. Er zijn twee onafhankelijke comités die zich bezighouden met de standaardisatie van modules: de Amerikaanse PICMG en het Duitse SGET onderhouden momenteel in totaal vier computer-on-module normen, waarvan de meeste talrijke varianten bieden. Deze standaarden zijn COM-HPC en COM Express voor het high-end segment, en SMARC en Qseven voor het low-power segment.

COM-HPC: de high-performance benchmark

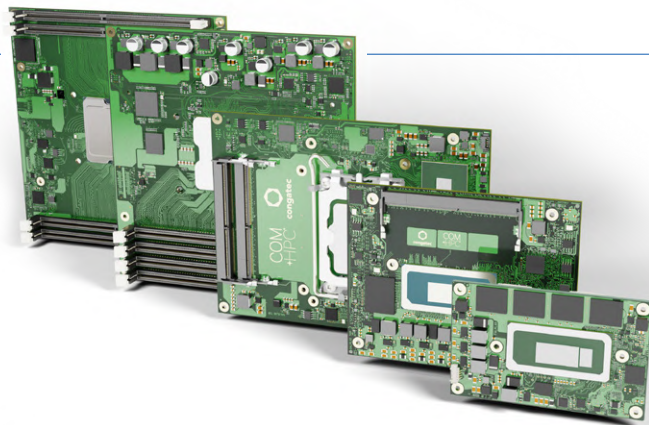
COM-HPC, de meest recente computer-op-module standaard van de PICMG, is gericht op krachtige embedded computerontwerpen, die met eerdere standaarden niet haalbaar waren. De prestaties liggen boven die van de toonaangevende COM Express-standaard. COM-HPC richt zich op nieuwe snelle interfaces zoals PCI Express 4.0 en 5.0, alsmede 25Gb-Ethernet. Hiervoor worden twee verschillende module-versies aangeboden, die zijn ontwikkeld door het COM-HPC subcomité onder leiding van zijn voorzitter, Christian Eder van Congatec: COM-HPC Server en COM-HPC Client. Hun belangrijkste technische verschillen betreffen de footprints, het aantal en type ondersteunde interfaces en de geheugencapaciteit. Geheel nieuw is de COM-HPC Mini-specificatie, die deze voordelen nu ook op creditcardformaat biedt.

COM-HPC Server-modules

COM-HPC Server definieert het ultra-high-end van embedded computing en richt zich op de nieuwe edge- en fog-servers in ruwe omgevingen die meer en massievere workloads aan moeten kunnen. Daartoe specificeert COM-HPC Server twee footprint-formaten met maximaal 64 PCIe-lanes en maximaal 256 GB/s, alsmede maximaal 8x Ethernet met elk 25 Gb/s. COM-HPC Server is niet beperkt tot x86-technologie, maar staat het gebruik van RISC-processoren, FPGA's en GPGU's toe – wat nieuwe modularisatie-perspectieven toevoegt. Om te voldoen aan de eisen van server-toepassingen bieden de modules ook master-slave modi en beheer op afstand. Door gebruik te maken van de instructieset van de krachtige IPMI-standaard komt gevestigde servertechnologie ook beschikbaar voor Server-on-Modules. Met COM-HPC Server-modules met een energiebudget tot 300 watt is deze standaard geschikt voor de ontwikkeling van ultra-high-performance embedded edge- en fog-servers. Ter vergelijking: de krachtigste COM Express Type 7 Server-on-Modules van vandaag ondersteunen maximaal 100 watt. Een ander belangrijk verschil ligt in het aantal signaalpinnen: de COM Express-connector heeft 440 pinnen, terwijl COM-HPC er met 800 bijna twee keer zoveel heeft.

COM-HPC Client-modules

COM-HPC Client-modules zijn ontworpen voor krachtige embedded systemen met geïntegreerde graphics. Ze bezitten vier grafische uitgangen: drie Digital Display-Interfaces (DDI) en één embedded DisplayPort (eDP). Ze bieden plaats aan maximaal vier SO-DIMM-modules voor maximaal 128 GB RAM. Voor de aansluiting van randapparatuur zijn 48 PCIe-lanes en 2x USB 4.0 beschikbaar; embedded cameramodules kunnen ook rechtstreeks worden aangesloten via



twee MIPI-CSI interfaces. COM-HPC Client-modules komen beschikbaar in drie verschillende afmetingen: 120 × 160 mm (maat C), 120 × 120 mm (maat B) en 120 × 95 mm (maat A).

COM-HPC Mini-modules

COM-HPC Mini is een nieuwe computer-on-module standaard die nu in ontwikkeling is bij de PICMG. Momenteel heeft de technische subcommissie de pinout en footprint goedgekeurd van de nieuwe krachtige computer-on-module specificatie ter grootte van een creditcard (95 × 60 mm). Met 400 pinnen is de nieuwe COM-HPC Mini-standaard ontworpen om te voldoen aan de toenemende interfacebehoeften van heterogene en multifunctionele edgecomputers. De uitbreidingen omvatten tot 4x USB 4.0 met volledige functionaliteit inclusief Thunderbolt en DisplayPort alternatieve modus, PCIe Gen 4/5 met tot 16 lanes, 2x 10 Gb/s Ethernet poort en nog veel meer. Voeg daarbij het feit dat de COM-HPC Mini-connector gekwalificeerd is voor bandbreedtes van meer dan 32 Gb/s – genoeg om PCIe Gen 5 of zelfs Gen 6 te ondersteunen – en het is duidelijk dat de mogelijkheden veel verder gaan dan die van alle andere module-standaarden op creditcard-formaat.

COM Express: de meest succesvolle modulestandaard wereldwijd

Dit betekent dat de kleinste footprint van COM-HPC Client met 125 × 95 mm bijna even groot is als die van COM Express Basic. Dit illustreert dat COM-HPC Client ver boven COM Express staat en gericht is op toepassingen die niet met COM Express kunnen

▲
Figuur 2. Congatec biedt een uitgebreid COM-HPC portfolio, van COM-HPC Server en Client tot COM-HPC Mini.



◀
Figuur 3. De nieuwste processorstechnologie zoals de high-end 13^e generatie Intel Core-processoren in BGA-behuizing monteert Congatec op zowel COM-HPC als COM Express modules.

worden aangepakt. COM Express werd in 2005 gelanceerd en is van de hier gepresenteerde computer-on-module standaarden het langst beschikbaar. De specificatie definieert een familie met verschillende modulegroottes en pintypes. In tegenstelling tot COM-HPC en de specificaties voor modules met een kleine vormfactor (SFF) Qseven en SMARC, richt COM Express zich uitsluitend op x86-processortechnologie. Met de bekrachtiging van de 3.1-specificatie ondersteunt COM Express nu de nieuwste snelle interfaces, zoals PCIe 4.0 en USB 4. Ondanks de verbeteringen zijn COM Express 3.1 Type 6-modules volledig achterwaarts compatibel met 3.0-modules en carrier boards, zodat zelfs oudere ontwerpen kunnen worden uitgerust met de nieuwste processortechnologieën.

COM Express Type 7 Server-on-Module

Net als COM-HPC biedt COM Express zowel server- als clientmodules, die voornamelijk beschikbaar zijn in de bekende Type 6 (client) en Type 7 (server) pinouts. Net als bij COM-HPC is de Type 7-serverpinout een headless Server-on-Module zonder grafische uitgangen, eveneens ontworpen voor embedded edge- en fog-servers. Hij ondersteunt met name tot 4×10 GbE en tot 32× high-speed PCIe Gen 3.0 lanes voor interfaces en opslagmedia. Deze server-on-modules zijn beschikbaar met Intel Xeon D-processoren of AMD EPYC Embedded 3000 processoren. Daarvoor biedt Congatec ook een 100-watt ecosysteem met gebruiksklare koeloplossingen om het ontwerp van de krachtigste COM Express server-on-modules te vereenvoudigen.

COM Express Type 6 computer-on-modules

Voor klassieke embedded-toepassingen met grafische voorzieningen zijn PICMG COM Express Type 6-modules de ideale keuze. Ze zijn voorzien van embedded processoren variërend van Intel Core, Pentium en Celeron tot de AMD Embedded R-serie. Verkrijgbaar met afmetingen van 95 × 125 mm (Basic) of 95 × 95 mm (Compact), bieden ze 440 pinnen naar de carrierprint voor een breed scala

aan moderne computerinterfaces. Met ondersteuning voor maximaal vier onafhankelijke beeldschermen, 24 PCIe-lanes, USB 2.0, n USB 3.0 en Ethernet, CAN-bus en seriële interfaces, bieden ze alles wat nodig is om krachtige PLC's, HMI's, systemen op de werkvloer of SCADA-werkstations in controlekamers te bouwen. Andere toepassingsgebieden zijn high-end digital signage-systemen en hoogwaardige medische apparatuur voor diagnostische beeldvorming. Met de spec-update naar 3.1 kunnen COM Express Type 6-modules nu optioneel worden voorzien van MIPI-CSI connectoren op de module.

COM Express Type 10 Mini-modules

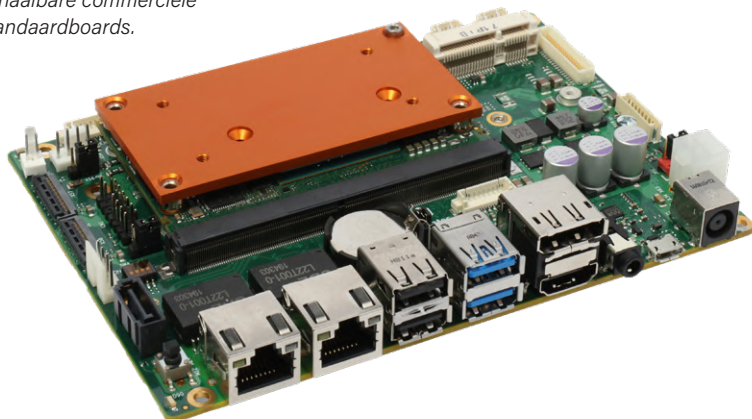
De kleinste vormfactor van de specificatie – COM Express Mini van 55 × 84 mm – wordt beschreven door de PICMG type 10 pinout en completeert de set COM Express-specificaties voor SFF-ontwerpen. Deze modules zijn ontworpen voor low-power Intel Atom- en Celeron-processoren. Ook hier kunnen de modules twee MIPI CSI-connectoren met COM Express 3.1 aansturen. Omdat in het gehele PICMG COM Express-ecosysteem gebruik wordt gemaakt van dezelfde connectortechnologie en ontwerp-richtlijnen, zijn ontwikkelaars in staat een aanzienlijk aantal functies te hergebruiken, hetgeen het belangrijkste voordeel is van deze mini-specificatie. De SGET-standaarden, SMARC en Qseven, zijn echter meer ingeburgerd en worden vaker gebruikt; beide standaarden ondersteunen zowel x86- als ARM-applicatieprocessoren.

SMARC voor embedded vision

SMARC richt zich op het hoge segment van SFF-toepassingen. Deze standaard heeft een belangrijke update gekregen met revisie 2.1. De nieuwe revisie voegt talrijke nieuwe functies toe, zoals SerDes-ondersteuning voor uitgebreide edge-connectiviteit en twee extra interfaces op de module die kunnen worden gebruikt om in totaal 4 MIPI-CSI-camera's aan te sluiten om tegemoet te komen aan de groeiende vraag naar een combinatie van embedded computing en embedded vision. De nieuwe functies zijn achterwaarts compatibel met Rev. 2.0 en alle uitbreidingen op Rev. 2.0 zijn optioneel, zodat alle Congatec SMARC 2.0-modules automatisch compatibel zijn met SMARC 2.1.

Een ander onderscheidend kenmerk, naast de twee MIPI-interfaces op de connector, is dat SMARC draadloze interfaces zoals WLAN en Bluetooth rechtstreeks op de module ondersteunt. Ideale processoren voor SMARC-modules zijn de nieuwste Intel Atom-processoren of de hele reeks nieuwe i.MX 8-applicatieprocessoren. 

Figuur 4. SMARC 2.1 carrier board en schaalbare 3,5-inch single board computer: de conga-SMC1 overbruggt de kloof tussen op modules gebaseerde ontwerpen en uiterst schaalbare commerciële standaardboards.



230071-03

Alle begin...

...wordt eindelijk actief!

Eric Bogers (Elektor)

We hebben het in een eerdere aflevering van deze serie al eens opgemerkt: alle onderdelen waar een halfgeleiderkristal in zit (zoals dioden), worden ‘actieve’ componenten genoemd. Maar eigenlijk klopt dat niet wanneer we het begrip ‘actief’ heel streng nemen: alleen onderdelen die een stroom of spanning versterken, zijn ‘echte’ actieve onderdelen. En dus betreden we pas nu, met de transistor, het territorium van de actieve onderdelen.

Het woord *transistor* is een neologisme dat is samengesteld uit de Engelse woorden *transfer* en *resistor* – met een knipoog zouden we de transistor dus een ‘overdrachtelijke weerstand’ kunnen noemen (maar wees gerust: dat doet niemand). En inderdaad: de weerstand van deze component kan met behulp van een stuurspanning worden gevarieerd. Transistoren hebben de triomftocht van de elektronica pas goed mogelijk gemaakt doordat ze veel kleiner en minder kwetsbaar zijn dan elektronenbuizen, en omdat ze een duidelijk geringer vermogen opnemen. Dankzij transistoren werden geïntegreerde schakelingen (*integrated circuits*, IC's) mogelijk, dat wil zeggen de combinatie van zeer veel (actieve en passieve) onderdelen op een *chip* – bij moderne microprocessoren praten we dan over vele miljoenen transistoren die in één behuizing zijn ondergebracht.

We beginnen echter met de transistor als ‘losse’ component; geïntegreerde schakelingen komen in een latere aflevering aan de orde. Ook al bestaat er tegenwoordig voor veel toepassingen een kant-en-klare oplossing in de vorm van een IC, toch moeten we regelmatig nog een greep in de bak met ‘losse’ transistoren doen – bijvoorbeeld wanneer we met de eindtransistoren van een vermogensversterker te maken hebben.

Figuur 1 toont enkele transistoren. Als vuistregel geldt een



Figuur 1. Enkele transistoren. (bron: Shutterstock / Edkida).

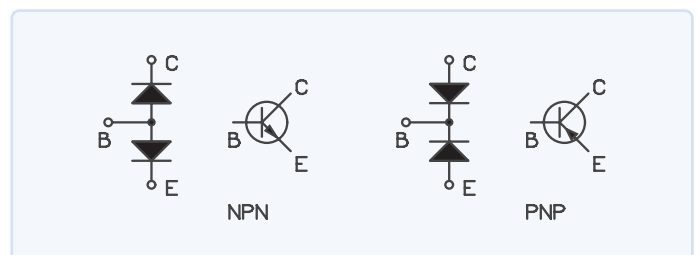
transistor des te meer vermogen kan verwerken naarmate hij groter is en zijn behuizing een ‘metaliger’ indruk maakt.

De bipolaire transistor als schakelaar

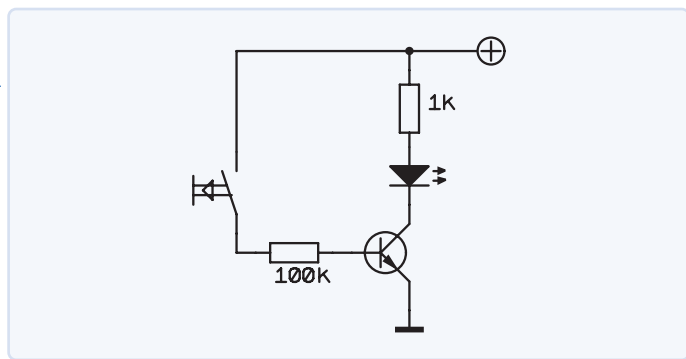
Wanneer we het over een ‘normale’ transistor hebben, bedoelen we een bipolaire transistor die in npn- en pnp-uitvoering verkrijgbaar is; deze letters geven de volgorde aan van de n- en p-gedoteerde siliciumlagen waaruit de transistor is opgebouwd.

Heel sterk vereenvoudigd gesteld kunnen transistoren als schakelaar of als versterker worden gebruikt. Als eerste werpen we een blik op de transistor als schakelaar omdat de werking daarvan gemakkelijker te begrijpen is. Pas daarna zullen we ons om de eigenschappen van de transistor als versterker bekommeren.

In **figuur 2** zien we links een npn-transistor en rechts een pnp-exemplaar, in beide gevallen als een combinatie van twee dioden. In de praktijk wordt een transistor nooit toegepast als een simpele vervanging voor twee diodes, maar wanneer we met een doorgangstester willen controleren of we met een npn- of een pnp-transistor te maken hebben (en of die het nog ‘doet’ dan wel is doorgebrand), dan is het handig wanneer we dit vervangingsschema in gedachten hebben.



Figuur 2. Vervangingsschema van de transistor. C = collector, B = basis en E = emitter.



Figuur 3. De transistor als schakelaar.

Figuur 3 toont een transistor die als schakelaar wordt gebruikt. In de voorbeelden in deze aflevering gebruiken we overigens steeds npn-transistoren omdat die gemakkelijker te begrijpen zijn wanneer u de 'technische stroomrichting' (van plus naar min) in gedachten houdt. Elk voorbeeld kan echter zonder meer naar pnp-transistoren worden vertaald door gewoon alle gepolariseerde onderdelen (dioden, elco's en uiteraard de voedingsspanning) om te polen.

Terug nu naar figuur 3. We hebben hier te maken met twee stroomkringen: een voor de stroom en een voor de belasting. De stroomkring loopt via de schakelaar, de serieweerstand en de basis/emitter-diode van de transistor. Het is via deze BE-overgang dat de transistor wordt gestuurd. Wanneer er een stroom door deze overgang loopt, gaat de transistor in geleiding; als er hier geen stroom loopt, dan spert de transistor.

De BE-overgang gedraagt zich als een gewone diode; pas wanneer de spanning erover een waarde van ongeveer 0,7 V bereikt (net als bij gewone siliciumdiodes) gaat er een stroom lopen.

De andere stroomkring loopt via de serieweerstand van de LED en de LED zelf via de CE-overgang van de transistor. Eigenlijk zou daar geen stroom mogen lopen omdat de CB-diode immers in sperrichting staat. In de praktijk loopt de stroom dan ook niet via de CB-diode, maar rechtstreeks van de collector naar de emitter – zodat er ook geen 0,7 V over de CE-overgang valt.

Dat dit functioneert, is te danken aan de speciale manier waarop de verschillende halfgeleiderlagen zijn gerangschikt – het is ten enen male onmogelijk om met een paar losse dioden een transistor in elkaar te knutselen!

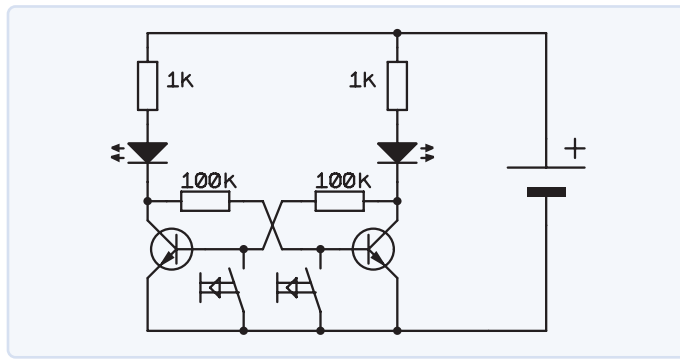
U vraagt zich nu waarschijnlijk (en terecht) af waarom we in de schakeling van figuur 3 een transistor en een extra weerstand gebruiken in plaats van de LED rechtstreeks met de schakelaar te bedienen. U hebt helemaal gelijk: in dit specifieke voorbeeld is dat nergens voor nodig...

Weliswaar is de stroom die door de schakelaar loopt hier een factor 100 of zo kleiner dan de stroom door de LED, maar u zult heel wat moeite moeten doen om een schakelaar te vinden die niet tegen de LED-stroom bestand zou zijn. De schakeling zoals hier geschetst is echter ook geschikt om veel grotere stromen (en spanningen) te schakelen: het is namelijk helemaal niet verplicht om de stroomkring en de belastingkring uit dezelfde spanningsbron te voeden. Dat betekent niets anders dan dat het dankzij de transistor mogelijk is om met een stuurspanning van een paar volt een veel hogere spanning te schakelen – vooropgesteld natuurlijk dat de juiste transistor wordt gebruikt.

De bistabiele multivibrator (flipflop)

We bekijken nu een paar zogenaamde multivibratoren – dat zijn schakelingen die tussen twee gedefinieerde toestanden omschakelen. **Figuur 4** toont de bistabiele multivibrator, die in het dagelijks leven meestal flipflop wordt genoemd.

Deze schakeling wordt bistabiel genoemd omdat hij twee stabiele



Figuur 4. Bistabiele multivibrator.

toestanden kent: ofwel brandt de linker LED ofwel de rechter. Wanneer niet op een van beide druktoetsen wordt gedrukt zal de schakeling nooit van toestand veranderen.

Laten we eens aannemen dat de linker transistor in geleiding is – de linker LED brandt dus. De spanning tussen collector en emitter van die transistor is dan laag (ongeveer 0,1 V) en dat is niet genoeg om de rechter transistor in geleiding te krijgen (daarvoor is minimaal een spanning van 0,7 V nodig). De rechter transistor spert zodoende, diens CE-spanning is ongeveer gelijk aan de voedingsspanning U_V , en dat betekent weer dat de linker transistor via de rechter weerstand van 100 kΩ gegarandeerd doorgeschakeld blijft. We hebben hier dus met een stabiele toestand te maken.

Zodra we echter op de linker toets drukken, neemt de BE-spanning van de linker transistor af tot nul, zodat deze transistor gaat sperren en diens CE-spanning tot ongeveer U_V toeneemt. De linker LED dooft en via de rechter weerstand van 100 kΩ wordt de rechter transistor in geleiding gebracht. Diens CE-spanning neemt tot vrijwel 0 V af en de rechter LED licht op. Nu kan de linker transistor niet meer in geleiding komen – ook niet als we de drukknop weer loslaten. We hebben dus weer een stabiele toestand, maar nu tegengesteld aan de beginsituatie.

De schakeling blijft net zolang in deze stabiele toestand tot iemand zo vriendelijk is om op de rechter druktoets te drukken en de schakeling weer omklapt naar de andere stabiele toestand.

Nu moeten we eigenlijk alleen nog de vraag beantwoorden welke transistor na het inschakelen van de voedingsspanning als eerste in geleiding komt. Dat valt echter aan de hand van het schema onmogelijk te voorspellen. Wanneer beide helften van de schakeling met exact gelijke onderdelen zouden zijn opgebouwd, dan zou het toeval het laatste woord hebben (anders uitgedrukt: dan hangt het antwoord op de vraag van de onvoorspelbare onderdelenruis af). In de echte wereld bestaan er echter geen exact identieke onderdelen: de weerstanden vertonen fabricagetoleranties, de stroomversterking van beide transistoren is niet gelijk (en die stroomversterking hangt bovendien van de temperatuur af – en dus ook van de vraag welke transistor onlangs nog in geleiding is geweest en daardoor wellicht wat warmer is dan de andere).

Daarom is het beter om de vraag om te keren: hoe kunnen we ervoor zorgen dat bij het inschakelen van de voedingsspanning altijd dezelfde transistor als eerste in geleiding komt? Simpel: verklein gewoon de basisweerstand van de betreffende transistor tot de helft van de in het schema vermelde waarde.

Dat was het voor deze aflevering; volgende keer nemen we nog twee andere multivibratorschakelingen onder de loep, en dan gaan we de transistor als versterker bekijken.

Nog even terug naar de diac

In het november/december-nummer van het vorige jaar [1] hebben we in het voorbijgaan de diac genoemd als (inmiddels min of meer



obsolete) vertegenwoordiger van het genus 'diode'. Daarbij hebben we geschreven:

"Bij uw elektronica-postorderaar (de 'elektronicawinkel om de hoek' bestaat helaas niet meer) kunt u onder de exotische benaming 'diac' twee op deze manier aangesloten zenerdioden in één behuizing kopen."

Twee Elektor-lezers (Pablo Medina en Maurizio Spagni) hebben ons onafhankelijk van elkaar erop gewezen dat dit eigenlijk helemaal niet klopt. En beide lezers hebben helemaal gelijk: een diac is een tamelijk complexe component die een negatieve weerstandskarakteristiek vertoont en die daarom gebruikt kan worden om thyristoren en triacs te triggeren [2].

De vergelijking van een diac met twee in 'antiserie' geschakelde zenerdiodes is dus wel heel erg kort door de bocht. Niettemin zijn we van mening dat de beginnende elektronica-hobbyist zich de werking van de diac beter kan voorstellen als twee zenerdioden in plaats van als een negatieve weerstand. Tegen de tijd dat hij of zij had begrepen wat met 'negatieve weerstandskarakteristiek' wordt bedoeld en hoe een halfgeleidercomponent zo'n gedrag kan vertonen had hij (of zij) zijn/haar soldeerbout al lang gefrustreerd aan de wilgen gehangen.

Maar zoals we al opmerkten: beide briefschrijvers hebben helemaal gelijk, en in de toekomst zullen we zorgvuldiger formuleren! 

220670-03

De artikelreeks "Alle begin..." is gebaseerd op het boek "Basiscursus elektronica" van Michael Ebner, dat bij Elektor is verschenen.

Vragen of opmerkingen?

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



Gerelateerde producten

> **Michael Ebner, Basiscursus elektronica (e-boek, SKU 18232)**
www.elektor.nl/18232

WEB LINKS

- [1] Eric Bogers, Michael Ebner: "Alle begin... zenert er vrolijk op los", ElektorMag 11-12/2022: <https://www.elektormagazine.nl/220384-03>
[2] Diac in Wikipedia: <https://nl.wikipedia.org/wiki/Diac>



LIGT JOUW PASSIE IN DE TECHNIEK?

Gaming Support is op zoek naar een Engineer voor de Afdeling Repair.

Houdt je van het oplossen van storingen of defecten aan onderdelen van onder andere speelautomaten op component niveau, dan zoeken we jou.

Interesse?

Solliciteer dan vandaag nog!

Stuur een e-mail naar: hr@gamingsupport.com of bel 010-5241153.

Meer informatie vind je op onze website.



**GAMING
SUPPORT**

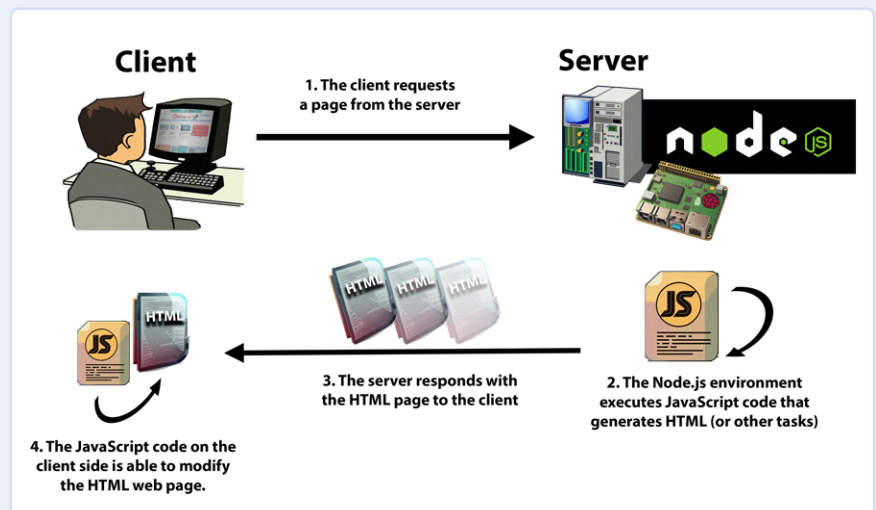
gamingsupport.com

I²C-communicatie met Node.js en een Raspberry Pi

bekijk sensordata in een browser

Franck Bigrat (Frankrijk)

De I²C-bus bestaat al sinds de jaren '80 en wordt goed ondersteund op de huidige populaire platforms, waaronder de Raspberry Pi-serie. Node.js maakt het mogelijk om een webserver in JavaScript te programmeren die gegevens naar een lokaal netwerk of het internet stuurt. Dit artikel laat zien hoe eenvoudig het is om een I²C-sensor aan te sluiten op een Raspberry Pi en de resultaten weer te geven in een webbrowser.



Figuur 1. Node.js server-communicatieflow (bron: sdz.tdct.org).

In dit project gebruiken we een simpele Raspberry Pi Zero en de DS1621 I²C-temperatuursensor. Het meetresultaat wordt weergegeven op een webpagina en is zichtbaar op een tablet of smartphone die op hetzelfde WiFi-netwerk is aangesloten.

Node.js heeft modules zoals *http* en *express*, waarmee de programmeur heel eenvoudig webserver kan bouwen, zodat we de alomtegenwoordige Apache-server niet nodig hebben.

Hoewel er een module bestaat voor de DS1621-sensor, is het interessanter om de *i2c-bus* module van Node.js te gebruiken, waarmee we het project kunnen uitbreiden naar een groot aantal andere I²C-apparaten.

Wat is Node.js?

Node.js in een paar woorden beschrijven is niet zo eenvoudig. Er zijn veel boeken en websites gewijd aan Node.js; hier moet je gewoon begrijpen dat Node.js *JavaScript is dat wordt uitgevoerd door de webserver, dus niet door een webbrowser die op een client draait* (maar de browser kan natuurlijk ook JavaScript-code uitvoeren als dat nodig is). Het is gebouwd op de JavaScript-engine van Google Chrome (V8). Eenvoudig gezegd schrijft de programmeur JavaScript-toepas-

singen, maar deze draaien op de server. In **figuur 1** is een en ander aanschouwelijk gemaakt.

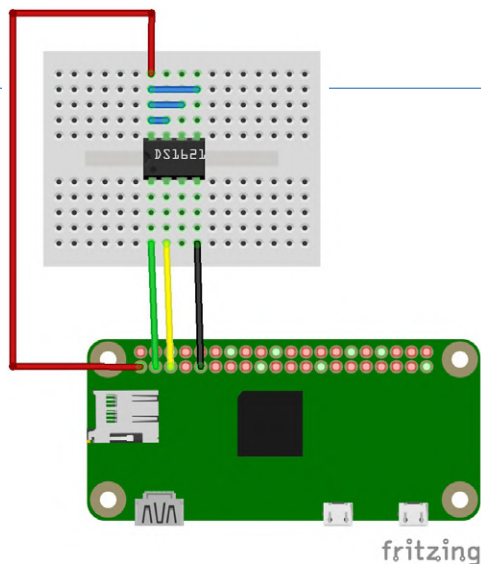
Om (bijna) alles over Node.js te leren, raad ik de lezer sterk aan de websites op [1] en [2] te bezoeken.

De lezer kan ook deze boeken raadplegen:

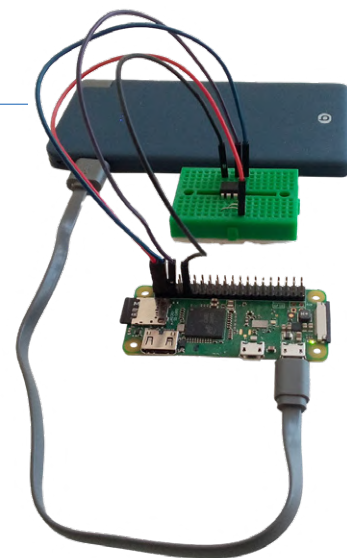
- *Node.js in Practice* door Alex Young en Marc Harter (Manning);
- *Beginning Node.js* door Basarat Ali Sayed (Apress);
- *The Node Beginner Book* door Manuel Kiessling;
- *Instant Node.js Starter* door Pedro Teixeira (Packt Publishing).

I²C en SMBus

In deze paragraaf ga ik ervan uit dat de lezer enigszins bekend is met het I²C-protocol (Inter-Integrated Circuit). Het belangrijkste is dat I²C werd ontwikkeld door Philips in de jaren '80 om het aantal verbindingen tussen een microprocessor en verschillende andere geïntegreerde schakelingen te minimaliseren. Het is een serieel, tweedraads (data en klok) synchroon datatransmissie-protocol. Verschillende schakelingen



Figuur 2. Zo wordt de Raspberry Pi Zero W op de DS1621 aangesloten.



Figuur 3. De praktische uitvoering, inclusief batterij.

kunnen tegelijkertijd op dezelfde data- en kloklijnen worden aangesloten.

Details over de functionaliteit van de i2c-bus Node.js-module zijn te vinden in [3].

Als je zorgvuldig leest, zie je dat sommige functies een smbus-prefix hebben of deel uitmaken van de SMBus-sectie. SMBus maakt communicatie mogelijk tussen de processor van een computer en verschillende periferie, zoals apparaten voor energiebeheer, temperatuursensoren of ventilatoren.

De Raspberry Pi ondersteunt dit protocol in hardware: de System Management Bus is afgeleid van de I²C-bus, en de twee protocollen zijn redelijk compatibel. Maar er zijn enkele verschillen tussen SMBus en I²C; deze worden uitgelegd in het Texas Instruments-document op [4].

De componenten aansluiten en configureren

Nu wordt het tijd om de Raspberry Pi aan te sluiten op de DS1621 temperatuursensor volgens de layout van **figuur 2**.

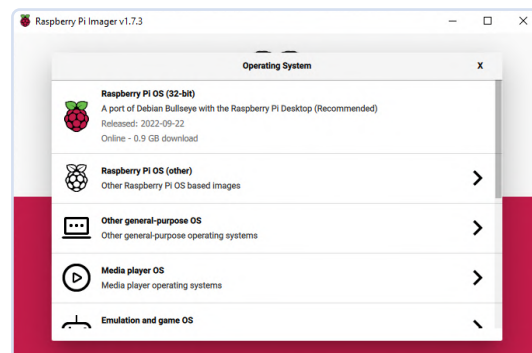
Vergeet niet de pinnen 5, 6 en 7 (A0, A1, A2) van de DS1621 aan VCC te leggen om het I²C-adres van de component in te stellen op 4F (hexadecimaal). De praktische uitvoering is te zien in **figuur 3**.

De Raspberry Pi configureren

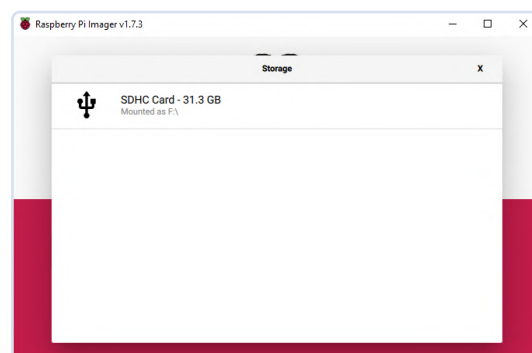
1. Download en installeer Raspberry Pi Imager van: <https://raspberrypi.com/software>
2. Start Raspberry Pi Imager (**figuur 4a**), klik op **CHOOSE OS** en vervolgens op **Raspberry Pi OS (Other)** in de lijst die dan verschijnt (**figuur 4b**), en dan **Raspberry Pi OS Lite (32-bit)** in het submenu.
3. Klik op **CHOOSE STORAGE** en selecteer vervolgens je microSD-kaart in het **Storage**-dialoogvenster (**figuur 4c**).
4. Klik op het **Settings**-pictogram (het kleine tandwiel) rechts van de toepassing en stel de initiële configuratie-opties van de Raspberry Pi in:
 - vink **Set hostname**: aan en voer de gewenste hostnaam in; **raspberry** voldoet prima;
 - vink **Enable SSH** aan en selecteer het keuzerondje **Use password authentication**;



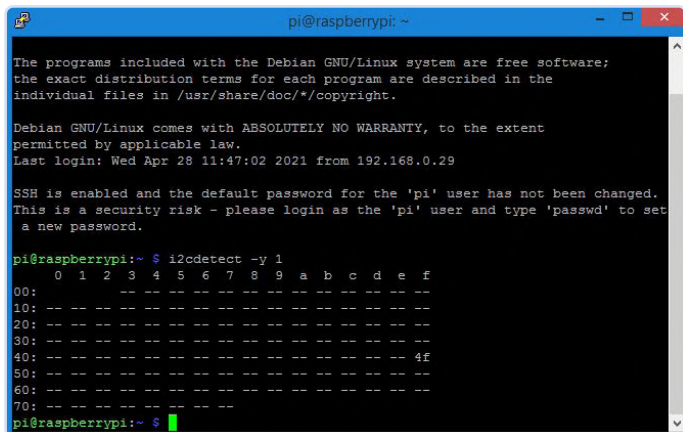
Figuur 4a. Raspberry Pi Imager.



Figuur 4b. Keuze van het besturingssysteem.



Figuur 4c. Keuze van het opslagmedium.



```
pi@raspberrypi:~$ i2cdetect -y 1
00:  0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
10:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
20:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
30:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
40:  -- -- -- -- -- -- -- -- -- -- -- -- 4f --
50:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
60:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
70:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
```

Figuur 5. De uitvoer van het i2cdetect-programma.

- › vink *Set username and password* aan en vul de *Username* (pi is prima) en een *Password* in (kies er een die je makkelijk kunt onthouden, bijvoorbeeld *raspberrypi*);
 - › vink *Configure wireless LAN* aan en voer de SSID en het wachtwoord van je WiFi-netwerk in;
 - › vink *Set local settings* aan en pas je *Time zone* en *Keyboard layout* indien nodig aan, of laat de standaardinstellingen zoals ze zijn;
 - › klik op de knop *SAVE*.
5. Installeer het OS door op *WRITE* te klikken (bevestig met *YES*).
6. Als het OS geschreven is is, kun je de microSD-kaart verwijderen.
7. Steek de kaart in de microSD-slot van de Raspberry Pi en schakel die in.
8. Zoek het IP-adres van de Raspberry Pi op het netwerk met een tool zoals Advanced IP Scanner, of (nog beter) haal het uit de DHCP lease-instellingen in je WiFi-router.
9. Gebruik een SSH-client zoals Putty om verbinding te maken met de Raspberry Pi via zijn IP-adres en gebruikersnaam/wachtwoord (in ons voorbeeld respectievelijk *pi* en *raspberrypi*).
10. Om de volledige opslagruimte van de microSD-kaart te benutten, moet je het bestandssysteem uitbreiden:
- › open het *raspi-config* tool: `sudo raspi-config`;
 - › kies *Advanced Options* en vervolgens de optie *Expand Filesystem*;
 - › bevestig met de Enter-toets.
 - › ga naar *<Finish>* en bevestig met Enter, bevestig dan de herstart door naar *<Yes>* te gaan en op Enter te drukken.
11. Om de I2C-bus te activeren:
- › open het *raspi-config* tool: `sudo raspi-config`;
 - › selecteer *Interfacing Options* en dan *I2C (Enable/disable automatic loading of I2C kernel module)*;
 - › bevestig met de Enter-toets;
 - › bevestig nogmaals door naar de knop *<Yes>* te gaan en nogmaals op Enter te drukken;
 - › sluit af met *<OK>* en dan *<Finish>*.

12. Sluit de Raspberry Pi aan op de DS1621 volgens het schema van figuur 2.
13. Controleer de I2C-communicatie:
- › voer eerst een globale update uit: `sudo apt update`;
 - › installeer de I2C-diagnosetools: `sudo apt install i2c-tools` (misschien moet je de Raspberry Pi eerst opnieuw opstarten);
 - › voer deze opdracht uit: `i2cdetect -y 1`
 - › in de lijst die verschijnt, moet de sensor staan met het adres (4f) dat wij 'hard' hebben ingesteld (figuur 5).
14. Installeer nu Node.js en NPM (Node Package Manager): `sudo apt install -y nodejs npm` (dit kan even duren).
15. Controleer de versies met `node -v` gevolgd door `npm -v`.
16. Installeer de *express*, *i2c-bus* en *sleep* modules:
- ```
npm install express
npm install i2c-bus
npm install sleep
```
- (kan waarschuwingen genereren)
17. Maak een map met de naam *Documents* en een submap met de naam *NodeJS* aan.
18. Kopieer met een FTP-client zoals FileZilla de bestanden *DS1621\_V3.js* en *DS1621\_V4.html* naar de map *NodeJS*.

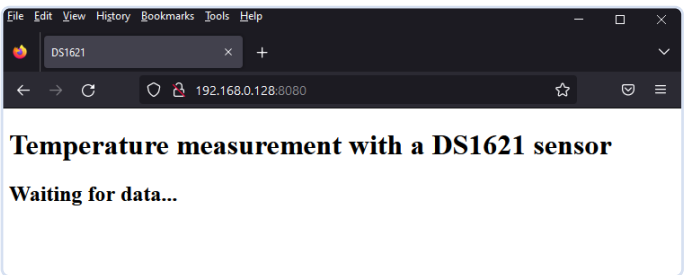
Proberen we het uit!

Zodra alle elementen zijn aangesloten, de Raspberry Pi is geconfigureerd, Node.js en de verschillende modules zijn geïnstalleerd, en de *.js* en *.html* bestanden naar de Raspberry Pi zijn gekopieerd, open je de map *Documents/NodeJS* en voer je het script uit met `node DS1621_V3.js`. Sluit gewoon een PC, een tablet of een smartphone aan op je WiFi-netwerk, open je favoriete browser en voer dit adres in:

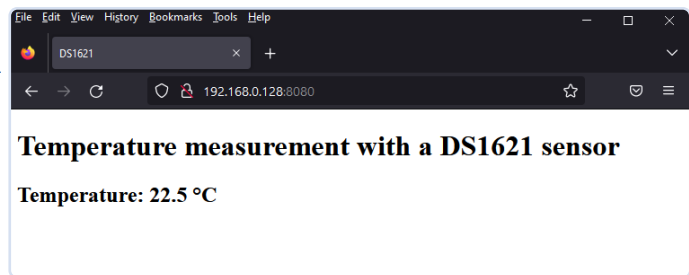
[je\_Raspberry\_Pi\_IP\_adres]:8080

Als de vorige stappen correct zijn uitgevoerd, moet je nu iets zien als figuur 6.

Na enkele seconden verschijnt de temperatuur (figuur 7). Vanaf dan wordt de temperatuur elke vijf seconden ververs.



Figuur 6. Wachten op gegevens.



Figuur 7. De gemeten temperatuur wordt weergegeven.

## De Node.js- en HTML-scripts nader bekijken

Belangrijk maar lastig bij Node.js is het begrijpen van de uitvoering van het script. In Node.js is de uitvoering van een script niet sequentieel. Zie **figuur 8**: de instructies in de functie `Server.get('/',...)` worden niet uitgevoerd vóór de instructies in de functie `Server.get('/temp',...)`, die niet worden uitgevoerd vóór de instructies onderaan. De instructies in de functie `Server.get('/',...)` worden alleen uitgevoerd als de server de `[Rpi_IP_Address]:8080`-request van de client ontvangt. De instructies in de functie `Server.get('/temp',...)` worden alleen uitgevoerd als de server de request `[Rpi_IP_Address]:8080/temp` van de client ontvangt. Dit wordt *event-driven programming* genoemd.

In de eerste vijf regels van het Node.js-script (links in figuur 8) importeren we de vereiste Node.js-modules en maken we verschillende objecten aan:

➤ We maken een `I2C`-object met behulp van de `i2c-bus` module:

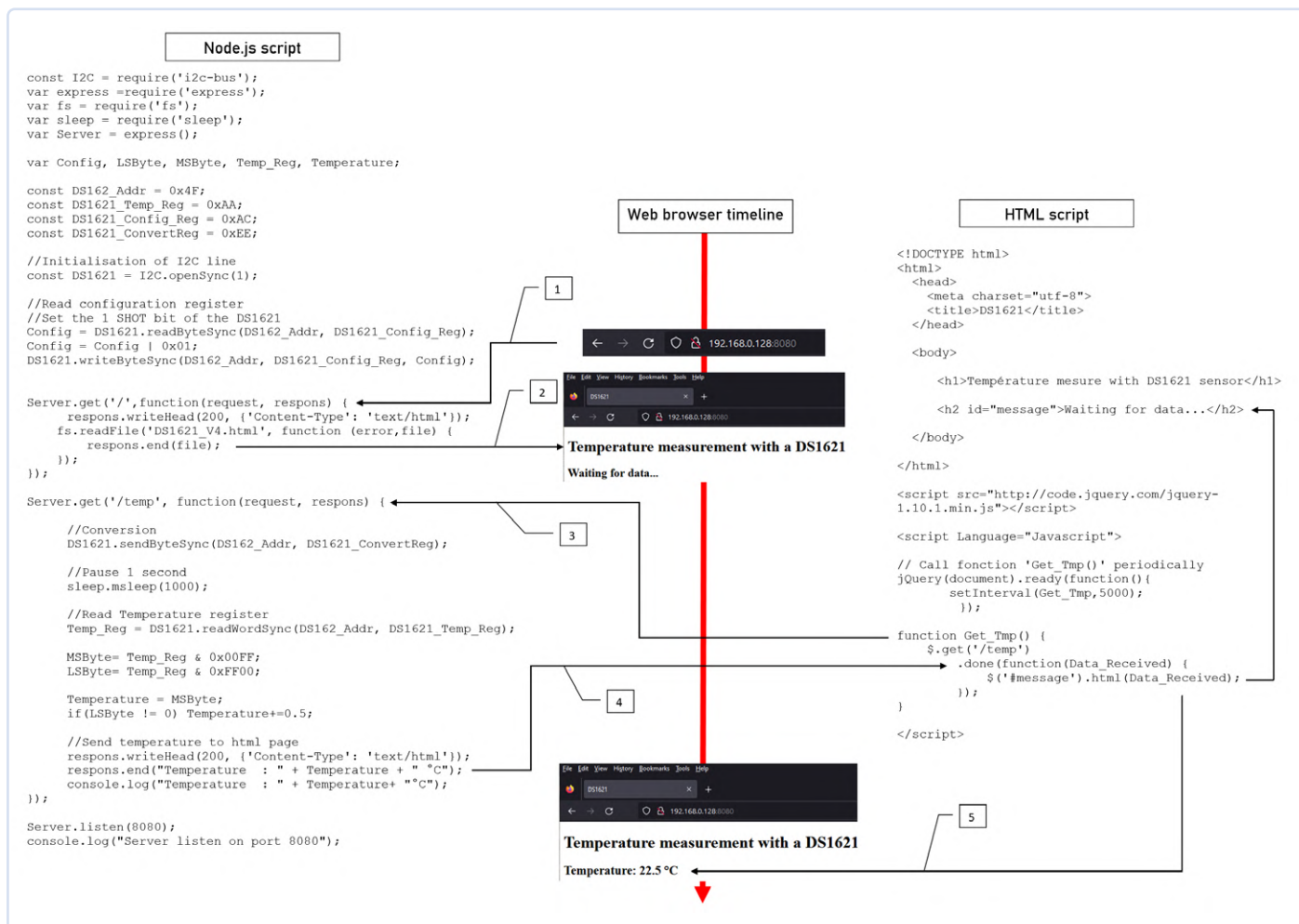
```
const I2C = require('i2c-bus');
```

➤ We maken een `express`-object aan met behulp van de `express`-module. Deze module is een framework dat is ontworpen voor het bouwen van webapplicaties met Node.js:

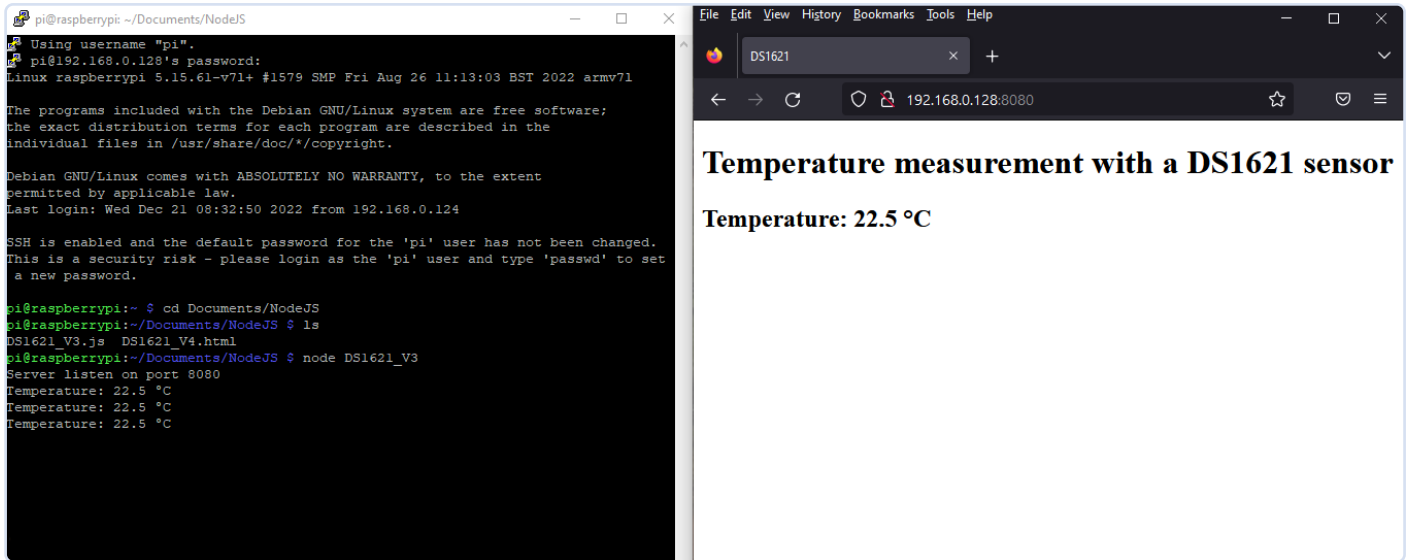
```
var express = require('express');
```

➤ We maken een `fs`-object aan met de `fs`-module (file system). Hiermee kunnen we bestanden op de microSD-kaart lezen en schrijven:

```
var fs = require('fs');
```



Figuur 8. Tijdslijn van de gebeurtenissen.



Figuur 9. De Linux-terminal en de webbrowser.

- We maken een `sleep`-object met behulp van de `sleep`-module. Met deze module kunnen we vertragingen creëren:

```
var sleep = require('sleep');
```

- Tenslotte maken we een `Server`-object aan:

```
var Server = express();
```

In de volgende vijf regels definiëren we vijf variabelen en vier constanten. Deze constanten bevatten de respectieve adressen van de DS1621-registers.

```
var Config, LSByte, MSByte, Temp_Reg, Temperatuur;
```

```
const DS162_Addr = 0x4F;
const DS1621_Temp_Reg = 0xAA;
const DS1621_Config_Reg = 0xAC;
const DS1621_ConvertReg = 0xEE;
```

Vervolgens openen we de I<sup>2</sup>C-bus en maken we een object met de naam `DS1621`. Dit object zal worden gebruikt om met de sensor te communiceren via I<sup>2</sup>C.

```
const DS1621 = I2C.openSync(1);
```

De volgende drie regels stellen het `1SHOT`-bit van de DS1621 in. De temperatuur wordt dus één keer (en niet meer dan één keer) gemeten en geconverteerd als de I<sup>2</sup>C Master (de Raspberry Pi) een conversie nodig heeft.

```
Config = DS1621.readByteSync(DS162_Addr,
DS1621_Config_Reg);
Config = Config | 0x01;
DS1621.writeByteSync(DS162_Addr,
DS1621_Config_Reg, Config);
```

Laten we nu eens kijken wat er gebeurt als de browser een request stuurt naar de server (figuur 9). Wanneer het Node.js-script wordt uitgevoerd, verschijnt na enkele seconden het bericht "Server listen on port 8080" in de Linux-terminal.

Open een webbrowser op je PC of tablet die verbonden is met het WiFi-netwerk. Typ in de adresbalk `http://` en het IP-adres van de Raspberry Pi en de poort waarop de server luistert; bijvoorbeeld `http://192.168.0.128:8080`. Poort 8080 is gekozen omdat die anders is dan de standaard gebruikt door de Apache webserver (poort 80).

1. Wanneer je het IP-adres opgeeft, stuurt de browser een `get` HTTP-request naar de server. Wat gebeurt er?
  - a. De server detecteert het verzoek met `Server.get('/', ...)` en voert de anonieme callback-functie `function(request, respons)` uit.
  - b. Het `request`-object bevat het verzoek van de browser.
  - c. In het `respons`-object sturen we twee headers mee als parameters:
    - i. Een `200` http-code, wat betekent dat het verzoek succesvol was.
    - ii. De string `"Content-Type: 'text/html'"`, hetgeen betekent dat de browser voorbereid moet zijn om tekstgegevens van het type HTML te ontvangen.
  - d. De functie-aanroep `respons.writeHead()` stuurt de parametergegevens terug naar de browser.
2. De functie `fs.readFile()` leest het bestand `DS1621_V4.html`. De anonieme callback-functie `function(error, file)` wordt uitgevoerd en, in het geval dat deze succesvol is (daar gaan we van uit), wordt het bestand gelezen en in het `file`-object geplaatst. De functie-aanroep `respons.end(file)` stuurt de HTML naar de browser, die de webpagina weergeeft. In geval van een fout wordt een foutcode geplaatst in het `error`-object.

De browser heeft nu de webpagina in HTML-formaat. Wat gebeurt er nu? Laten we eens kijken naar het HTML-script (rechts in figuur 8). Het eerste wat we zien is de regel `<h2 id="message">Waiting for data...</h2>`. Dit is een plaatshouder voor de kop waar de temperatuur zal worden geschreven. We zullen later zien hoe. Het tweede wat we zien is het blok dat volgt. Het is een script geschreven in JavaScript, maar het is client-side, wat betekent dat het wordt uitgevoerd door de browser:

```
<script src="http://code.jquery.com/jquery-1.10.1.min.js"></script>
```



```
<script Language="Javascript">
// Call function Get_Tmp() periodically:
jQuery(document).ready(function() {
 setInterval(Get_Tmp, 5000);
});

function Get_Tmp() {
 $.get('/temp')
 .done(function(Data_Received) {
 $('#message').html(Data_Received);
 });
}
</script>
```

Eerst importeren we vanaf het web de jQuery-bibliotheek, die ons enkele interessante functies biedt. Dankzij jQuery kunnen we een timer instellen die elke 5 seconden een functie genaamd `Get_Tmp()` aanroept:

```
jQuery(document).ready(function(){
 setInterval(Get_Tmp, 5000);
});
```

Tot slot hebben we de functie `Get_Tmp()`:

```
function Get_Tmp() {
$.get('/temp')
 .done(function(Data_Received) {
 $('#message').html(Data_Received);
 });
}
```

En dat is het interessante deel van het script!

Laten we eens kijken hoe het samenwerkt met het Node.js-script aan de serverkant (zie tijdlijn in het midden van figuur 8):

1. Wanneer de functie `Get_Tmp()` wordt aangeroepen, stuurt hij een *get*-verzoek naar de server. Wat gebeurt er dan?
  - a. De server detecteert het verzoek met `Server.get('/temp')` en voert de anonieme callback-functie `function(request, respons)` uit.
  - b. De I<sup>2</sup>C-master (Raspberry Pi) communiceert met de DS1621 via de I<sup>2</sup>C-bus, vraagt om een dataconversie, leest de data en zet de data om in °C.
  - c. Het `request`-object bevat het verzoek van de browser.
  - d. In het `respons`-object slaan we weer onze twee parameters op:
    - i. De waarde `200`, een code die betekent dat het verzoek succesvol is.

- ii. De string `"Content-Type: 'text/html'"` betekent dat de browser tekst ontvangt die als HTML moet worden geïnterpreteerd.
- e. De functie `respons.writeHead()` retourneert deze gegevens naar de browser.

2. De functie-aanroep `respons.end("Temperature: " + Temperature + " °C")` stuurt het bericht *Temperatuur: XX °C* (een ASCII-string) naar de browser. XX is de door de DS1621 gemeten temperatuur.
3. Deze string wordt in het object `Data_Received` geplaatst.
4. Herinner je je de header-sectie `<h2 id="message">Waiting for data...</h2>`? We zien dat deze header-tag de `id` van `message` heeft, dus wanneer de string wordt ontvangen, wordt dankzij de regel `$('#message').html(Data_Received);` de door de server verzonden string in de header-sectie geplaatst en zal de webpagina de temperatuur tonen.

Ten slotte nog een paar woorden over de twee laatste regels van het Node.js-script:

```
Server.listen(8080);
console.log("Server listen on port 8080");
```

De regel `Server.listen(8080);` start de server, en de regel `console.log("Server listen on port 8080");` schrijft het bericht *Server listen on port 8080* naar de Linux-console (shell). Bestudeer figuur 8 zorgvuldig; ik denk dat alles beter te begrijpen is aan de hand van een tijdlijn. ◀

210367-03

### Vragen of opmerkingen?

Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de redactie van Elektor via [redactie@elektor.com](mailto:redactie@elektor.com).



### Gerelateerde producten

> **Vincent Himpe, Mastering the I<sup>2</sup>C Bus** (SKU 15385)  
[www.elektor.nl/15385](http://www.elektor.nl/15385)

> **Raspberry Pi Zero W Starter Kit** (SKU 18328)  
[www.elektor.nl/18328](http://www.elektor.nl/18328)

## WEBLINKS

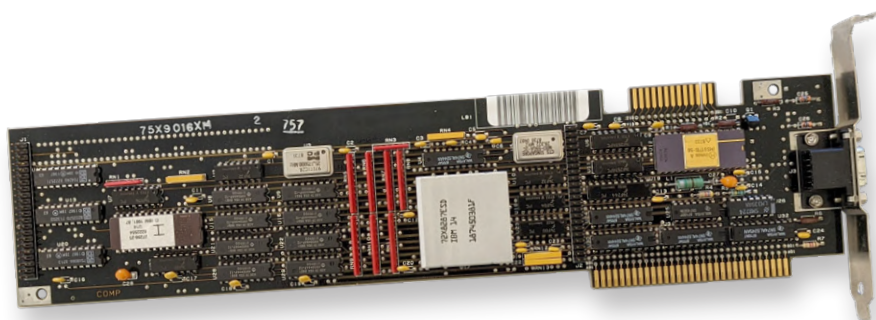
- [1] Introduction to Node.js: A Beginner's Guide to Node.js and NPM: <https://elektor.link/udaynode>
- [2] What is Node.js: A Comprehensive Guide: <https://simplilearn.com/tutorials/nodejs-tutorial/what-is-nodejs>
- [3] 'i2c-bus' module: <https://npmjs.com/package/i2c-bus>
- [4] SMBus Compatibility With an I<sup>2</sup>C Device: <https://ti.com/lit/an/sloa132/sloa132.pdf>

# Video-output met microcontrollers (2)

VGA en DVI

Mathias Claussen (Elektor Labs)

Microcontrollers kunnen via een VGA- of DVI-interface pixels naar een monitor sturen. Met de ESP32 of de RP2040 van de Raspberry Pi Pico zijn aanzienlijk meer dan 256 kleuren mogelijk. Dankzij sprites, tegels en een paar trucs kunnen de microcontrollers van vandaag graphics genereren die minstens zo geavanceerd zijn als die van de klassieke spelconsoles uit de jaren '90.



Figuur 1. IBM VGA-kaart (bron: Wikimedia / Vlask, CCASA 4.0 <https://elektor.link/cc4dot0>).

Het eerste deel van dit artikel behandelde de uitvoer van composiet-videosignalen in PAL (*Phase Alternating Line*) of NTSC (*National Television Systems Committee*) formaat. Terwijl de uitvoer van monochrome beelden zelfs mogelijk is met een Arduino UNO, stellen beelden in kleur hogere eisen. Met een paar trucs kun je echter een ESP32 of zelfs een ATmega overtuigen om kleuren-output te produceren. In principe is de beeldkwaliteit met composiet video echter matig. Vooral de scherpte ontbreekt, en tekst is dus niet goed leesbaar. De eerste thuiscompu-

ters, zoals de Sinclair ZX81 en de Commodore 64, genereerden composiet video. Voor professionele toepassingen was echter een zo scherp mogelijke tekstweergave nodig, en in sommige gevallen zelfs afbeeldingen in kleur. Daarom bood IBM in 1987 iets nieuws en beters voor zijn PC's met zijn VGA-interface (*Video Graphics Array*). Ook microcontrollers kunnen deze analoge VGA-signalen genereren om een scherp en kleurrijk beeld weer te geven. Sommige microcontrollers kunnen zelfs een beeld volledig digitaal weergeven via DVI (*Digital Visual Interface*). Het tweede deel

van deze serie gaat dus over VGA en DVI en de trucs die gebruikt worden voor grafische uitvoer en effecten.

## VGA: video voor de IBM-C

VGA was de grafische kaart (**figuur 1**) voor PS/2 PC's die IBM in 1987 introduceerde. Deze grafische kaart maakte de toen nieuwe, 3-rijige D-sub DE-15 connector (**figuur 2**) populair, hoewel VGA niet alleen de fysieke connector specificeert, maar ook de gebruikte signalen en hun timing.

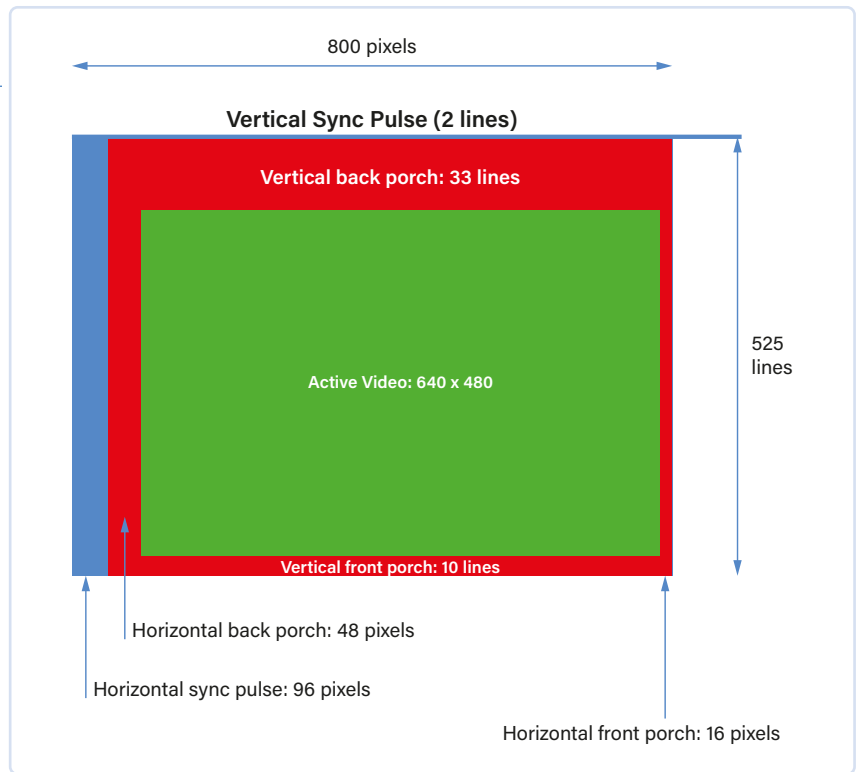
De kleursignalen in VGA zijn analoog en worden verzonden via drie signaallijnen (rood, groen en blauw). De horizontale en verticale synchronisatie gebeuren digitaal via afzonderlijke TTL-signalen. De eerste VGA-kaart van IBM kon 256 kleuren uitvoeren uit een RGB-palet van  $3 \times 6$  bits (= 262.144 kleuren) bij een resolutie van  $320 \times 200$  pixels en slaagde er in om  $640 \times 480$  pixels met 16 kleuren te genereren. De eerste IBM-monitoren die hiervoor geschikt waren, konden slechts  $640 \times 480$  of  $640 \times 400$  pixels aan bij een herhalingsfrequentie van 60 of 70 Hz. De VGA-kaarten van IBM werden al snel gekloond door veel andere fabrikanten, waardoor VGA een de facto standaard werd voor video-uitvoer op wat IBM-compatibele PC's werden genoemd.



Figuur 2. VGA-connector op een PC (bron: dr. Thomas Scherer).



Figuur 3. Geopende TV met beeldbuis (bron: Shutterstock / Serhiy Stakhnyk).



Figuur 4. Timing van een VGA-sigitaal met 640x480 zichtbare pixels (bron: <https://elektor.link/VGAtiming>).

## Zichtbare en onzichtbare pixels

Een videosignaal met 640x480 zichtbare pixels bij een herhalingsfrequentie van 60 Hz werkt met een pixelfrequentie van 25,175 MHz. Als we de pixels van een beeld echter vermenigvuldigen met de beeldsnelheid, levert dat slechts 18,432 MHz op. Waar komt dit verschil vandaan?

Net als composiet PAL- en NTSC-signalen heeft ook VGA niet-zichtbare signaalgebieden, omdat ook deze norm is ontworpen voor monitoren die op beeldbuizen zijn gebaseerd (figuur 3). In principe gelden hier dezelfde timingvereisten als voor de aansturing van de beeldbuis van een televisietoestel.

**Figuur 4** toont de timing voor de 640x480 pixel-modus met een pixelklok van 25,175 MHz. Een beeld begint met een verticale synchronisatiepuls van 63,555142  $\mu$ s (= twee beeldlijnen van elk 800 pixels). De drie elektronenbundels worden gedurende

deze tijd gedoofd, terwijl zij van het einde van het laatste volledige beeld rechtsonder naar het begin linksboven van het nieuwe beeld bewegen. De lijnen beginnen met een horizontale synchronisatiepuls van 69 pixels ( $\approx 3,81 \mu$ s). In de volgende 33 lijnen (de lijnen 3...35) wordt de horizontale synchronisatiepuls gevolgd door de zogenaamde 'vertical back porch' (Nederlands: 'achterstoep') van in dit geval 704 pixels ( $\approx 27,96 \mu$ s). Deze is bedoeld om de elektronenbundels of de magnetische afbuiging voldoende tijd te geven om terug te keren naar de uitgangspositie voor het nieuwe beeld. Daarna volgen 480 lijnen met de eigenlijke beeldinformatie, te beginnen met de horizontale sync-puls van 96 pixels en vervolgens de 'horizontal back porch' ('achterstoep') van 48 pixels ( $\approx 1,9 \mu$ s), plus de 640 beeldpixels van één lijn ( $\approx 25,42 \mu$ s) en de afsluitende voorstoep van 16 pixels ( $\approx 0,64 \mu$ s).

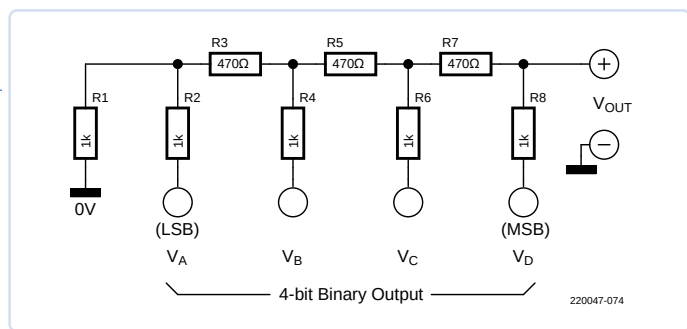
**Figuur 5** toont de structuur van een dergelijke

beeldlijn. Tenslotte zijn er 10 lijnen, elk met een horizontale sync-puls gevolgd door de 'verticale voorstoep' (ook weer elk bestaande uit 704 pixels). Het uitgezonden beeld van 640x480 zichtbare pixels bestaat dus eigenlijk uit 800x525 pixels wat de timing betreft. De elektronenbundels zijn alleen actief tijdens het eigenlijke beeld en blijven anders gedoofd. Hoewel dit complex klinkt, is het eigenlijk een van de eenvoudigste beeldsignalen die door een microcontroller kunnen worden gegenereerd. En om verwarring te voorkomen: op het internet kun je iets andere timing-specificaties vinden voor de VGA-framerate. En als je de pixelklok van 25,175 MHz deelt door de 800x525 pixels, is het resultaat niet precies 60 Hz voor de framerate, maar  $\approx 59,94$  Hz. De analoge elektronica van de betreffende monitoren stelt in dit opzicht echter niet al te hoge eisen en zal zelfs bij een grotere afwijking van 60 Hz nog goed synchroniseren.

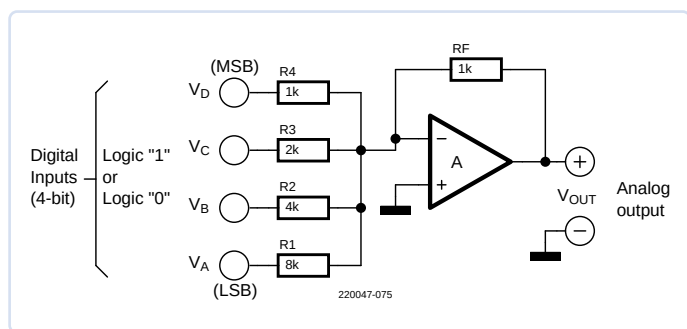


Figuur 5. Timing van een VGA-beeldlijn.

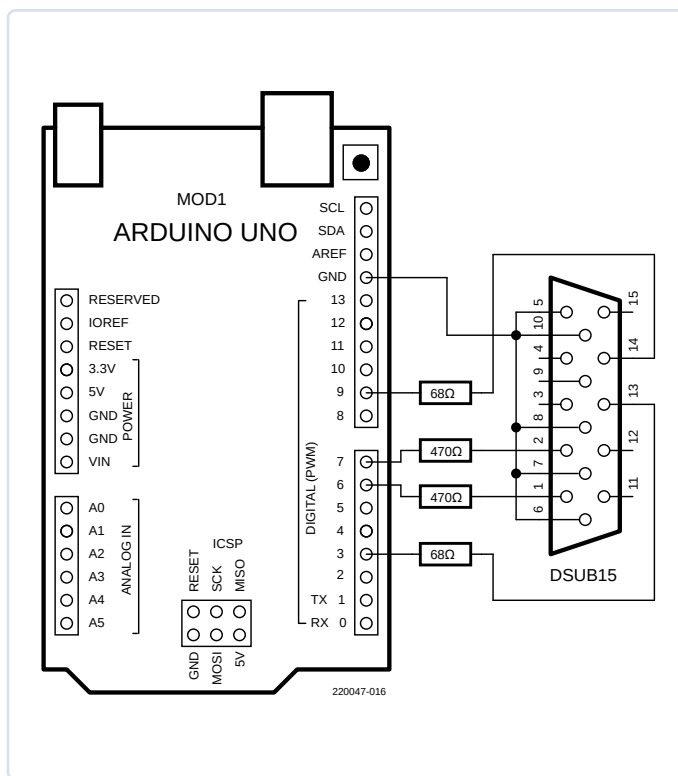




Figuur 6. DAC met R2R-laddernetwerk.



Figuur 7. Binair-gewogen DAC met opamp (bron: <https://elektor.link/BinaryWeightedDAC>).



Figuur 8. Bedrading voor een VGA-uitgang bij een Arduino UNO.

## VGA-signaalniveaus

Volgens de VGA-specificatie is het spanningsbereik van de analoge RGB-beeldsignalen 0...0,7 V, waarbij 0,7 V overeenkomt met de maximale helderheid. De horizontale en verticale synchronisatiesignalen hebben TTL-niveaus (0 V en 5 V).

Microcontrollers met voedingsspanningen van 2,5...5 V kunnen de signalen voor verticale en horizontale synchronisatie rechtstreeks via hun I/O-pinnen uitvoeren, aangezien 2,5 V betrouwbaar wordt geïnterpreteerd als een 'hoog' TTL-niveau. Om de wisselende helderheidsinformatie voor elke kleur uit te voeren, zijn echter spanningsdelers nodig.

Aangezien de interne DAC's van de meeste microcontrollers niet snel genoeg zijn, worden de GPIO-pennen gewoonlijk gebruikt als eenvoudige DAC's met externe weerstanden voor het genereren van de drie analoge signalen. Bij het berekenen van de vereiste weerstandswaarden moet er rekening mee worden gehouden dat de analoge RGB-ingangen van de monitor zijn afgesloten op 75  $\Omega$ . De DAC's worden gewoonlijk geïmplementeerd met behulp van R2R-netwerken (figuur 6) of binair gewogen weerstanden (figuur 7).

## VGA en de ATmega

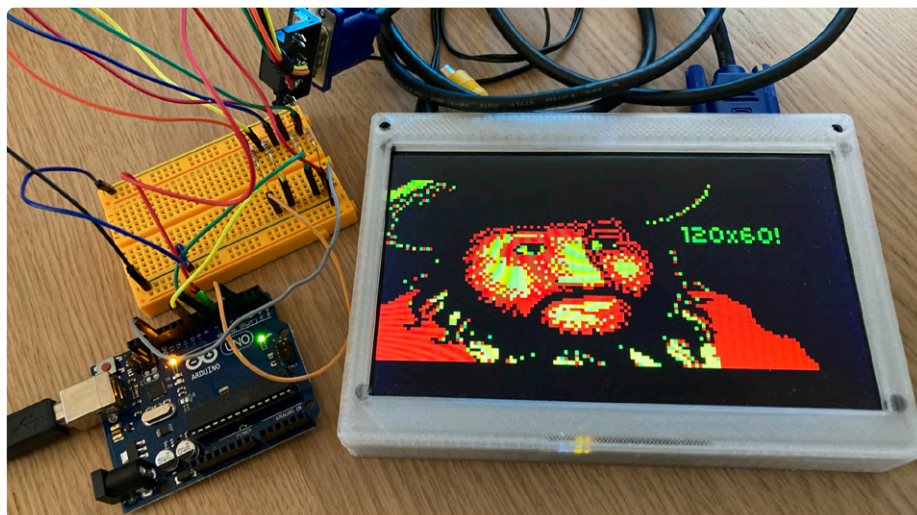
VGA signalen kunnen zelfs met ATmega-microcontrollers worden gegenereerd, zodat een Arduino UNO kan worden uitgerust met VGA graphics. Met vier weerstanden en een

15-polige D-sub connector kunnen we een Arduino UNO 120x60 pixels in vier kleuren laten weergeven. De bijbehorende Arduino-bibliotheek is *VGAX* [1] en kan worden gedownload van de GitHub-pagina van het project. Als een Arduino UNO wordt bedraad zoals in **figuur 8**, is het zelfs mogelijk om graphics in vier kleuren uit te voeren, zoals in **figuur 9**. De beperkende factor hierbij is de hoeveelheid geheugen die een Arduino UNO kan hebben. Aangezien de kleursignalen door de GPIO-pinnen worden gegenereerd, moeten

we die snel genoeg van nieuwe waarden kunnen voorzien.

De timing voor de signalen wordt in dit geval opgelost met interrupts. Een neveneffect hiervan is dat interrupts van andere periferie mogelijk kunnen botsen met die voor de videosignalen. Timer1 en Timer2 genereren hier de horizontale en verticale synchronisatiepuls. Timer0 wordt gebruikt om de jitter in de interrupt-calls te omzeilen.

De *VGAXUA*-bibliotheek [2] voor de Arduino UNO en Arduino Mega kan monochrome



Figuur 9. VGA-uitgang op een scherm dat is aangesloten op de Arduino UNO.



beelden van  $192 \times 080$  tot  $200 \times 240$  pixels uitvoeren. Om rekentijd te besparen wordt hier de USART gebruikt om het beeld te genereren. Elke byte in het (beeld)geheugen komt dan overeen met 8 pixels. Oudere oplossingen voor VGA-uitvoer met AVR-chips overklokten de microcontroller tot 32 MHz [3] of gebruikten extern geheugen voor de video-uitvoer [4].

## VGA en de ESP32

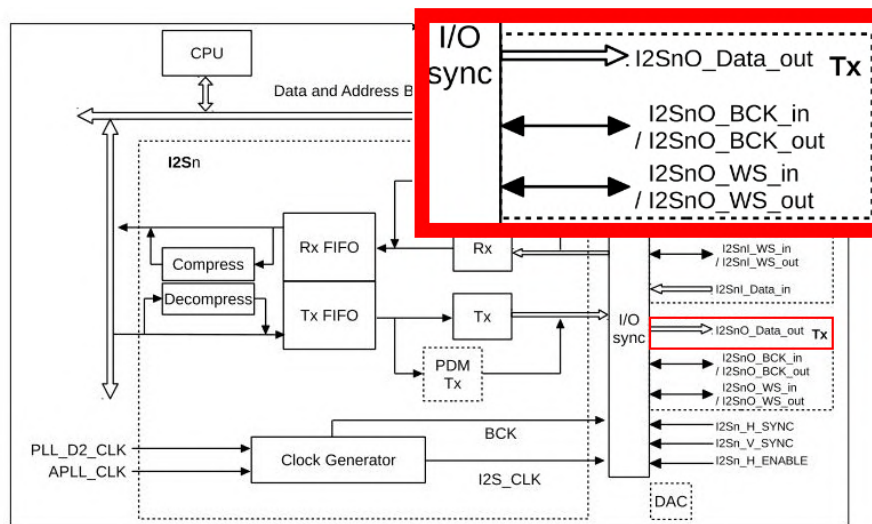
Een ESP32 kan ook VGA 'doen'. Ideale omstandigheden worden geboden door zijn klok van 240 MHz en de flexibele I/O-matrix, waarmee digitale signalen tot 80 MHz kunnen worden uitgevoerd. Het interne geheugen van de ESP32 is met 520 KB voldoende voor een uitvoer van  $640 \times 480$  pixels met 256 kleuren (= 307.200 bytes), waarbij dan nog voldoende RAM overblijft voor andere software.

De timing-specificatie voor een VGA-sig-naal bij  $640 \times 480$  pixels en 60 Hz is te vinden in [5]. Het project "ESP32 VGA" [6] toont een interessante manier om een VGA-sig-naal te genereren met behulp van de interne I<sup>2</sup>S-controller (Inter-IC Sound) [6]. De I<sup>2</sup>S-controller van de ESP32 kan meer dan alleen audiogegevens uitvoeren [7]: hij biedt extra bedrijfsmodi als interface voor een camera of LCD (figuur 10 toont het bijbehorende deel van het blokschema).

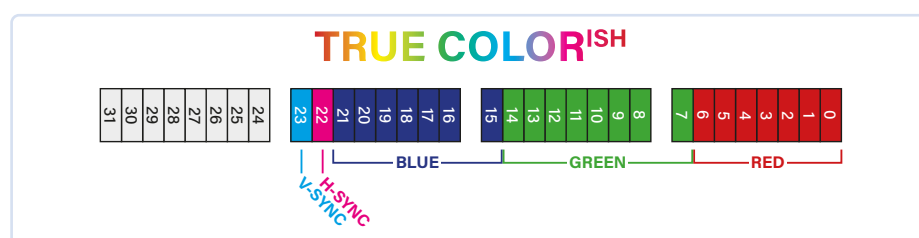
De LCD-interface van de I<sup>2</sup>S-controller heeft een busbreedte tot 24 bit. Dit zou zelfs voldoende zijn voor TrueColor (16.777.216 kleuren) indien die 24 bits volledig gebruikt konden worden voor de uitvoer van kleur. Een VGA-sig-naal vereist echter horizontale en verticale synchronisatie naast de RGB-signalen, zodat slechts 22 bits beschikbaar zijn voor kleur (figuur 11).

Er zijn nog twee hordes te nemen: een beeld van  $640 \times 480$  pixels neemt 921.600 bytes RAM in beslag bij 24bit-kleur – dat is meer dan een ESP32 kan bieden. Bovendien moet een pixelklok van 25,175 MHz worden gehaald. Voor de uitvoer van 24 bit moet de I<sup>2</sup>S-controller 32 bit per pixel uit het geheugen lezen. Het blijkt dat de I<sup>2</sup>S-controller in de ESP32 de 32bit-woorden met maximaal 10 MHz kan uitvoeren, wat niet genoeg is. De seriële uitvoer van de pixels is daarom beperkt tot 10 MHz. De maximale resolutie daarentegen wordt beperkt door het interne geheugen.

Zie [8] voor een lijst van de pixel-klokkfrequenties versus VGA-resolutie. In de modus met  $800 \times 600$  pixels bij 60 Hz is de pixelklok van



Figuur 10. De I<sup>2</sup>S-controller van de ESP32 (bron: Espressif – <https://elektor.link/ESP32TechMan>).



Figuur 11. 22-bit video-uitvoer met H- en V-sync op ESP32 (bron: YouTube – <https://elektor.link/YTESP32VGA>).

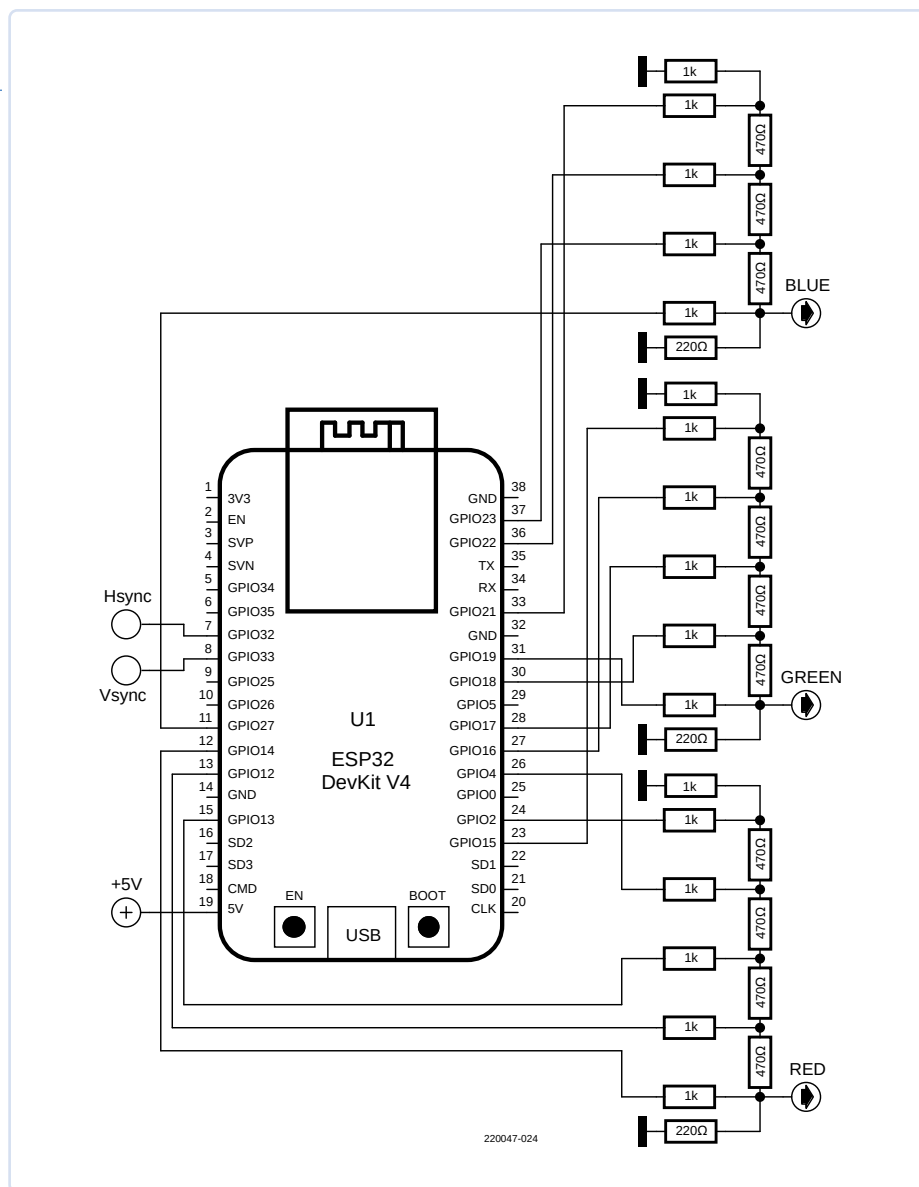
40 MHz precies vier keer zo hoog als wat een ESP32 kan leveren. Dus  $200 \times 600$  pixels bij 60 Hz en 22 bit kleur zouden wiskundig gezien mogelijk zijn qua timing en RAM, hoewel deze pixels dan niet meer rechthoekig zijn maar horizontaal uitgerekt. Voor rechthoekige pixels moet de verticale resolutie worden verviervoudigd, en elke beeldlijn moet vier keer worden uitgevoerd. Het resultaat is  $200 \times 150$  pixels in TrueColor. Een beeld met theoretisch  $800 \times 600$  pixels (figuur 12a) wordt dus vrij

grof weergegeven, maar wel in fraaie kleuren (figuur 12b).

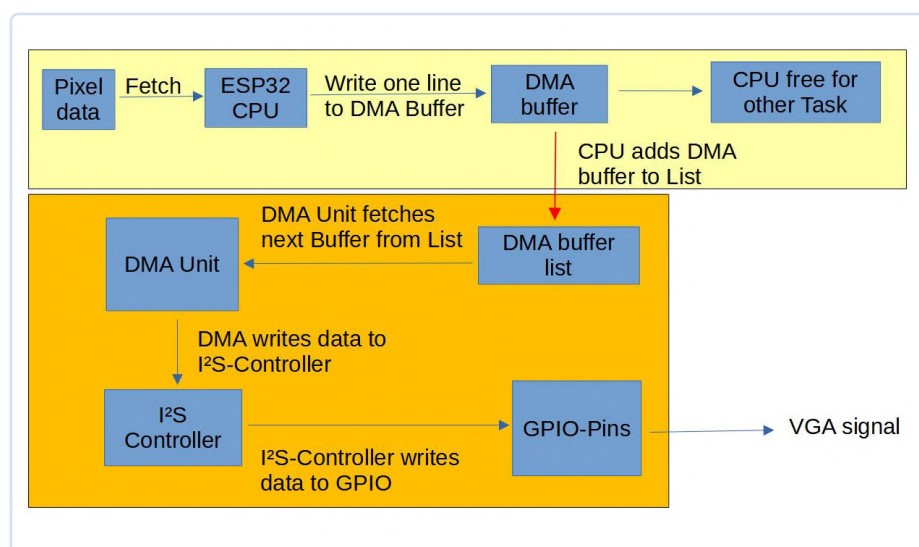
Maar er bestaat een bruikbaar compromis, want de I<sup>2</sup>S-controller kan ook 16bit-woorden verwerken. Van die 16 bit zijn dan 14 bit beschikbaar voor de beeldinformatie en 2 bit voor de synchronisatiesignalen, wat zorgt voor 16.384 kleuren die kunnen worden weergegeven. Voor een stabiel VGA-sig-naal moet de APLL van de ESP32 worden gebruikt. Dit zorgt voor een stabiele tijdbasis voor de VGA-timing.



Figuur 12.  $800 \times 600$  (a) wordt  $200 \times 150$  (b) en  $400 \times 360$  pixels (c). (b) en (c) zijn geschaald tot dezelfde afmetingen als (a) voor een betere vergelijking.



Figuur 13. Opwekken van een VGA-sigitaal via een R2R-DAC met de ESP32.



Figuur 14. Pixel-uitvoer met ESP32.

Op deze manier kunnen beelden in 14bit-kleur met resoluties van 320×240, 360×400, of 480×400 pixels worden gerealiseerd.

Voor de uitvoer van de RGB-signalen moet de 14bit-kleurinformatie van digitaal naar analoog worden geconverteerd. Een R2R-DAC is hiervoor een eenvoudige en doeltreffende oplossing. De schakeling van **figuur 13** laat zien hoe een dergelijke aanpak kan worden geïmplementeerd. De afbeelding van figuur 12a ziet er veel beter uit met een resolutie van 360×400 pixels (**figuur 12c**).

Om de gegevens uit het RAM op te halen en weer te geven, bestaan er enkele trucs: om de gegevens naar de I<sup>2</sup>S-controller te transporteren, wordt de DMA-controller (*Direct Memory Access*) van de ESP32 gebruikt om de belasting van de CPU te verminderen. Hij kan gegevens van de ene plaats naar de andere transporteren zonder de CPU daarvoor lastig te vallen. De bron, de bestemming, de hoeveelheid gegevens, het overdrachtstype en de startvoorwaarde volstaan voor dit doel. De ESP32 bereidt de gegevens voor elke beeldlijn voor en slaat ze op in een lijnbuffer. Naast de zichtbare pixels wordt ook de informatie voor de horizontale en verticale synchronisatie toegevoegd. Wanneer een lijn is voorbereid, draagt de DMA-controller de gegevens uit de buffer over aan de I<sup>2</sup>S-controller. De CPU's van de ESP32 kunnen gedurende deze tijd andere taken uitvoeren. De ruwe volgorde van het proces is te zien in **figuur 14**. Deze procedure maakt ook de uitvoer van afbeeldingen mogelijk zonder het volledige beeld in het geheugen te moeten opslaan. Hiervoor moet de CPU van de ESP32 echter de pixels voor de volgende beeldlijn kunnen berekenen in de tijd dat de pixels van de huidige beeldlijn worden uitgevoerd.

De VGA-bibliotheek van bitluni voor het ESP32 Arduino-framework is beschikbaar op GitHub [9]. De ESP32Lib bibliotheek kan echter veel meer dan alleen VGA uitvoeren. Hij omvat ook functies voor sprites [10] en een 3D-renderer. Verder maakt de FabGL-bibliotheek [11] niet alleen VGA-uitvoer met de ESP32 mogelijk, maar levert hij ingrediënten voor een compleet systeem voor graphics en audio. Er zijn zelfs video-tutorials voor FabGL op YouTube beschikbaar [12].

## VGA op de Raspberry Pi Pico

Dankzij de RP2040-SoC is de Raspberry Pi Pico niet alleen klein, maar ook zeer veelzijdig.





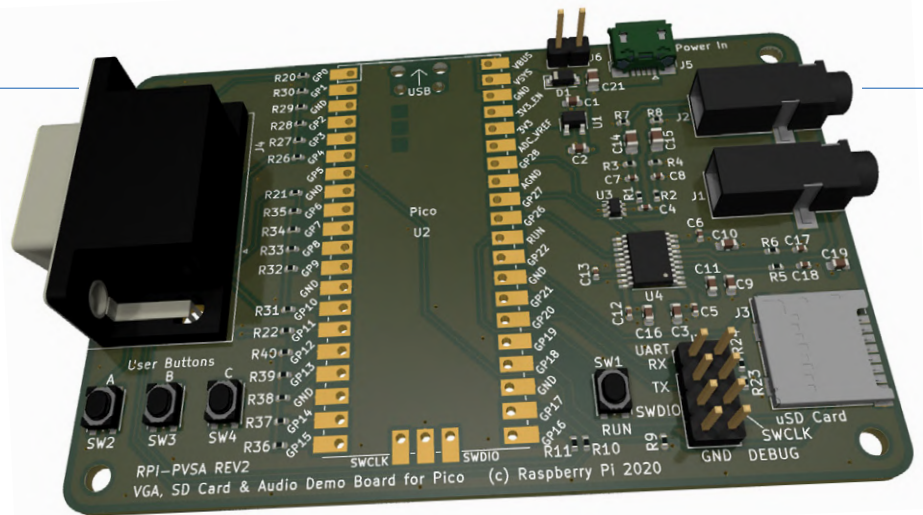
Met twee CPU-kernen, 264 KB intern RAM plus extern flash-geheugen is dit board goedkoop, maar toch geschikt voor vele projecten. Al bij de release van de Raspberry Pi Pico werd het genereren van VGA-signalen gepresenteerd met een passend demoboard (**figuur 15**). Inmiddels is het board geactualiseerd zodat ook een Raspberry Pi Pico W gebruikt kan worden. De benodigde bestanden voor het board [13] zijn beschikbaar als KiCad-project, zodat iedereen het kan bouwen.

De RP2040 biedt voldoende geheugen om een beeld van 320×240 pixels bij 16 bit (= 65.536 kleuren) volledig in RAM op te slaan. De PIO-toestandsmachines van de RP2040 zijn hiervoor bij uitstek geschikt.

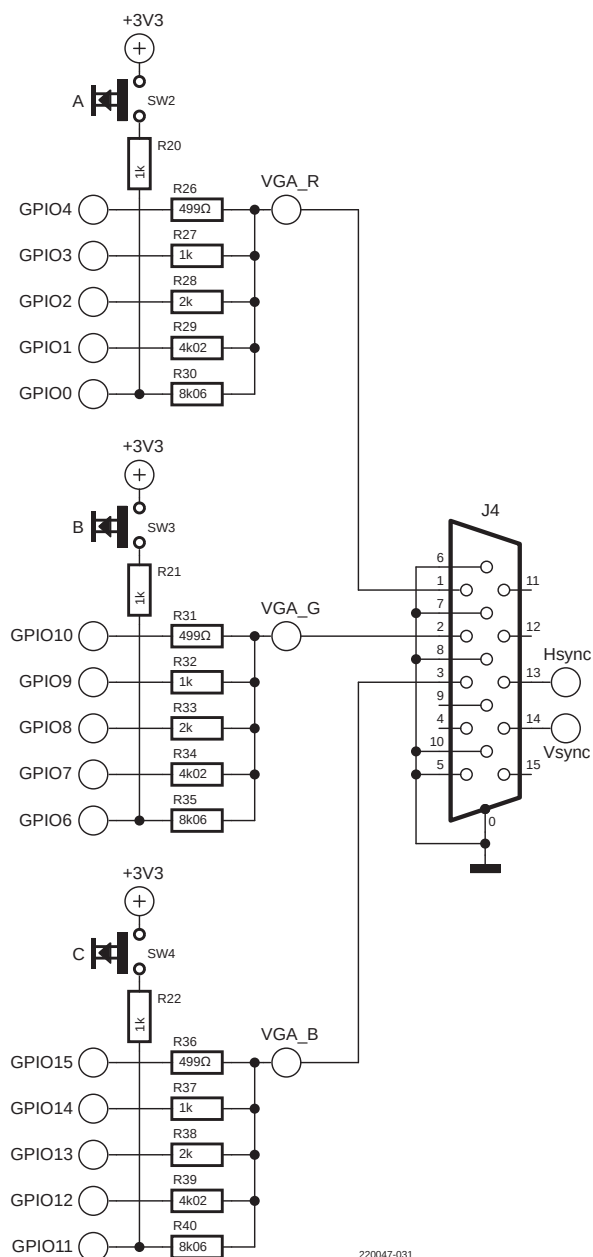
Voor het VGA-sigitaal zijn waarden voor de RGB-niveaus en voor de horizontale en verticale synchronisatie nodig. Voor het genereren van deze signalen met de PIO-toestandsmachines van de RP2040 is alleen wat software nodig. Helaas heeft de RP2040 geen D/A-omzetter aan boord, zodat een externe schakeling nodig is.

Hiervoor is een eenvoudige R2R-DAC of een binair gewogen DAC bruikbaar. Voor een R2R-DAC zijn slechts twee weerstanden nodig, wat de zaken sterk vereenvoudigt. Voor een binair gewogen DAC is voor elk bit een andere weerstandswaarde nodig. De schakeling van het demoboard met VGA-uitgang in **figuur 16** is afkomstig uit een PDF-bestand met de titel "Hardware design with RP2040" [14] van Raspberry Pi Ltd. Voor RGB565-kleuren worden telkens 5 GPIO-pinnen gebruikt voor de kleuren rood en blauw, terwijl voor groen zes pinnen nodig zijn voor de in totaal 65.536 kleuren. De weerstandswaarden vloeien voort uit de structuur van de DAC, de benodigde uitgangsspanning en de afsluitweerstand van de monitor.

De waarden van de analoge RGB-signalen liggen tussen 0 V en 0,7 V. De RGB-ingangen van een monitor zijn afgesloten met 75 Ω naar massa. Daarom moeten de spanningsdelers maximaal 0,7 V kunnen leveren voor elk van de drie kleuren met behulp van de 3,3 V van de GPIO's. Dit resulteert in weerstandsverhoudingen van 1:2:4:8:16. Om 0,7 V uit 3,3 V te maken, heb je een spanningsdeler nodig met een verhouding van 3,7:1. Per kleur worden vijf of zes weerstanden parallel geschakeld. De parallelschakelingen moeten uitkomen op 278 Ω om bij een voeding van 3,3 V een



Figuur 15. VGA-, SD-card- en audio-demokaart voor de Pico.



Figuur 16. Een deel van de VGA-basischakeling voor de Raspberry Pi Pico.



Figuur 17. Pimoroni VGA Base voor de Raspberry Pi Pico.

maximum van 0,7 V te krijgen bij een afsluitweerstand van 75  $\Omega$ .

Met weerstanden uit de E96-reeks zijn waarden van 499, 1000, 2000, 4020 en 8060  $\Omega$  beschikbaar met een tolerantie van 1%. Parallel geschakeld resulteert dit in 258  $\Omega$ , wat leidt tot een voldoende nauwkeurige 0,74 V over de afsluitweerstand.

Als je een volledig geassembleerd board op prijs stelt, kun je de Pimoroni Pico VGA Demo Base kopen (figuur 17) – zie ook het kader Gerelateerde producten. Naast de VGA-uitgang zijn een audio-codec en een SD-kaartslot aanwezig.

### Pixeluitvoer met de RP2040

Voor de uitvoer van een VGA-sigitaal moeten de beeldgegevens worden gegenereerd en

opgeslagen. Het RAM-geheugen van de RP2040 is voldoende om een beeld van 320×240 pixels bij 16-bit kleur op te slaan. De pico-playground GitHub-repo [16] bevat geschikte demo's.

Voor een eenvoudige uitvoer wordt 150 KB geheugen gereserveerd, waarin de beeldgegevens worden opgeslagen als RGB565-waarden. Het beeld wordt regel voor regel uit het geheugen gelezen, waarna voor elke regel de juiste timing voor de beelduitvoer wordt gegenereerd. Figuur 18 toont een vereenvoudigd stroomdiagram.

Dit soort beelduitvoer vanuit RAM zorgt ervoor dat beide CPU's vrij blijven voor andere taken. Er hoeft slechts een nieuwe beeldlijn met geschikte timing te worden aangeleverd. De hiervoor verantwoordelijke *pico\_scanvi-*

*deo*-bibliotheek is te vinden in [17]. Deze maakt het mogelijk meer dan één beeldlijn vooraf voor te bereiden. Het aantal beeldlijnen is configureerbaar, evenals de schaling van de schermresolutie. Zo kun je bijvoorbeeld 320×200 pixels uitvoeren met een resolutie van 640×480 pixels. Het bufferen van de beeldlijnen maakt ook andere trucs mogelijk, zoals het genereren van beelden "on the fly". Dit is min of meer een race tegen de tijd of tegen de elektronenbundel.

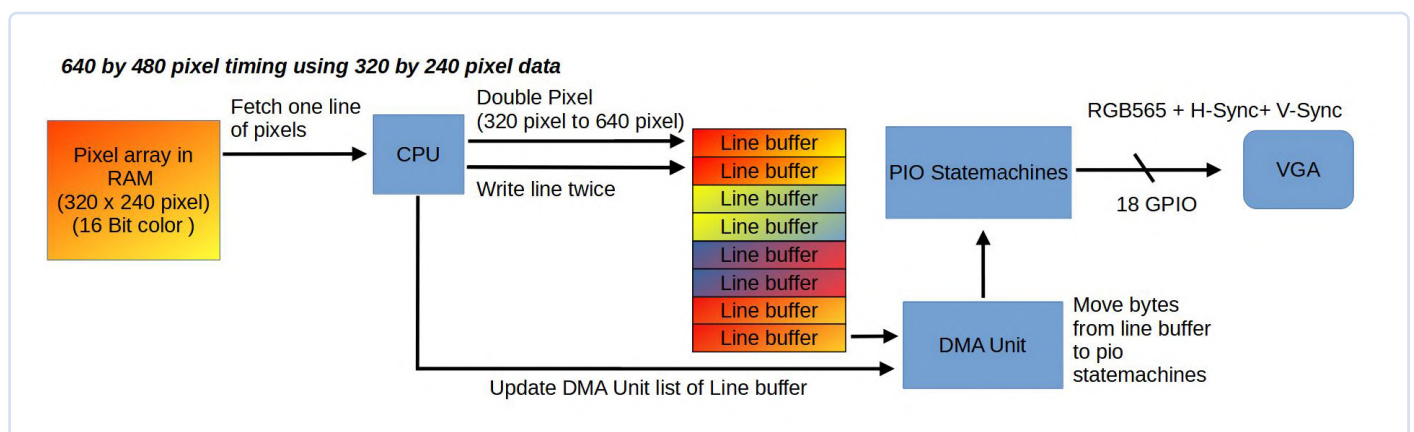
### Tekenen met de Raspberry Pi Pico

De Raspberry Pi Pico en andere boards gebaseerd op de RP2040 kunnen ware tekenaars zijn. In figuur 19 bewegen diverse sprites over het scherm. De uitvoer is 640×480 pixels bij 16bit-kleur. Dit is meer dan in het RAM van de Raspberry Pi Pico past. Daarom worden alle pixels opnieuw getekend voor elke nieuw uit te voeren afbeelding.

Terwijl deze demo met basisfuncties 2D-graphics op het scherm kan tekenen, gaat Miroslav Nemecek's PicoVGA bibliotheek een stuk verder. De grafische uitvoer is beperkt tot



Figuur 19. Bewegende sprites.



Figuur 18. Stroomdiagram VGA-uitvoer met Raspberry Pi RP2040.





Figuur 20. Ants op een Raspberry Pi Pico.



Figuur 21. Pac-Man op een Raspberry Pi Pico.



Figuur 22. Pseudo-3D met een Raspberry Pi Pico.

8 bit, waardoor meer IO-pinnen vrij blijven en minder RAM nodig is voor een compleet beeld (75 in plaats van 150 KB). De GitHub-repo van de bibliotheek bevat verschillende demo's en kleine spelletjes [18]. **Figuur 20** toont het spel *Ants* dat draait op de Raspberry Pi Pico. **Figuur 21** toont de klassieker *Pac-Man*. Een korte beschrijving van de bibliotheek is te vinden op de Duitse MagPi-pagina [19]. De Raspberry Pi Pico bevat functionele eenheden die zelfs pseudo-3D-effecten mogelijk maken (**figuur 22**). Dit type display is ooit beroemd geworden door het SNES (Super Nintendo Entertainment System) met spellen als *Super Mario Kart* (**figuur 23**) of *F-Zero* als Mode 7 graphics [20]. De bijbehorende bibliotheek is beschikbaar op GitHub [21].

## DVI

DVI is een in 1999 geïntroduceerde interface [22] die beeldgegevens digitaal aan een monitor levert zonder kwaliteitsverlies. Een typische witte connector voor tweekaans DVI-D is te zien in **figuur 24**. De exacte DVI-specificaties zijn te vinden in een PDF-bestand op [23].

Toen de eerste LCD-monitoren betaalbaar werden, voerden veel grafische kaarten de beeldinformatie nog uit via de VGA-interface. Maar bij Full HD (1920×1080 pixels) was het beeld ten gevolge van de analoge signaalinterface niet echt scherp. Met DVI was er al snel een wijdverbreide digitale interface die een LCD-monitor rechtstreeks, zonder conversieverlies, kon voorzien van hoge-resolutie beelden in 24bit-kleur.

## 8b/10b, TMDS en 1:10-klok

Terwijl bij VGA drie van de vijf signalen analoog waren, is bij DVI alles digitaal. Vier differentiële lijnpaaren ( $\pm$ Data 0,  $\pm$ Data 1,

$\pm$ Data 2 en  $\pm$ Clock) vormen het minimum voor eenkanaals beeldoverdracht met DVI. De RGB-beeldgegevens worden verzonden via Data 0, Data 1 en Data 2. Voor een beeld in 24bit-kleur wordt elke kleur verzonden met 8 bit. Bovendien is er een kloksignaal dat helpt om de kleuren aan ontvangerzijde te combineren tot volledige pixels.

Digitale transmissie maakt gebruik van de TMDS-methode (Transition-Minimized Differential Signaling) [24], die dankzij de gegevenscodering zorgt voor een robuust signaaltransport en weinig elektromagnetische straling.

De gebruikte coderingsmethode is 8b/10b [25], waarbij slechts 460 van de 1024 mogelijke combinaties worden gebruikt om de 8bit-informatie voor de drie kleuren rood, groen en blauw te coderen. Verschillende 8bit-waarden kunnen dus door twee combinaties worden weergegeven. Vier combinaties worden gebruikt voor het signaleren van C0 en C1 voor horizontale en verticale synchronisatie. Bij de uitvoer van de pixels voor een lijn wordt het aantal nullen en enen in de gegevensstroom in evenwicht gebracht om DC-neutrale gegevensstromen te verkrijgen, hetgeen mogelijk wordt gemaakt door de dubbele weergave van sommige 8bit-waarden.

Zelfs wanneer de beeldinformatie volledig digitaal wordt doorgegeven, zijn er nog steeds horizontale en verticale synchronisatiesignalen. Dit zijn echter geen specifieke datalijnen meer, maar ze worden met behulp van C0 en C1 gecodeerd in de Data 0-stroom. De DVI-gegevensstroom is zo ontworpen dat hij gemakkelijk kan worden omgezet in een analoog VGA-signaal.

Het kloksignaal in DVI komt niet overeen met de bitsnelheid voor rood, groen en blauw,

maar heeft een verhouding van 10:1 en komt overeen met de pixelklok als gevolg van de 8b/10b-codering. Voor de laagste DVI-resolutie van 640×480 pixels bij 60 Hz is een pixelklok van 25,175 MHz of een bitsnelheid van 251,75 MHz nodig voor de kleurinformatie, net als bij VGA. De oorsprong van VGA is dus nog steeds duidelijk te bespeuren bij DVI.

## DVI met de Raspberry Pi Pico

Uitgaande van de eerder opgegeven laagst mogelijke resolutie moet een Raspberry Pi Pico drie datastromen uitvoeren met 251,75 Mbps. Dit is eenvoudigweg niet mogelijk met zijn maximale klok van 133 MHz, omdat dan bijna 2 bits per klokcyclus moeten worden uitgevoerd voor elk van de drie kleuren-datastromen.



Figuur 23. Super Mario Kart met pseudo-3D effect.

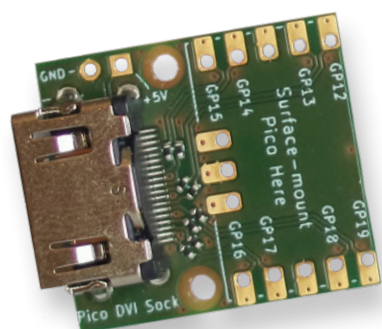


Figuur 24. DVI-D aansluiting op een PC (bron: dr. Thomas Scherer).





Figuur 25. PicoDVI – RP2040 met DVI-video-uitgang (bron: Wren6991 / <https://elektor.link/picodviimg>).

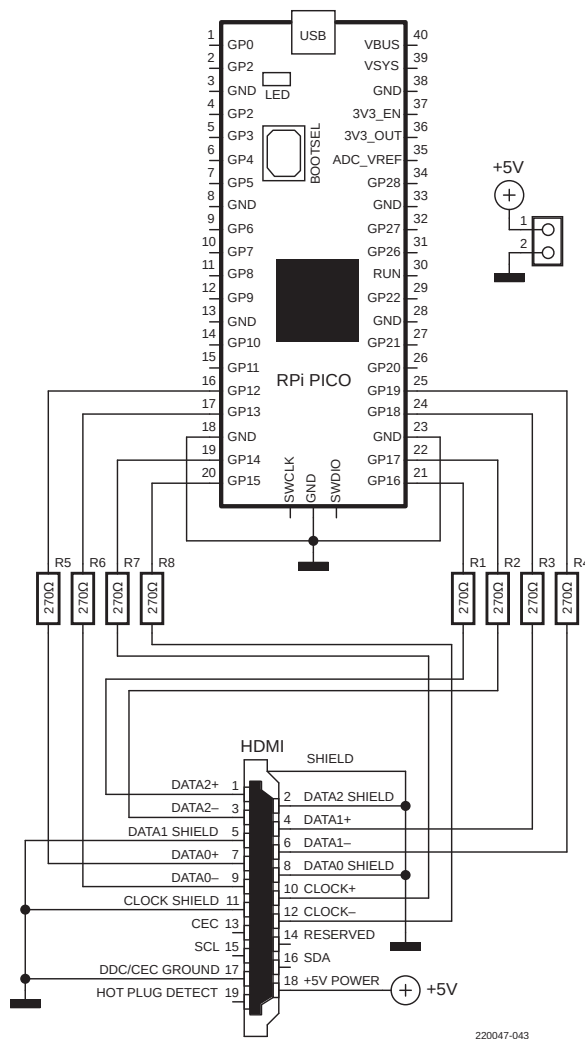


Figuur 26. Pico DVI Sock voor de Raspberry Pi Pico.

De Raspberry Pi Pico zou dus al bij deze horde kunnen falen. De RP2040 is echter extreem overklokbaar. Een klok van 251,75 MHz voor beide kernen is bij kamertemperatuur mogelijk zonder merkbare storingen. De GPIO-pinnen van de RP2040 moeten deze enorme snelheid echter ook bijhouden. Bovendien moeten de beeldgegevens nog steeds worden gecodeerd als een TMDS-stream en moeten de gegevens dus goed worden voorbereid. Luke Wren [26] kon aantonen dat een RP2040 hiertoe in staat is (zie **figuur 25**); lees voor meer informatie het interview bij [27].

Een van de twee kernen van de RP2040 is voor ongeveer 60% bezig met TMDS-codering. Bovendien worden drie van de acht PIO-toestandsmachines gebruikt om de gegevens met de vereiste 251,75 MHz uit te voeren. Sommige functies van de DMA-controller ontlasten de CPU-kernen van datatransport. Zo blijft één core volledig vrij en is deze volledig beschikbaar voor eigen software. Een DVI- of HDMI-aansluiting (**figuur 26**) plus

acht weerstanden volstaan om de Raspberry Pi Pico elektrisch aan te sluiten op de DVI- of HDMI-ingang van een monitor. De vereiste bedrading (**figuur 27**) is vrij eenvoudig. In de GitHub repo van de PicoDVI [26] beschrijft Luke Wren hoe de interpolator van de RP2040 kan worden gebruikt om TMDS-codering te versnellen, wat rekentijd bespaart. De interpolator was oorspronkelijk bedoeld om pseudo-3D graphics te genereren, zoals in *Pilotwings* (**figuur 28**). Hij is echter flexibel genoeg om de CPU-kernen te ontlasten bij het genereren van de drie TMDS-datastromen uit de RGB-gegevens. Hoe komen de pixels dan van het RAM naar de monitor? Voor uitvoer met een resolutie van 640×480 pixels wordt een beeld van 320×240 pixels opgeschaald. Daarom hoeven slechts 240 lijnen 60 keer per seconde te worden TMDS-gecodeerd. Elke lijn wordt tweemaal uitgevoerd, en dienovereenkomstig zijn de pixels tweemaal zo groot in verticale richting.



Figuur 27. Schema van de Pico DVI Sock.

Om dit te bereiken worden de ruwe gegevens regel voor regel aangeleverd, zoals bij de uitvoer van een VGA-sigitaal. De lijnen kunnen afkomstig zijn van een framebuffer in de Raspberry Pi Pico of kunnen "on the fly" worden aangeleverd. Net als met VGA zijn ook met een DVI-uitgang interessante projecten mogelijk. Hier is slechts één van de twee CPU-kernen, samen met zijn interpolator, bezig met het genereren van het DVI-sigitaal.



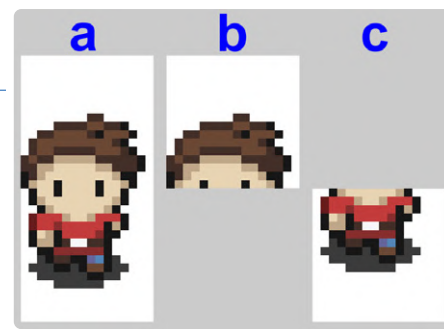
Figuur 28. Pseudo-3D in het spel Pilotwings.



Figuur 29. Walker-demo voor de Raspberry Pi Pico.



Figuur 30. Tiles in de Walker-demo.



Figuur 31. Spelpersonage met transparante achtergrond (a) en bovenste (b) en onderste (c) sprite van het spelpersonage.

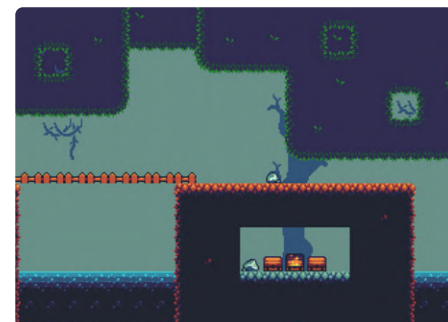
De Walker-demo (figuur 29) laat indrukwekkend zien dat een Raspberry Pi Pico nog voldoende reserves heeft om wat magie op het scherm te toveren.

### Sprites, tiles en tile maps met de RP2040

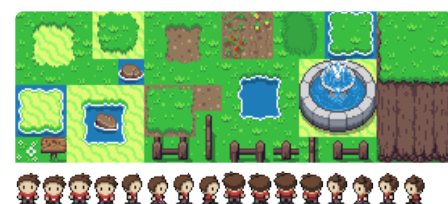
Voor zowel VGA- als DVI-uitvoer is er een grafische bibliotheek die sprites ondersteunt. In de Walker-demo was het voltooide beeld samengesteld uit veel kleine elementen die tegels (tiles) worden genoemd (zie figuur 30). Elke tile heeft een grootte van 16×16 pixels. Deze tiles zijn bedoeld als achtergrondafbeelding en hebben meestal geen transparantie-informatie – in tegenstelling tot het spelerspersonage (figuur 31a), dat is samengesteld uit twee sprites (figuur 31b en figuur 31c). Zowel de achtergrond als de sprites kunnen afzonderlijk worden geplaatst. Met de juiste selectie kunnen mooie achtergronden worden samengesteld, zoals in figuur 32.

De tile maps vormen hier de basis voor de grafische uitvoer (figuur 33), waarvan de 16×16 pixel tiles worden opgehaald en vervolgens door de Raspberry Pi Pico samengesteld tot een afbeelding. Maar waarom zou je zo iets doen? Omdat het aardige van sprites, tiles en tile maps het relatief bescheiden geheugen-gebruik en de flexibiliteit bij het gebruik is. Naast de tile map hoeft alleen de positie waarop de tiles of sprites moeten worden getekend in het geheugen te worden opgeslagen. Animaties of scrollen zijn op deze manier ook mogelijk door relatief eenvoudige updates van sprite-posities.

Voor de broncode van de demo's wordt verwezen naar de bijbehorende GitHub-repository [28] van Luke Wren. De hier getoonde afbeeldingen zijn van de apps `tiles_and_sprites`, `tiles_parallax` en `tiles`. Als je een geschikte Raspberry Pi Pico met DVI-uitgang hebt, kun je de apps zelf uitproberen en met de broncode spelen.



Figuur 32. 2D-platformdemo.



Figuur 33. Tile maps (bron: ArMM1998 / <https://elektor.link/ZeldaTiles>).



WÜRTH  
ELEKTRONIK  
MORE THAN  
YOU EXPECT

WE meet @ embedded world  
Hall 2, Booth 110

#### Noise free e-mobility

e-Mobility is no longer a question of tomorrow and the number of e-vehicles is increasing day by day. Handling EMI noise is becoming more and more crucial, when it comes to design new electronic devices and systems. Würth Elektronik offers a wide range of EMC components, which support the best possible EMI suppression for all kinds of e-mobility applications. With an outstanding design-in support, catalogue products ex stock and samples free of charge, the time to market can significantly be accelerated. Besides ferrites for assembly into cables or harnesses, Würth Elektronik offers many PCB mounted ferrites and common mode chokes as well as EMI shielding products.

[www.we-online.com/emobility](http://www.we-online.com/emobility)

- Large portfolio of EMC components
- Design-in-support
- Samples free of charge

- Orders below MOQ
- Design kits with lifelong free refill



## Een microcontroller als tekenaar?

Tot nu toe hebben we alleen demo's gepresenteerd – geen complete spellen of toepassingen. De demo's geven echter genoeg informatie om grafische elementen toe te voegen aan je eigen projecten. Zowel de ESP32 als de Raspberry Pi Pico zijn goedkope kaarten waarmee verrassend goede grafische uitvoer mogelijk is. Of het nu gaat om Pac-Man of Tetris via DVI, en of het nu gaat om VGA of composiet video, met elke generatie nemen de mogelijkheden en de pure prestaties van MCU's toe. Hun hardware maakt geweldige effecten mogelijk, zoals 3D-rendering of bewegende kleurenbeelden met sprites. Net als destijds met de home computers van de jaren '80 kunnen met veel creativiteit en een paar trucs interessante beelden

uit deze beperkte hardware worden geperst. De microcontrollers moeten gewoon op de juiste manier worden gechoreografeerd bij het tekenen, net als alle ingrediënten die in dit artikel worden beschreven. Het is nu aan jou om er spannende projecten mee te maken. Post je creaties op Elektor Labs [29], en dan kun je technische problemen en mogelijkheden bespreken met andere Elektor-leden en nuttige feedback krijgen. ◀

220614-03

### 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](mailto:mathias.claussen@elektor.com) of naar de redactie van Elektor via [redactie@elektor.com](mailto:redactie@elektor.com).



### Gerelateerde producten

- **Pimoroni Raspberry Pi Pico VGA Demo Base (SKU 19769)**  
<https://elektor.nl/19769>
- **Raspberry Pi Pico (SKU 19562)**  
<https://elektor.nl/19562>
- **DVI Sock for Raspberry Pi Pico (SKU 19925)**  
<https://elektor.nl/19925>
- **ESP32-PICO-Kit V4 (SKU 18423)**  
<https://elektor.nl/18423>

## WEB LINKS

- [1] vgax-bibliotheek: <https://github.com/smaffer/vgax>
- [2] vgaxua-bibliotheek: <https://github.com/smaffer/vgaxua>
- [3] VGA-PacMan (Duits): <https://niklas-rother.de/artikel/vga-pacman>
- [4] AAtmel ATmega videogenerator met SDRAM: <http://tinyvga.com/avr-sdram-vga>
- [5] VGA timingtabellen: <http://tinyvga.com/vga-timing/640x480@60Hz>
- [6] ESP32 VGA: <https://bitluni.net/esp32-vga>
- [7] ESP32 technisch referentiehandboek (PDF): <https://elektor.link/ESP32TechMan>
- [8] VGA signaaltiming: <http://tinyvga.com/vga-timing>
- [9] ESP32Lib van bitluni: <https://github.com/bitluni/ESP32Lib>
- [10] Sprite (computer graphics) – Wikipedia: [https://en.wikipedia.org/wiki/Sprite\\_\(computer\\_graphics\)](https://en.wikipedia.org/wiki/Sprite_(computer_graphics))
- [11] FabGL-bibliotheek: <http://fabglib.org>
- [12] FabGL-videotutorials (YouTube): <https://elektor.link/YTFabGLTuts>
- [13] RPI-PVSA VGA, SD-kaart en audio-demoboard voor Pico – KiCad project (zip): <https://elektor.link/RP2040VGAKiCad>
- [14] Hardware-ontwerp met RP2040 (PDF): <https://elektor.link/RP2040HardwareDesign>
- [15] Pimoroni Raspberry Pi Pico VGA Demo Base: <https://elektor.com/pimoroni-raspberry-pi-pico-vga-demo-base>
- [16] pico-playground: <https://github.com/raspberrypi/pico-playground>
- [17] pico\_scanvideo-bibliotheek: [https://elektor.link/pico\\_scanvideoLib](https://elektor.link/pico_scanvideoLib)
- [18] PicoVGA-bibliotheek: <https://github.com/Panda381/PicoVGA>
- [19] MagPi – VGA grafische bibliotheek voor de Raspberry Pi Pico (Duits): <https://elektor.link/MPPicoVGALib>
- [20] Mode 7 – Wikipedia: [https://en.wikipedia.org/wiki/Mode\\_7](https://en.wikipedia.org/wiki/Mode_7)
- [21] Interpolator-demo – GitHub: <https://elektor.link/GHInterpolatorDemo>
- [22] DVI – Wikipedia: <https://elektor.link/WPDVI>
- [23] DVI-specificatie (PDF): <https://elektor.link/DVISpec>
- [24] Transition-minimized differential signaling – Wikipedia: <https://elektor.link/WPTMDS>
- [25] 8b/10b codering – Wikipedia: [https://en.wikipedia.org/wiki/8b/10b\\_encoding](https://en.wikipedia.org/wiki/8b/10b_encoding)
- [26] Luke Wren's PicoDVD: <https://github.com/Wren6991/PicoDVI>
- [27] Mathias Claussen interviewt Luke Wren, "DVI met de RP2040", Elektor maart/april 2023: <https://elektormagazine.nl/220575-03>
- [28] PicoDVI-demo's, inclusief tiles\_and\_sprites "walker" – GitHub: <https://elektor.link/GHPicoDVIDemos>
- [29] Elektor Labs: <https://elektormagazine.com/labs>



# Het Metronom real-time besturingssysteem

een RTOS voor AVR-processoren

Dieter Profos, dr. sc. techn. ETH (Zwitserland)

Voor veel taken – zoals het verwerken van continue signalen – moeten microcontrollers taken uitvoeren met een exacte timing. Het hier gepresenteerde real-time besturingssysteem is (ook) geschikt voor AVR-controllers met weinig geheugen. Je moet enkele beperkingen accepteren, zoals puur in assembler programmeren, maar dat is nog altijd een goed compromis voor projecten waar snelheid en real-time mogelijkheden belangrijk zijn.

## Waarom wéér een besturingssysteem?

Met het verschijnen van kleine en zeer kleine processoren of controllers werden processen automatiseerbaar waarvoor het gebruik van een 'echte' computer vroeger niet te rechtvaardigen was. Deze microcontrollers hoeven geen periferie aan te sturen (toetsenbord, muis, scherm, harde schijf enzovoort), zodat de besturingssystemen kunnen worden teruggebracht tot wat strikt noodzakelijk is om de verwerking van gebruikersprogramma's te organiseren.

De meeste besturingssystemen zijn ontworpen om zoveel mogelijk programma's zo efficiënt mogelijk en (vanuit het oogpunt van de gebruiker) gelijktijdig te laten draaien. De situatie is echter anders wanneer continue, tijdgebonden signalen moeten worden verwerkt: dit vereist processen die met een exacte timing verlopen. De `delay()` functie in Arduino is dan bijvoorbeeld niet meer nauwkeurig genoeg, omdat deze alleen wachttijden genereert zonder rekening te houden met de voor de verwerking benodigde runtimes, wat duidelijk merkbaar wordt bij samplingtijden van 1 ms of korter.

Daarom moeten de volgende twee problemen worden opgelost:

- Bepaalde taken moeten precies op vooraf bepaalde tijden worden uitgevoerd, andere alleen als er tijd voor over is.
- Elke onderbreekbare taak heeft zijn eigen stack nodig voor

het bufferen van de registerinhoud. Bij kleine controllers is de geheugenruimte echter vrij beperkt: bijvoorbeeld 750 bytes bij de ATtiny25 of 1 K bij de ATmega8.

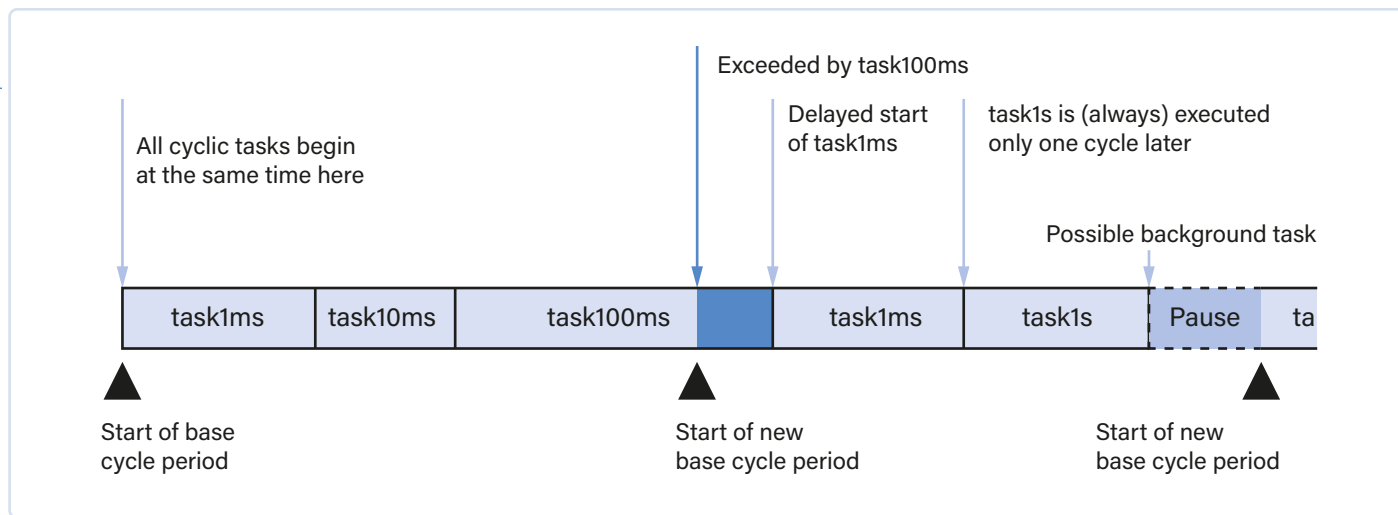
Het hier gepresenteerde besturingssysteem *Metronom* kan worden gedownload van de Elektor-website [1] als open-source software onder de BSD-2 licentie; een release via GitHub staat voor de komende maanden op de planning.

## Cyclische taken

*Metronom* is speciaal ontworpen voor de uitvoering van zogenaamde *cyclische* taken met nauwkeurig bepaalde tijdsintervallen (met maximaal 8 verschillende cyclustijden). Er is precies één taak per cyclustijd; als meerdere activiteiten met onafhankelijke inhoud in dezelfde cyclus moeten worden uitgevoerd, moeten ze in dezelfde taak worden gecombineerd.

De cyclustijden worden als volgt gegenereerd:

- de basis-cyclusperiode (bijvoorbeeld 1 ms) wordt vastgelegd met behulp van de processorhardware (kristal of interne RC-oscillator, hardware- en interruptgestuurde softwareteller), en
- de andere cyclustijden worden gegenereerd door een keten van



Figuur 1. Een opeenvolging van verschillende cyclische taken (extreem geval).

tellers, en wel zo dat elke cyclisperiode een veelvoud is van de vorige (de standaardinstelling is bijvoorbeeld 1 ms → 10 ms → 100 ms → 1 s).

Een belangrijk kenmerk van het hier gepresenteerde besturingssysteem is dat cyclische taken elkaar niet kunnen onderbreken (*non-preemptive*). Enerzijds zorgt dit ervoor dat de timing van deze taken zo nauwkeurig mogelijk is, en anderzijds verliest de processor geen 'onproductieve tijd' door taakwisseling. En aangezien elke cyclische taak – als die eenmaal gestart is – zonder onderbreking (behalve door interrupts) wordt voltooid voordat de volgende cyclische taak wordt gestart, kunnen alle cyclische taken samen dezelfde stack gebruiken.

Maar wat gebeurt er als een taak langer loopt dan de basis-cyclustijd (wat eigenlijk door de programmeur vermeden moet worden) of als er meerdere cyclische taken in dezelfde basis-cyclustijd zijn gestart, waarbij de som van de runtimes de basis-cyclustijd overschrijdt (wat zeker legitiem is)? Hier komt een andere eigenschap van Metronom tot uiting: de cyclische taken hebben prioriteiten. De taak met de kortste cyclustijd heeft de hoogste prioriteit, de 'op één na snelste' taak heeft de op één na hoogste prioriteit – enzovoort. Als niet alle gelijktijdig gestarte cyclische taken binnen de basis-cyclustijd kunnen worden afgewerkt, wordt de lopende taak na het verstrijken van de basis-cyclustijd voortgezet – maar dan wordt eerst de 'snelste' taak met de hoogste prioriteit opnieuw uitgevoerd, en pas daarna worden andere eerder gestarte cyclische taken opnieuw uitgevoerd.

Een voorbeeld: de implementatie van de cyclustijden zorgt ervoor dat alle cyclische taken elke seconde gelijktijdig worden gestart. Wat er dan gebeurt is te zien in **figuur 1**.

Dit betekent dat, als absolute bovengrens, elke cyclische taak niet langer mag duren dan de kortste cyclustijd – dat wil zeggen 1 ms in ons voorbeeld. Dit komt overeen met ongeveer 6000 instructies bij een ATtiny op 8 MHz en ongeveer 14000 instructies bij een ATmega op 16 MHz (de rest van de instructies wordt – gemiddeld – gebruikt door het besturingssysteem zelf en voor de afhandeling van interrupts).

## Achtergrondtaken

Er zijn echter bepaalde bewerkingen die door hun aard langer duren:

- schrijven naar EEPROM duurt bijvoorbeeld enkele milliseconden (meestal ongeveer 3,3 ms), dat wil zeggen een onacceptabel lange tijd voor een basis-cyclustijd van 1 ms;
- tekst versturen met 9600 bd is niet haalbaar met een basis-cyclustijd

van 1 ms, omdat zelfs de verzending van één enkel teken al langer dan 1 ms duurt;

- wanneer langere berekeningen (bijvoorbeeld geïmplementeerde rekenkundige bewerkingen!) moeten worden uitgevoerd of tekenreeksen moeten worden verwerkt, duurt dit vaak te lang binnen een cyclische taak en worden de tijdgebonden processen geblokkeerd.

Dit betekent dat er nog een manier moet zijn om dergelijke processen te delegeren naar een soort onderbreekbare taak. Hiervoor wordt een combinatie van twee methoden gebruikt:

- het gebruik van interrupts in plaats van actief wachten: hierdoor kan het wachten op het einde van een bewerking (bijvoorbeeld het verzenden van een teken) worden 'gedelegeerd' aan de hardware; deze methode wordt gebruikt voor interruptgestuurde bewerkingen. Dit lost de problemen op voor de overdracht van een enkel teken of voor het schrijven van een enkele waarde naar de EEPROM, maar niet voor het wachten op het einde van de complete bewerking (zoals het verzenden van een volledige tekst);
- uitvoering van achtergrondtaken: een achtergrondtaak loopt alleen gedurende de tijd die niet door cyclische taken wordt bezet. Bovendien kan hij op elk moment worden onderbroken, zodat hij de tijdige uitvoering van de cyclische taken niet verstoort.

Als een achtergrondtaak echter eenmaal loopt, kan deze niet meer door andere achtergrondtaken worden onderbroken. Er wordt dus maar één achtergrondtaak tegelijk uitgevoerd, en als die moet wachten, wacht de hele verwerking van de achtergrondtaken. Hoewel dit de verwerking van de achtergrondtaken vertraagt, betekent het dat slechts één stackgedeelte hoeft te worden gereserveerd voor alle achtergrondtaken.

Achtergrondtaken worden gekenmerkt door de volgende eigenschappen:

- een achtergrondtaak kan te allen tijde worden onderbroken ten gunste van cyclische taken, maar niet ten gunste van een andere achtergrondtaak;
- de uitvoering van een achtergrondtaak wordt gestart door een startoproep aan de dispatcher;

- achtergrondtaken worden na elkaar uitgevoerd in de volgorde waarin ze zijn gestart;
- achtergrondtaken kunnen wachten op gebeurtenissen (**WAIT\_EVENT**), die bijvoorbeeld door interrupt-gestuurde processen worden getriggerd;
- achtergrondtaken kunnen ook wachten op vooraf gedefinieerde tijden (**DELAY**);
- elke achtergrondtaak kan een startbericht van drie 16bit-woorden krijgen, waarmee zijn taak kan worden gespecificeerd (een vierde woord is gereserveerd voor het startadres van de taak);
- binnen het gebruikersprogramma kan een willekeurig aantal achtergrondtaakroutines bestaan; er kunnen er echter maximaal acht tegelijk worden gestart.

De coördinatie van de taken onderling ('Wanneer mag welke taak draaien?') wordt afgehandeld door wat de *dispatcher* wordt genoemd. Deze voert alle 'administratieve processen' uit, zoals het starten van taken, het in veiligheid brengen/herstellen van de processorregisters of het in- en uitschakelen van interrupts.

## Exceptions

Aangezien microcontrollers meestal geen tekstgeoriënteerde periferie hebben, is debuggen erg moeilijk, vooral voor tijdsafhankelijke functies, aangezien breakpoints en dergelijke de timing volledig verstoren. Daarom biedt de kernel van het besturingssysteem een vereenvoudigd mechanisme voor het afhandelen van exceptions, dat in twee fasen is verdeeld:

- een globaal **try-catch** gebied vangt alle uitzonderingen (exceptions/errors) op die zich voordoen in de kernel en in de rekenkundige emulaties. De uitzonderingsspecifieke gegevens kunnen worden opgeslagen in de EEPROM en/of uitgevoerd via USART; daarna voert het besturingssysteem een totale systeem-RESET uit (inclusief **user-reset**). Dit exception-gebied is altijd actief;
- daarnaast kan een toepassingsgericht **try-catch** gebied worden gebruikt, dat alleen betrekking heeft op het eigenlijke gebruikersprogramma. De afhandeling van dergelijke exceptions is in eerste instantie hetzelfde als hierboven: de betreffende gegevens worden opgeslagen en/of uitgevoerd via USART; vervolgens wordt een door de gebruiker op te geven 'application restart'-routine uitgevoerd (subroutine **user\_restart**).

## Interruptverwerking

Interrupts worden op vier verschillende manieren behandeld:

- de reset-interrupt wordt gebruikt door het besturingssysteem en is niet direct toegankelijk voor de gebruiker. Omdat de gebruiker deze interrupt echter ook nodig heeft om zijn eigen processen te initialiseren, roept het besturingssysteem na zijn eigen initialisatie de subroutine **user\_init** aan, die de gebruiker kan vullen met zijn applicatiespecifieke initialisatiecode;
- timer/counter0 wordt gebruikt voor het genereren van de basis-klok voor alle cyclische processen; hij is daarom niet toegankelijk voor de gebruiker;
- voor het gebruik van de EEPROM en de USART biedt het

besturingssysteem kant-en-klare driverblokken, die tijdens het genereren van het besturingssysteem kunnen worden geïntegreerd (zie hieronder). De gebruiker kan echter ook zijn eigen serviceroutines aan deze interrupts koppelen of ze gewoon open laten wanneer ze niet worden gebruikt;

- alle andere interrupts zijn direct beschikbaar voor de gebruiker. Voor elke interrupt moet een interrupt service routine en een interrupt initialisatie routine worden gespecificeerd; als meer dan één interrupt bij een apparaat hoort (bijvoorbeeld timers of USART), is een gedeelde initialiseroutine voldoende. Hiervoor activeert de gebruiker de overeenkomstige parameters in het generatorbestand en voegt de inhoud van de relevante initialisatie- en serviceroutines in zijn gebruikersprogramma in;
- ongebruikte interrupts worden automatisch 'onderschept' door het besturingssysteem.

## Programmeeromgeving

Om efficiencyredenen is Metronom geschreven in AVR assembler (Atmel/Microchip) en gaat er dus van uit dat gebruikersprogramma's ook in assembler zijn geschreven; een interface naar C is niet geïmplementeerd. Wel is er een bibliotheek met vele subroutines, bijvoorbeeld voor 8-bit en 16-bit rekenen (vier rekenkundige basisbewerkingen); een 16-bit breukenbibliotheek is in voorbereiding.

Om het programmeerwerk te vergemakkelijken, zijn alle aanroepen van het besturingssysteem beschikbaar als macro's. Om benamingsbotsingen te voorkomen, geldt de volgende naamgevingsconventie: alle variabelen en jump targets binnen het besturingssysteem en de bibliotheken beginnen met een underscore (" \_ ") – daarom mogen alle namen in het gebruikersprogramma alleen met letters beginnen. Andere tekens dan letters, cijfers en de underscore zijn niet toegestaan. De algemene structuur van Metronom en het bijbehorende gebruikersprogramma is te zien in **figuur 2**.

## Aanroepen van het besturingssysteem

Voor de volledige lijst van besturingssysteem-aanroepen wordt verwezen naar de referenties aan het eind van het artikel; hier volgt slechts een ruw overzicht:

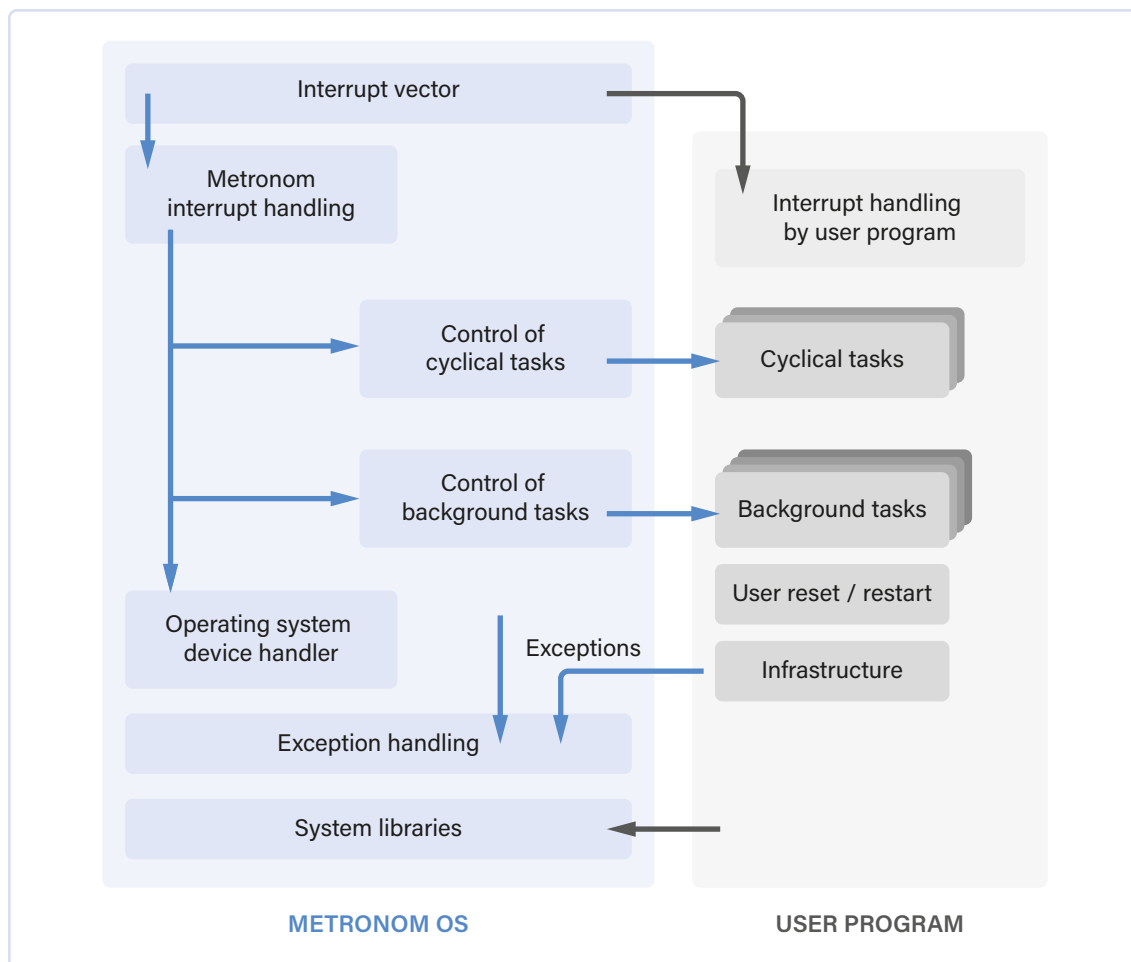
### Macro's voor het afhandelen van exceptions

- **KKTHROW** zorgt voor een systeembrede exception, dat wil zeggen na het opslaan/uitvoeren van de exception-informatie wordt het hele systeem opnieuw opgestart;
- **KTHROW** zorgt voor een exception die beperkt is tot het gebruikersprogramma, dat wil zeggen na het opslaan/uitvoeren van de exception-informatie wordt alleen de gebruikerssubroutine **user\_restart** uitgevoerd; daarna worden de cyclische taken opnieuw gestart.

### Macro's voor het gebruik van achtergrondtaken

- **\_KSTART\_BTASK** start een achtergrondtaak;
- **\_KDELAY** brengt de aanroepende achtergrondtaak in slaap gedurende n (0 tot 65535) ms;
- **\_KWAIT** brengt de aanroepende achtergrondtaak in slaap, waaruit hij kan worden gewekt door middel van...
- **\_KCONTINUE**.





Figuur 2. Algemene structuur van Metronom en het gebruikersprogramma.

### Macro's voor 8-bit en 16-bit rekenen

Over het algemeen worden voor rekenkundige bewerkingen van allerlei aard de registers r25:r24 gebruikt als accumulator en r23:r22 als geheugen voor de tweede operand (indien nodig). Voor dit doel zijn meer dan 20 verschillende functies beschikbaar, zoals `_mul8u8` voor een 8x8-bit vermenigvuldiging of `_abs16` voor een 16-bit absolute waarde. Verder zijn er veel pseudocodes voor laden en opslaan, zoals `_ld16` (16-bit getal in de accumulator laden).

### Macro's voor EEPROM-gebruik

- > `_KWRITE_TO_EEPROM` voor het schrijven naar EEPROM;
- > `_KREAD_FROM_EEPROM` voor het uitlezen van EEPROM.

### Macro's voor USART-gebruik

- > `_KWRITE_TO_LCD` is een specifiek USART-stuurprogramma, dat de benodigde stuurkarakters voor een 2x16 LC-display toevoegt aan de weer te geven tekst;
- > `_KREAD_FROM_USART` (nog niet geïmplementeerd).

### Systeemgenerator SysGen

Voor het genereren van een systeem (dat wil zeggen de volledige code) wordt een speciale systeemgenerator SysGen gebruikt, die ook deel uitmaakt van het package. SysGen is niet beperkt tot Metronom, maar kan ook worden gebruikt voor algemene generatortaken.

Sommige lezers vragen zich misschien af waarom er een aparte systeemgenerator is ontwikkeld, aangezien er een grote verscheidenheid aan preprocessoren en macrogeneratoren bestaat. Maar voor het genereren van het Metronom-besturingssysteem zijn de functio-

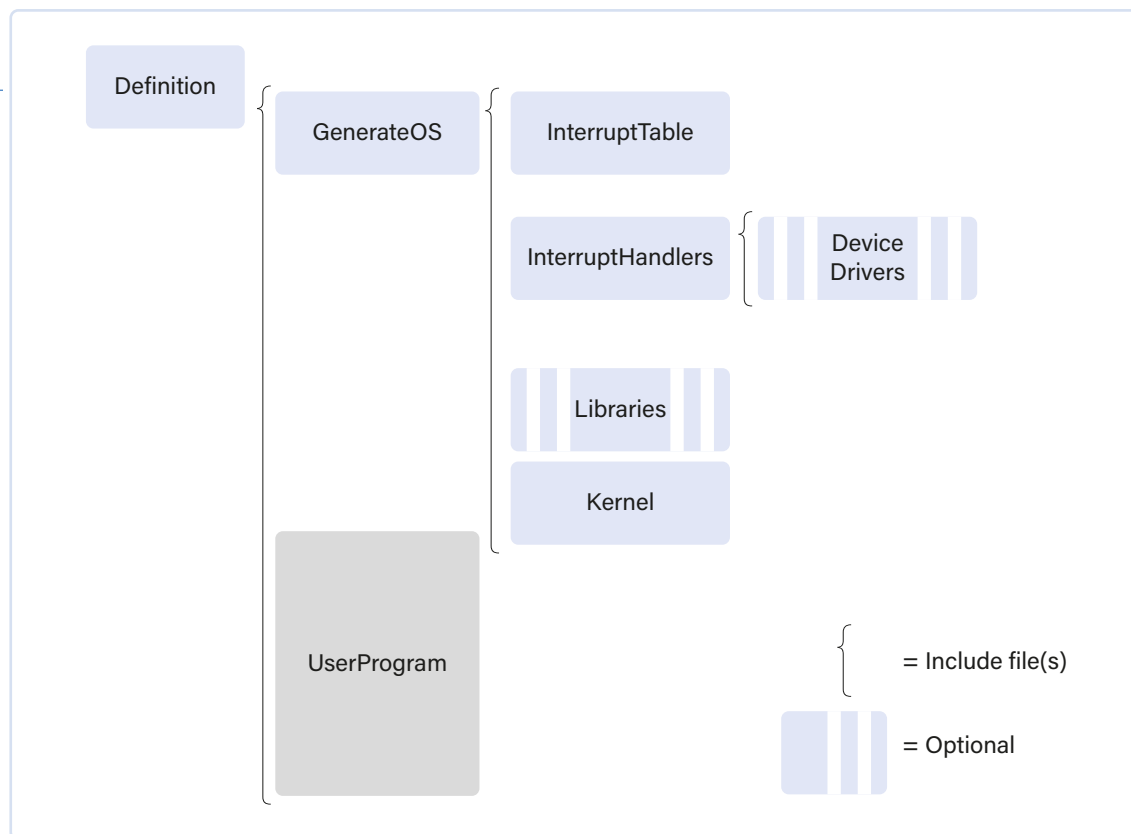
naliteiten van de preprocessoren in Atmel Studio en standaard C niet voldoende. Met name omdat de preprocessor geen string-'arithmetic' ondersteunt, is het niet mogelijk een 'default directory' of 'library directory' op te geven en van daaruit bestanden te selecteren die zich daarin bevinden. Een zoektocht op *Stack Overflow* toonde aan dat andere mensen hetzelfde probleem hebben als ik, maar geen van de huidige preprocessoren kan ermee overweg.

De preprocessor van Atmel Studio (net als de GNU-preprocessor) biedt de volgende functies voor het samenstellen van de vereiste bestanden:

- > `define / set <name> = <numeric_expression>`
- > `if ... elif ... else ... endif`, ook genest
- > `ifdef, ifndef`
- > `include <path> | exit`

De volgende functionaliteiten ontbreken:

- > `<path>` kan alleen worden doorgegeven als een vaste string, maar een stringuitdrukking van (een willekeurig aantal) partiële strings, zowel stringvariabelen als stringconstanten, zou nodig zijn;
- > `define` en `set` kunnen alleen numerieke waarden toekennen, geen strings, geen aaneenschakeling van strings, en ook geen logische uitdrukkingen;
- > voor de (eenmalige) opname van bibliotheekprogramma's zijn de macromogelijkheden in AVRASM of de C-preprocessor niet voldoende; aangezien macro's in AVRASM geen include-state-ments mogen bevatten, is de automatische opname van bijvoorbeeld emulatie-routines niet mogelijk.



Figuur 3. Generatorstructuur van Metronom-systemen.

Dit leidt tot de volgende reikwijdte van de functies:

- > `define / set <name> = <numeric_expression> | <string expression> | <boolean expression>`
- > `if ... elif ... else ... endif`, ook `genest`;
- > `ifdef, ifndef` wordt omgezet in `if ifdef(...)` of `ifndef(...)` en kan dus ook worden gebruikt binnen Booleaanse uitdrukkingen;
- > `include <string expression> | exit`
- > `message | error`
- > `code` (om regels code te maken);
- > `macro/endmacro` met passende parameterlabeling;
- > een bijkomende eis is dat instructies van bestaande preprocessoren kunnen worden gemengd met die van SysGen zonder elkaar te storen.

Het SysGen-programma kan ook worden gedownload van de opgegeven webpagina [1]. SysGen is geschreven in Java (versie 12) en vereist een corresponderende Java-installatie om te kunnen draaien.

### Programmeren met Metronom

Om de gebruiker de vervelende taak van het doorwerken van de broncode van het besturingssysteem te besparen, is het hele besturingssysteem zo opgebouwd dat het automatisch wordt gegenereerd. Dat betekent dat de gebruiker alleen het definitiebestand en – indien nodig – de door de gebruiker geprogrammeerde interruptroutines hoeft in te vullen; waar ze thuishoren en hoe ze worden gekoppeld wordt automatisch door het generatorproces verzorgd.

In zijn basisvorm bestaat een gebruikerssysteem uit de in **figuur 3** geschetste onderdelen.

De interrupttabel en de kernel worden altijd als één geheel opgenomen in het resulterende algemene programma. In het geval van de

device handlers en de libraries daarentegen worden alleen de werkelijk benodigde delen opgenomen.

Ter illustratie van het generatorproces is het generatorscript van een van mijn eigen projecten te zien in **listing 1**. ◀

210719-03

### Vragen of opmerkingen?

Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur via [profos@rspd.ch](mailto:profos@rspd.ch) of naar de redactie van Elektor via [redactie@elektor.com](mailto:redactie@elektor.com).



### Gerelateerde producten

- > **A. He and L. He, *Embedded Operating System* (SKU 19228).**  
[www.elektor.com/19228](http://www.elektor.com/19228)
- > **A. He and L. He, *Embedded Operating System* (e-boek, SKU 19214).**  
[www.elektor.com/19214](http://www.elektor.com/19214)

### WEBLINKS

- [1] Generatorbestanden voor Metronom, generatorprogramma SysGen, documentatie:  
<http://www.elektormagazine.com/labs/metronom-a-real-time-operating-system-for-avr-processors>



## Listing 1

```
; *****
; Master Definition
; *****
; Stand: 03.05.2022
;
; This file contains all informations required to generate your user system for
; AVR processors.
; It consists of three parts:
;
; 1. Definitions
; A bunch of variable definitions defining which functionalities to include.
; This part must be edited by the user.
;
; 2. An $include statement for the actual generation of the operating system.
; DO NOT MODIFY THIS STATEMENT!
;
; 3. The $include statement(s) adding the user program(s).
; This part must be edited by the user.
;

; *****
; PART 1: DEFINITIONS
;
; This script is valid for ATmega8, ATmega328/P and ATtiny25/45/85 processors.
; If you want to use it for any other processors feel free to adapt it accordingly.

$define processor = "ATmega8"

; Remove the ; in front of the $set directive if you want to use the EEPROM
; $set _GEEPROM=1
; if you want to write your own routines to write to the EEPROM use the following
; definition:
; $set _GEEPROM=2
; Enabling this definition will insert an appropriate JMP instruction to your
; interrupt service routine e_rdy_isr in the InterruptHandlers.asm file
; Remove the ; in front of the $set directive if
; ... you want to output serial data via the USART, or
; ... you want exception messages to be sent outside via the USART
; $set _GUSART=1
; if you want to write your own routines to use the USART
; use the following definition instead
; $set _GUSART=2
; Enabling this definition will enable the interrupt service routines usart_udre_isr,
; usart_rxc_isr and usart_txc_isr in the InterruptHandlers.asm file.

; -----
; Define the division ratios of the time intervals for cyclic tasks
; The definition shown here is the standard preset for 1 : 10 : 100 : 1000 ms
; The first ratio being 0 ends the divider chain.
.equ _KRATIO1 = 1000000000 ; 1 -> 10ms
.equ _KRATIO2 = 1000000000 ; 10 -> 100ms
.equ _KRATIO3 = 1000000000 ; 100ms -> 1s
.equ _KRATIO4 = 0000000000 ; end of divider chain
.equ _KRATIO5 = 0
.equ _KRATIO6 = 0
.equ _KRATIO7 = 0
; NOTE: Do not remove "superfluous" .EQU statements but set them to 0 if not used!

; -----
; Define the constants used for generation of the 1ms timer interrupt
; IMPORTANT: The following definitions depend on the processor being used
; and the frequency of the master clock
```





```
$if (processor == "ATmega8")
; The definitions below are based on a system frequency of 12.288 MHz (crystal)
; This frequency has been chosen in order to use the crystal also for USART@9600 Bd
;
; set prescaler for counter0 to divide by 256, yields 48kHz counting freq for Counter0
.equ _KTCCR0B_SETUP = 4
; Counter0 should divide by 48 in order to produce interrupts every 1ms;
; since counter0 produces an interrupt only at overflow we must preset
; with (256-48) - 1 = 207.
$code ".equ _KTCNT0_SETUP = " + (256 - 48) - 1

$elif ... similar for other processors
;
$endif
;
; -----
; Define the characteristics of USART transmission
; (if you don't use the USART just neglect these definitions):
$set fOSC = 12288000
$set baud_rate = 9600
$code ".equ _KUBRR_SETUP = " + (fOSC / (16 *baudrate) - 1)

; parity: 0 = Disabled,
; (1 = Reserved), 2 = Enable Even, 3 = Enable Odd
.equ _KPARITY = 0

; stop bits: 0 = 1 stop bit, 1 = 2 stop bits
.equ _KSTOP_BITS = 1

; data bits transferred: 0 = 5-bits, 1 = 6-bits, 2 = 7-bits, 3 = 8-bits, 7 = 9-bits
.equ _KDATA_BITS = 3
;
; -----
; Connect a user defined interrupt handler (except RESET and Timer0)
; by removing the ; in front of the appropriate $set directive;
; don't change any names but just let the $set statement as is

; Interrupts for ATmega8
; $set _kext_int0 = 1 ; IRQ0 handler
$set _kext_int1 = 1 ; IRQ1 handler/initializer is supplied by user
; $set _ktim2_cmp = 1 ; Timer 2 Compare Handler
; $set _ktim2_ovf = 1 ; Timer 2 Overflow Handler
; $set _ktim1_capt = 1 ; Timer 1 Capture Handler
;
; etc. etc. etc.

;
; *****
; PART 2: GENERATING THE OPERATING SYSTEM
;
; .LISTMAC
;
; $include lib_path + "\GenerateOS.asm"
;
;
; *****
; PART 3: ADD THE USER PROGRAM
;
; $include user_path + "\MyApplication.asm"
;
$exit
```

# DVI op de RP2040

vraaggesprek met Luke Wren, chipontwikkelaar bij Raspberry Pi

**Mathias Claußen (Elektor Lab)**

Luke Wren, een van de engineers van de Raspberry Pi RP2040-microcontroller, heeft verbazingwekkende dingen bereikt door deze kleine chip tot het uiterste te drijven. Elektor-engineer Mathias Claussen wilde meer weten, vooral over de trucs en hacks die nodig zijn om video-uitvoer uit zo'n kleine chip te persen.

De Raspberry Pi RP2040 is een 'sub-\$1' MCU met verbazingwekkende mogelijkheden. Een van de ontwerpers is Luke Wren, die niet alleen aan de Raspberry Pi RP2040 heeft gewerkt, maar ook heeft laten zien dat de kleine RP2040 tot DVI in staat is (zie het artikel "Video-output met microcontrollers (2)" [1]). Als hij niet bezig is met geheime projecten voor Raspberry Pi, deelt hij enkele van zijn hobbyprojecten met de community op Twitter (@wren6991) en de code op GitHub [2].

**Mathias Claussen: Kun je ons iets over jezelf vertellen?**

**Luke Wren:** Je begint met de moeilijkste vraag! Ik ben engineer bij Raspberry Pi, en als ik dat niet doe, werk ik meestal aan mijn hobbyprojecten, speel ik niet erg goed gitaar, of (meer recent) steek ik tijd in het leren van talen. Ik woon in Cambridge in het Verenigd Koninkrijk, niet omdat het een bijzonder opwindende stad is, maar meer uit luiheid na mijn afstuderen aan de universiteit. Ik heb in Duitsland gewoond toen ik jonger was, maar mijn Duits is nu behoorlijk roestig, dus ik ben blij dat we dit gesprek in het Engels voeren.

**Mathias: Hoe lang ben je al bij Raspberry Pi?**

**Luke:** Ik ben in september 2018 begonnen als werknemer, maar ik was daarvoor stagiair bij Raspberry Pi.

**Mathias: Wat was je rol bij de ontwikkeling van de RP2040?**

**Luke:** Ik werkte aan een gedeelte van het digitale ontwerp – voornamelijk PIO, DMA, XIP-cache, busstructuur en PWM. Ik heb ook gewerkt aan de boot-ROM en de SDK, en natuurlijk moest ik ook mijn steentje aan de documentatie bijdragen.

**Mathias: Video uit een MCU/CPU persen lukte met de Sinclair ZX81, maar DVI is iets nieuws voor een MCU die minder dan 1 dollar kost. De meeste MCU's kunnen VGA genereren, maar wat is de uitdaging als het om DVI gaat?**

**Luke:** Er zijn twee dingen die DVI-D moeilijker maken dan VGA. Het eerste is het serialiseren van de gegevens: de minimale pixelklok voor DVI-D is 25 MHz, en de bitklok is 10 keer zo hoog, dus minimaal stuur je  $3 \times 250$  Mbps differentiële seriële lijnen aan (rood/groen/blauw).

Het tweede is dat DVI-D niet slechts ruwe pixelgegevens verstuurt, maar deze eerst codeert. De codering is eenvoudig in hardware, maar ietwat lastig in de software, vooral wanneer die software de brute snelheid van de seriële uitgang moet bijhouden. Al het andere is vergelijkbaar. Het is eigenlijk gewoon DPI door een snellere pijplijn. (Noot van de redactie: Display Pixel Interface is een parallelle RGB-pixelinterface – inclusief pixelklok – om pixelgegevens naar een beeldscherm te transporteren.)

**Mathias: Waarom wilde je DVI bij de RP2040 proberen?**

**Luke:** Toen de stress van het silicium eenmaal voorbij was, wilden een paar van ons eens uitproberen hoe hoog we de klokfrequentie van het systeem konden opvoeren. In de praktijk is er enige marge boven de nominale frequentie van 133 MHz. Ik had met DVI-D op een FPGA gespeeld, als onderdeel van mijn RISCBoy-project [3], en toen ik merkte dat er een overlap was tussen de laagste DVI-bitklokfrequenties en de hoogste systeemklokfrequenties bij de RP2040, ging er bij mij een lampje branden. Ik vroeg me af of het mogelijk was – daarom dus.

**Mathias: Wat was de grootste uitdaging om een DVI-uitgang (figuur 1) op de RP2040 te krijgen?**

**Luke:** TMDS coderen. Als je het algoritme in de DVI-specificatie volgt, lukt het nooit om het snel genoeg te krijgen op twee Cortex-Mo+ kernen die op de bitklokfrequentie draaien. Er zijn dus wat trucs en sluiptwegen nodig om het toch mogelijk te maken, en vervolgens wat zorgvuldig handgeschreven code om het voldoende snel te maken. De RP2040 heeft veel geheugen, maar niet genoeg om een compleet frame met TMDS-gecodeerde pixels op te slaan, dus het is een “wedloop tegen de elektronenstraal” bij het coderen.

**Mathias:** Je moest de RP2040 iets overklokken (van 133 MHz standaard naar 252 MHz). Is er een kritisch pad in de chip voor de DVI-signalen (je moet I/O-pinnen ook sneller aansturen dan standaard)?

**Luke:** De eerste beperking waar je bij de RP2040 tegenaan loopt is dat de systeemklok 1:1 moet lopen met de bitklok, dus als je probeert hogere-resolutiemodi te bereiken, dan lopen de processoren gewoon vast. Het kritieke instellingspad voor het systeemklokdomen op de RP2040 wordt gevormd door de adresfase-signalen van de processor naar de SRAM's. En afgezien daarvan zitten we ook vrij dicht bij de grenzen van wat je door die algemene 3V3-paden kunt sturen; als je kijkt naar het oogdiagram van **figuur 2** voor 720p30 (372 Mbps) op mijn GitHub [2], dan kan dat net, maar het is marginaal. Ik betwijfel of je 1080p30 zou zien zonder speciale hardware.

**Mathias:** Hoe cruciaal zijn, naast de hogere snelheid, de PIO's en de interpolator in de RP2040 om DVI aan de praat te krijgen?

**Luke:** Je moet hoe dan ook 3 seriële databits plus hun differentiële complementen op GPIO's met minimaal 250 Mbps beschikbaar maken. De mogelijkheid om de single-ended/pseudo-differentiële conversie in de PIO te doen halveert de DMA-bandbreedte, en het opdelen van de TMDS-lanes in 3 FIFO's is handig als je de codering in software doet omdat je dan je code op maat kunt maken voor de codering van de rood/

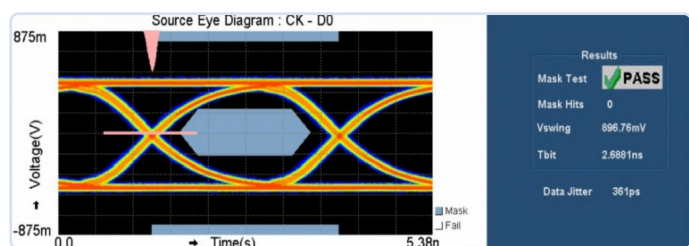


Figuur 1. Raspberry Pi Pico met de Pico DVI Sock (bron: Luke Wren).

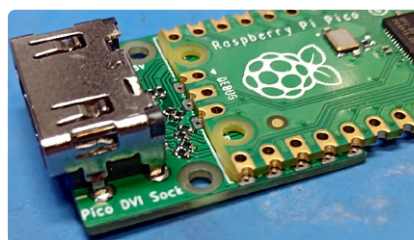
groen/blauw-componenten. Iets als een PIO is dus cruciaal als je geen speciale hardware hebt. De interpolatoren helpen bij het genereren van de adressen bij de TMDS-codering, wat zeker belangrijk is voor sommige van de demo's die je hebt gezien, maar mijn pixel-verdubbelde TMDS-coderingstruc zou nog steeds passen op een enkele Cortex-Mo+ kern zonder interpolatoren.

**Mathias:** Je Pico DVI Sock voor de RP2040 gebruikt een fysieke HDMI-aansluiting (figuur 3), duidelijk gelabeld met DVI-only, dus we krijgen geen audio. Is dat ‘alleen maar’ een licentie-kwestie met het HDMI-consortium?

**Luke:** Er is niets dat je weerhoudt om HDMI data-eilanden toe te voegen en audio uit te voeren. Iemand heeft dat al gedaan gedaan met een NES-emulatorpoort [4][5]! Er zijn geen extra fysieke verbindingen nodig voor de audiosignalen, hoewel je strikt genomen geen HDMI-functies mag gebruiken voordat je het display-datakanaal hebt opgevraagd, dat niet is geïmplementeerd op mijn Pico DVI Sock. De situatie met de HDMI-licentie bezorgt zeker hoofdbrekens, en ik heb mezelf al min of meer vastgezet door de repository “PicoDVI” te noemen, dus dit laat ik aan de community over.



Figuur 2. Oogdiagram voor RP2040 DVI bij 720p30 (bron: Luke Wren).



Figuur 3: Pico DVI Sock voor de Raspberry Pi Pico (bron: Luke Wren).





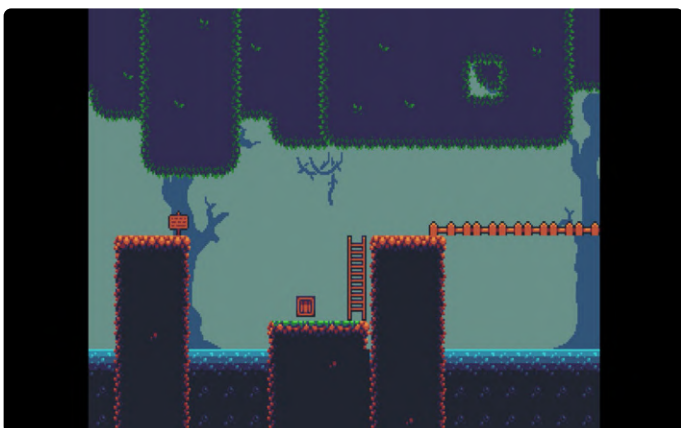
▲  
Figuur 4. Hoge-resolutiebeelden dankzij enkele trucs (bron: Luke Wren).

**Mathias:** Als je de DVI-uitgang gebruikt, hoeveel van de RP2040-resources worden dan daardoor in beslag genomen? Is er tijd over om andere code op de MCU te draaien?

**Luke:** Het hangt af van de videomodus. Voor een RGB565-uitvoer met dubbele pixels gebruik je uiteindelijk ongeveer 65% van een kern voor TMDS-codering en DMA-interrupts; de andere kern is dan compleet beschikbaar voor het genereren van de video en het draaien van de logica van je hoofdprogramma.

*Noot van de redactie: Naast het genereren van pure video, werden enkele toepassingen voor de Pico DVI Sock toegevoegd. Een daarvan is het bewegen van verschillende Eben Upton-gezichten over het scherm. Als we even rekenen, zou een beeld van 640×480 pixels dat als een volledig frame met 8-bit resolutie is opgeslagen ongeveer 308 KB RAM in beslag nemen (meer dan de RP2040 heeft), dus beperken we ons tot maximaal 320×240 met 16bit-kleur (154 KB) in RAM, maar de demo (figuur 4) is niet op die manier gepixeld. Er lijkt dus sprake te zijn van een softwarematige truc.*

▼  
Figuur 5. Sprite-demo voor de Raspberry Pi RP2040.



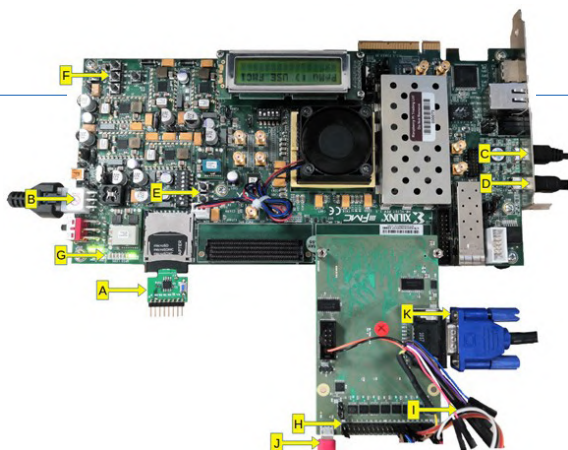
Figuur 6. Walker-demo met DVI-uitvoer.

**Mathias:** Bij de software om een DVI-sigitaal te genereren hoort een bibliotheek die ook sprites verwerkt. Kun je daar meer over vertellen?

**Luke:** Zeker! Wanneer je een video-uitvoer schrijft, is het volgende probleem dat je tegenkomt dat je video moet hebben om daadwerkelijk uit te voeren, en terwijl ik aan PicoDVI werkte, heb ik precies voor dat doel de ARMv6-M sprite-bibliotheek in elkaar gehackt. Het belangrijkste kenmerk van deze bibliotheek is dat er geen framebuffer nodig is om in te renderen, alleen een lijnbuffer. Het renderen gebeurt net 'voor de straal', dus vlak voor de TMDS-codering. Dit betekent dat je video-uitvoerresoluties kunt ondersteunen die niet in het geheugen zouden passen als een gewone framebuffer, en het laat het grootste deel van het geheugen vrij voor de eigenlijke grafische activa. Er zijn enkele redelijk snelle 'blit&fill'-routines, een paar tiling-routines en enkele 'affine transformed sprite'-routines waarmee je geschaalde/geroteerde/gekantelde sprites kunt maken. Genoeg voor sommige spelletjes op het niveau van een Game Boy Advance of zo. (Noot van de redactie: Je kunt een voorbeeld zien in **figuur 5**.)

**Mathias:** Waar haalde je de inspiratie voor de bibliotheek vandaan – en de tijd om die te schrijven?

**Luke:** Ik heb enige tijd gewerkt aan scanline-gebaseerde grafische hardware voor RISCBoy, dus nadat ik dat in hardware had gedaan, was het vrij eenvoudig om het in software te repliceren. Alles in de PicoDVI-repository heb ik in mijn vrije tijd op mijn laptop gedaan, behalve de oogdiagrammen, waarvoor ik een 'scoop op mijn werk heb gebruikt.



Figuur 7: Raspberry Pi RP2040 FPGA-prototype (bron: Luke Wren).

**Mathias:** Er is een door Zelda geïnspireerde Sprite-demo voor de RP2040 (figuur 6). Kun je ons vertellen over het idee daarachter? Een NES of SNES zou speciale hardware gebruiken om zulke beelden samen te stellen, en hier hebben we gewoon twee CPU's die pixels verplaatsen.

**Luke:** Dat is eigenlijk een geportte versie van een van de RISCBoy-demo's. Zoals je zegt is er veel overhead om dit alles in software te doen, en RISCBoy op 36 MHz kan evenveel sprites op het scherm zetten als de RP2040 op 252 MHz [6].

**Mathias:** In de genoemde documentatie van de bibliotheek staat het idee van een Mario Kart-kloon voor de RP2040? Was het gewoon een idee, of ben je aan het sleutelen om het aan de praat te krijgen? Er wordt ook opgemerkt dat de interpolator daarvoor nuttig zou zijn.

**Luke:** Op dit punt moet ik dus toegeven dat de oorspronkelijke reden voor de interpolator in de chip was dat we texture- en tile-mapping in SNES MODE7-stijl wilden doen, hoewel we tussen dat moment en de tape-out wat tijd hebben besteed om er een bruikbaar en krachtiger stuk hardware van te maken. We hebben nooit een echte MK-kloon gemaakt, hoewel je online genoeg voorbeelden kunt zien van mensen die soortgelijke technieken gebruiken; we hebben wel een texture-mapped 3D-kubus met Eben's gezicht erop op een FPGA laten draaien.

**Mathias:** In de documentatie van jullie Pico DVI Sock voor de RP2040 Pico, vermelden jullie een 48-MHz FPGA prototype. Kun je ons daar iets over vertellen?

**Luke:** We moesten 's nachts een FPGA-image maken van de laatste RP2040-broncode zodat we die de volgende dag konden gebruiken voor software-ontwikkeling. Wij hebben gewoon een standaard Virtex 7-ontwikkelboard gebruikt, met een dochterboard om de IO's van de FPGA naar 3,3 V te brengen (figuur 7), en een ander board om een QSPI-flash in de SD-aanslui-

ting te steken, die is aangesloten op de RP2040 XIP-interface [7].

De FPGA is vrijwel een volwaardige RP2040 [8] – de klok/reset-schakeling is vereenvoudigd en de ADC is weggelaten, maar verder is alles aanwezig en functioneel. Dit maakt het een ideaal platform voor software-ontwikkeling, hoewel we voor de eigenlijke verificatie van de chip gebruik maakten van conventionele simulaties en formele modelcontrole.

**Mathias:** Naast de DVI-uitgang zijn/waren jullie bezig met enkele andere projecten. Een daarvan is de PicoStation 3D. Kun je daar iets over vertellen? Als je alle onderdelen bij elkaar kon scharrelen, zou het dan nog steeds hetzelfde ontwerp zijn?

**Luke:** PicoStation 3D (figuur 8) is één van de vele hobby-PCB's die ik in de aanloop naar de lancering van de RP2040 had. Het is een board met een RP2040, een iCE40UP5K FPGA, microSD, audio out, DVI-D out via HDMI, en twee SNES controller-aansluitingen. Ik las op dat moment veel over grafische 3D-hardware en wilde een platform om daarmee te spelen, in de context van een soort kleine spelcomputer.

Het doet me pijn dat ik dat project zo lang op een laag pitje moest zetten, maar naast de problemen met de onderdelen heb ik gewoon te veel andere projecten lopen. Het is allemaal open source, dus ik zou het leuk vinden als iemand anders het idee oppikt en ermee aan de slag gaat.

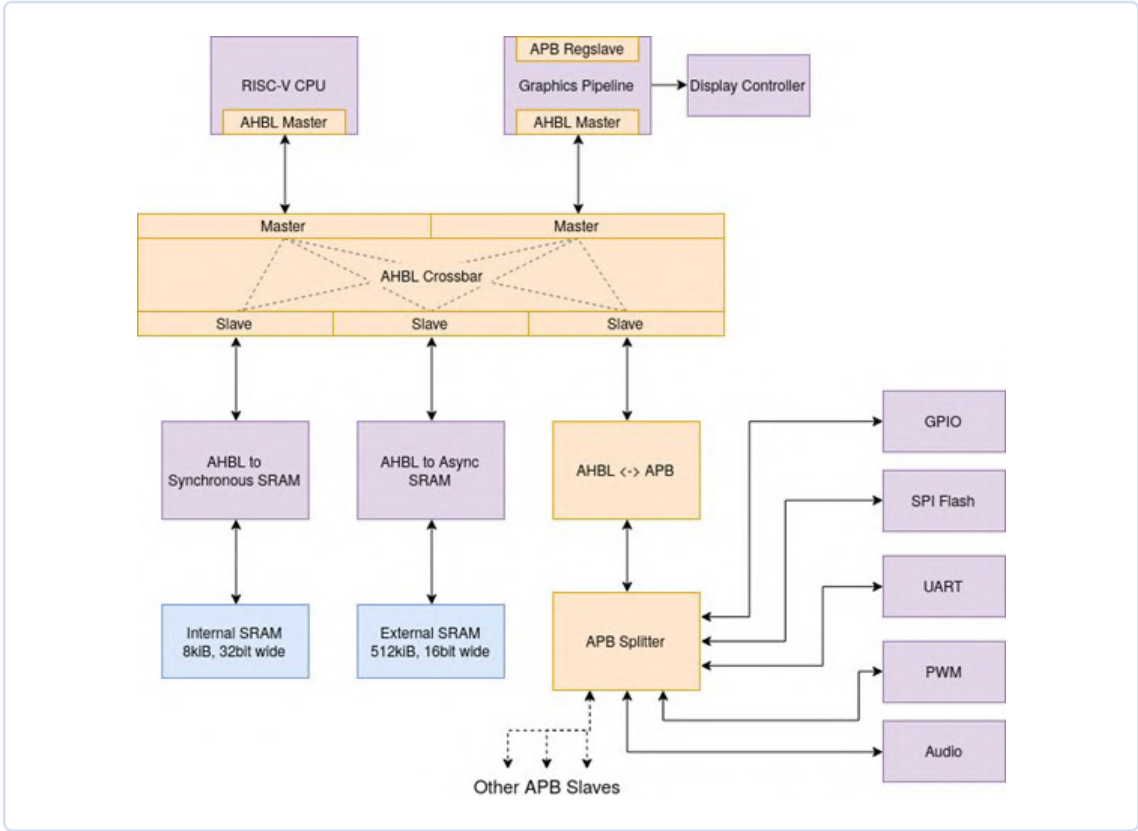
Ik denk dat de keuze van de FPGA precies goed is – hij is klein en traag genoeg om je hard te laten werken voor je demo's, net geen DVI-D-capaciteit, en hij heeft royaal intern geheugen en een handvol 16-bit DSP-tiles, dus het is een briljant platform om te spelen met grafische spelletjes-hardware. Hij is ook goed te

Figuur 8. Prototype van de PicoStation3D (bron: Luke Wren).





Figuur 9. Architectuur van de RISCBoy (bron: Luke Wren).



combineren met de RP2040. Waar ik over na zou willen denken is de IO. Wat als je bijvoorbeeld de DVI naar de microcontroller verplaatst en de SNES-controllers naar de FPGA, en wat als je het audio-circuit wat beter maakt, dat soort dingen.

Ik denk dat de fysieke vormfactor ongeveer goed is, aangezien deze wordt bepaald door de twee SNES controller-aansluitingen.

**Mathias:** Naast de Raspberry Pi-gebaseerde items heb je ook een RISCBoy gemaakt met onder de motorkap een zelfontwikkelde RISC-V Core en andere periferie, zoals een grafische engine (2D-sprite-gebaseerd?). Wat kun je ons daarover vertellen?

**Luke:** RISCBoy is mijn ietwat vertraagde concurrent voor de Game Boy Advance. De volledige hardware past in een iCE40 HX8K met wat extern parallel SRAM

(figuur 9). Uiteindelijk zal er een fysieke versie komen, maar nu is het nog een HX8K ontwikkelboard met wat SRAM en knoppen, en een SPI-display dat er los bij hangt (figuur 10). Ik begon eraan te werken rond de tijd dat ik de universiteit verliet.

Alles is zonder uitzondering vanaf nul geschreven – niet de beste engineering-werkwijze, maar het is geweldig om te leren, en het maakt debuggen leuker wanneer er geen enkel deel van de hardware/software-stack is dat je kunt vertrouwen. Er is een 32-bit processor (Hazard5), wat programmeerbare 2D grafische hardware, en alle infrastructuur om het met elkaar te koppelen.

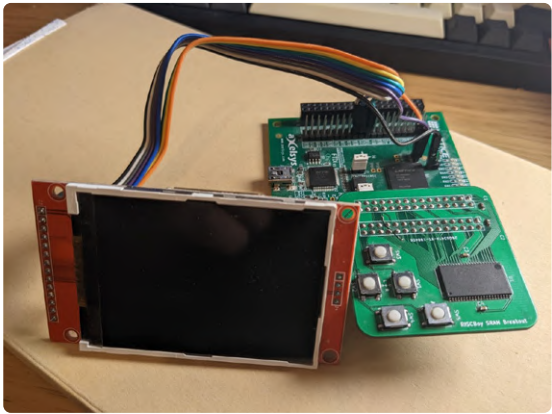
De grafische hardware doet alle gebruikelijke sprites, tiles, affine-transformed sprites/tiles enzovoort, maar doet dit door vanuit het geheugen commandolijsten uit te voeren die beperkte ondersteuning bieden voor control flow en vertakkingen naar subroutines. Tijdens elk frame schrijft de processor de commandolijst voor het volgende frame uit. Er zijn een paar scanline-buffers in de hardware, een waarnaar wordt gerenderd en een die naar het scherm wordt gestuurd, zodat je de bandbreedte, latentie en geheugenruimte van renderen in een framebuffer vermijdt.

Het project staat voorlopig op een laag pitje, maar ik wil het in elk geval ooit voltooien.

**Mathias:** Je RISC-V Core wordt met de dag volwassener. Wat was de oorsprong ervan?

**Luke:** Ik heb momenteel een paar processoren in verschillende stadia van ontwikkeling, maar ik denk dat je de Hazard3 bedoelt. Het is een 3-traps in-order scalaire processor, oorspronkelijk afgeleid van de

Figuur 10. Prototype van de RISCBoy (bron: Luke Wren).







Hazard5-broncode (de RISCBoy-processor). Hazard3 is werkelijk een leerplatform voor mij – het is allemaal goed en wel om RISC-V specificaties te lezen, maar dat is niet hetzelfde als ze daadwerkelijk te implementeren; en door een eenvoudige 3-traps als basis te gebruiken kan ik me concentreren op de periferie omdat de eigenlijke core-pijplijn gewoon werkt.

Toch zijn de prestaties tegenwoordig redelijk competitief (rond de 3,2 CoreMark/MHz) en ik heb veel tijd gestoken in verificatie en documentatie. Het op gang brengen van hardware-debugging en het uitvoeren van end-to-end tests met GDB, OpenOCD en mijn eigen JTAG-transportmodule is waarschijnlijk het hoogtepunt van het project tot nu toe, maar ik heb de hele tijd veel geleerd.

Ik denk dat Hazard3 in de toekomst een componenten-goudmijn zal zijn voor alle toekomstige 32-bit embedded-klasse processoren waaraan ik werk. Ik heb al iemand op Twitter mijn debug-module zien gebruiken om debug-ondersteuning toe te voegen aan de core van iemand anders. De broncode is volledig zelfstandig en zou behoorlijk porteerbaar moeten zijn. Hij heeft ook een Apache-2.0 licentie.

**Mathias: Momenteel is het ontwerp een 32-Bit RISC-V, volledig als open-source. Heb je op een bepaald moment middelen voor de ontwikkeling gekregen van Raspberry Pi (tijd, hardware, software-tools)?**

**Luke:** Ik doe al mijn thuisprojecten in mijn eigen tijd en met mijn eigen hardware, met open-source tools. Op dit moment heb ik thuis een Ryzen 7 5800X, wat helpt bij het uitvoeren van batch jobs. Ik gebruik Yosys voor FPGA-synthese, nextpnr voor place en route en CXXRTL voor simuleren.

Ik gebruik een iCEBreaker (iCE40UP5K) en een ULX3S (ECP5 85F) als referentieplatforms voor Hazard3, hoewel ik nog een behoorlijk aantal andere FPGA-ontwikkelboards heb die ik eigenlijk vaker zou moeten gebruiken. Als ik iets moet debuggen op een FPGA, kan ik prima uit de voeten met mijn Saleae Logic 8 die ik kocht toen ik nog studeerde, maar meestal kan ik vertrouwen op simulaties.

**Mathias: Momenteel ga je met je core van 32 bit naar 64 bit. Wat was (tot nu toe) het lastigste obstakel tijdens de ontwikkeling?**

**Luke:** Ik heb tot nu toe slechts een weekend met RV64 gespeeld, en dat was genoeg om de RV64IC compliance en de debug compliance tests te doen slagen, dus er is geen enorme leercurve – het wordt gewoon groter en verloopt langzamer.

Ik hackte op de Hazard3-codebase voor het gemak, maar als ik serieus werk wilde maken van een RV64-implementatie, dan zouden er enkele micro-architecturale veranderingen nodig zijn. Er is een reden waarom je niet veel drietraps 64-bit processoren ziet, en zeker niet die met de branche decision in trap 2. Hazard3 blijft een 32-bit embedded-klasse processor.

**Mathias: Zijn er plannen om de core in de toekomst op echt silicium uit te proberen? Of zelfs om de core te combineren met een 2D-engine om een RISCBoy64 te krijgen?**

**Luke:** Ik zou graag eens een SkyWater PDK tape-out willen proberen, maar voorlopig concentreer ik me op andere projecten. Ik wil iets interessanter bouwen dan 'gewoon een ander RISC-V systeem' en ik weet nog niet precies wat dat gaat worden. Voor iets als RISCBoy biedt het gebruik van een 64-bit processor niet veel voordelen.

**Mathias: Bedankt voor je tijd, Luke Wren. ◀**

220575-03



### Gerelateerde producten

- > **Raspberry Pi Pico (SKU 19562)**  
[www.elektor.nl/19562](http://www.elektor.nl/19562)
- > **DVI Sock for Raspberry Pi Pico (SKU 19925)**  
[www.elektor.nl/19925](http://www.elektor.nl/19925)



## WEBLINKS

- [1] Mathias Claußen, "Video-output met microcontrollers (2)", Elektor maart/april 2023: <http://www.elektormagazine.nl/220614-03>
- [2] Luke Wren's GitHub-repository: <https://github.com/Wren6991>
- [3] RISCBoy @ GitHub: <https://github.com/Wren6991/RISCBoy>
- [4] pico-infones @ GitHub: <https://github.com/shuichitakano/pico-infones>
- [5] Video @ Twitter: [https://twitter.com/shuichi\\_takano/status/1477702448907419649](https://twitter.com/shuichi_takano/status/1477702448907419649)
- [6] Sprite-demo op de RISCBoy-hardware bij 36 MHz: <https://twitter.com/wren6991/status/1333708886956707840>
- [7] Foto van het QSPI SD-board: <https://twitter.com/wren6991/status/1134719550027632640>
- [8] RP2040-microcontroller: <https://raspberrypi.com/products/rp2040/>



# Display-HAT Mini

toont het weer op de Raspberry Pi

**Clemens Valens (Elektor)**

De display-HAT Mini van Pimoroni is uitgerust met een rechthoekige 2,0" 320x240-pixel LCD, vier drukknoppen en een RGB-LED. Bedoeld voor de Raspberry Pi Zero en Zero 2 W is hij zeer geschikt voor bijvoorbeeld IoT- en domotica-toepassingen.

De display-HAT Mini van Pimoroni [1] is uitgerust met een rechthoekig IPS-LCD (In-Plane Switching) met een diagonaal van 2,0" en een SPI-interface. Hij is bedoeld voor de Raspberry Pi Zero en Zero 2 W, maar omdat hij een standaard 40-pins HAT-connector heeft, kan hij worden aangesloten op elke Raspberry Pi die is uitgerust met zo'n connector – als je voorzichtig bent, omdat de I²C-connector van de HAT ('Breakout Garden-header') in aanraking komt met de display-connector van de Raspberry Pi.

## Specificaties

De resolutie van het scherm is 320x240 pixels (3:2), wat overeenkomt met ongeveer 200 ppi (pixels per inch). De kleurdiepte bedraagt 65K. Gemonteerd op een Raspberry Pi Zero zonder afstandshouders bedraagt de totale hoogte ca. 15 mm. Het scherm wordt geleverd met twee 10 mm hoge afstandshouders en vier kleine schroefjes (vijf in mijn geval). Voor mij zijn deze afstandshouders te hoog, 8,5 mm zou veel beter zijn geweest (maar dat is geen standaardmaat).

De HAT heeft niet alleen een display maar ook vier drukknoppen en een RGB-LED (**figuur 1**). De drukknoppen zitten zeer dicht bij het display, waardoor ze een beetje moeilijk te gebruiken zijn. Ik vermoed dat ze zo zijn geplaatst zijn om het display mechanisch een beetje extra steun te geven.

Hoewel de HAT de toegang tot de 40-pins HAT connector blokkeert, zijn uitbreidingen toch mogelijk dankzij de Qw/ST-connector (Qwiic/STEMMA QT) en de zogenaamde Breakout Garden-header. Beide headers bieden toegang tot de I²C-bus (**figuur 2**).

**Figuur 1.** De display-HAT Mini heeft ook vier drukknoppen en een RGB-LED.



**Figuur 2.** I²C-uitbreidingsconnectoren zijn op de achterzijde te vinden.



### Ondersteund door Python-bibliotheken

Om de display-HAT Mini op een Raspberry Pi te kunnen gebruiken moet je een bibliotheek installeren. Gedetailleerde instructies over hoe dit te doen zijn te vinden op de GitHub-stek van de fabrikant [2]. Voorbeelden voor het gebruik van de display met Pygame en PIL worden ook gegeven.

Ik sloot de display-HAT Mini aan op een Raspberry Pi Zero 2 W met Buster en stuurde hem via SSH aan. Na het installeren van de bibliotheken werkten alle voorbeelden probleemloos.

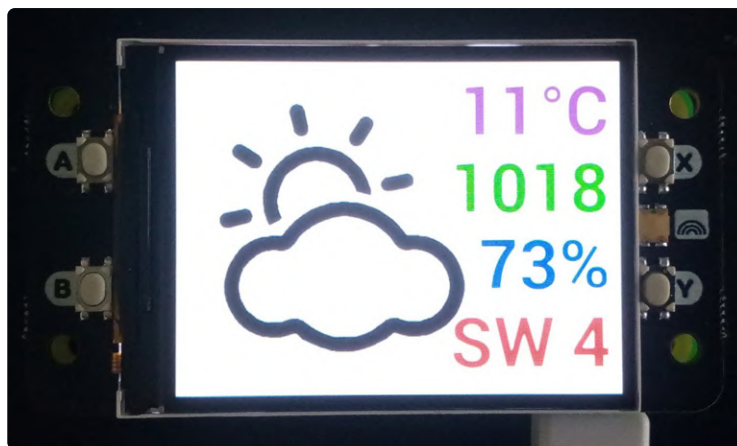
### Het weer op het display

Nu het display functioneert wilde ik er iets mee 'doen'. Op het internet had ik een verzameling fraaie weer-icoontjes gevonden, en daarom besloot ik een display voor weersvoorspellingen te maken in Python3, dat het corresponderende icoontje samen met temperatuur, luchtdruk en -vochtigheid en windrichting en -snelheid toont (**figuur 3**). Deze gegevens kunnen worden verkregen van een online-weerserver; er zijn er genoeg die gratis toegang bieden.

Ik gebruik de drukknoppen 'A' en 'B' om de helderheid van de achtergrondverlichting van het display te regelen. Met knop 'X' kun je kiezen tussen graden Celsius en graden Fahrenheit, terwijl je met knop 'Y' kunt kiezen tussen windrichting in graden of in hoofdletters (bijvoorbeeld 'SW' of 'N'). De RGB-LED geeft enige statusinformatie. Eigenlijk geeft groen aan dat alles goed werkt, maar ik vond de LED veel te helder, zelfs bij zeer lage waarden, en daarom heb ik hem uitgezet. Nu licht de LED alleen op bij een probleem: rood als de weerdata niet konden worden opgehaald en oranje als de weerdata ongeldig zijn.

Merk op dat het scherm standaard ondersteboven staat ten opzichte van de drukknop-bijchriften op de HAT. Om deze reden draait het programma de display-buffer 180° alvorens deze naar het scherm te kopiëren. Het stroomverbruik van mijn systeem bij maximale intensiteit van de achtergrondverlichting bedroeg ongeveer 200 mA.

Mijn code (inclusief de complete verzameling icoontjes) kan worden gevonden op [3].



Figuur 3. De display-HAT Mini is een geweldig display voor IoT of domotica.

### Conclusie

Samenvattend is de display-HAT Mini een coole add-on voor een Raspberry Pi Zero (2 W). De beeldkwaliteit is geweldig, en hij is makkelijk te gebruiken in eigen applicaties. De ondersteunende bibliotheek biedt ook toegang tot de drukknoppen (die naar mijn smaak een beetje te dicht bij het display zitten), de RGB-LED (een beetje aan de felle kant) en de achtergrondverlichting. Merk op dat door het display de HAT iets groter is dan een Raspberry Pi Zero (2 W) (35 mm in plaats van 30 mm), dus let op bij het kiezen van een behuizing. ◀

220126-03

### Vragen of opmerkingen?

Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur via [clemens.valens@elektor.com](mailto:clemens.valens@elektor.com) of naar de redactie van Elektor via [redactie@elektor.com](mailto:redactie@elektor.com).



### Gerelateerde producten

- > **Pimoroni Display HAT Mini for Raspberry Pi Zero (SKU 19990)**  
[www.elektor.nl/19990](http://www.elektor.nl/19990)
- > **HyperPixel 2.1 Round - Hi-Res Display for Raspberry Pi (SKU 19870)**  
[www.elektor.nl/19870](http://www.elektor.nl/19870)



### WEBLINKS

- [1] Display HAT Mini van Pimoroni: <https://www.elektor.nl/pimoroni-display-hat-mini-for-raspberry-pi-zero>
- [2] Pimoroni op GitHub: <https://github.com/pimoroni/displayhatmini-python>
- [3] ClemensAtElektor op GitHub: [https://github.com/ClemensAtElektor/rpi\\_weather\\_display](https://github.com/ClemensAtElektor/rpi_weather_display)





# De WEEF 2022 Awards het goede vieren

**Priscilla Haring-Kuipers (Nederland)**

Op 15 november 2022, tijdens de electronica-beurs in München, vonden bij het tweede jaarlijkse World Ethical Electronics Forum (WEEF) diepgaande voordrachten en debatten plaats over een reeks elektronica-gerelateerde ethische vragen. Vier samen met ons de 4 winnaars van de WEEF Awards.

Tijdens ons tweede World Ethical Electronics Forum (WEEF) in München op 15 november 2022 hebben we vier prijzen uitgereikt aan een aantal zeer indrukwekkende mensen. Uit onze WEEF-index selecteerde onze jury vier winnaars op basis van hun bijdrage aan ethische elektronica in drie opzichten:

- 1) hun mate van invloed
- 2) hun niveau van innovatie
- 3) hun bereidheid om te delen

Onze World Ethical Electronics Forum Index is gevuld met mensen waarvan we bij WEEF weten dat ze zich inzetten voor voor ethiek in elektronica, en met mensen die zijn genomineerd door onze leden en mensen uit de industrie. Als u iemand kent die in deze verzameling van goede mensen zou moeten worden opgenomen (en die kandidaat zou moeten zijn voor de volgende WEEF-award), kunt u hem of haar nomineren door het nominatieformulier op de WEEF-website in te vullen [1].

## WEEF-winnaar Sven Krumpel

Sven Krumpel [2] is de eigenaar van CODICO, dat voor zijn werknemers veel nadruk legt op de balans tussen werk en privé. Zij bieden bijna complete flexibiliteit in werkplek en werktijd en zijn een

goed voorbeeld van werkelijk investeren in werknemers.

“Bedankt voor de nominatie! Ik wist niet dat ik een ethisch persoon was. We zijn een familiebedrijf; we ontwerpen en verbeteren elektronica. We zien hoe belangrijk het is om projecten op het gebied van energie-efficiëntie te ontwikkelen. Ik denk dat ethiek iets van mensen is; jij als persoon bent een voorbeeld. Als bedrijfseigenaar heb je niet alleen werknemers, maar ook mensen die dezelfde waarden koesteren. Onze mensen maken deel uit van de familie. We hebben net een park van 12.000 vierkante meter geopend naast onze kantoren, niet alleen om buiten te kunnen werken maar ook voor vrije tijd. Onze mensen sporten in het park, nodigen hun familie uit en geven verjaardagsfeestjes. We verbouwen groenten en kruiden. We hebben bijen. Ik denk dat we een te comfortabel leven hebben geleid, waarin we te veel grondstoffen hebben verbruikt, ook die van anderen, en we moeten wat teruggeven. Dat is wat we doen, en dat is niets bijzonders.”

## WEEF-winnaar Gopal Kumar Mohoto

Gopal Kumar Mohoto [3] werkt als ingenieur aan de elektrificatie van het transport in Bangladesh. Hij werd gekozen tot Global

No-1 ondernemer bij het ClimateLaunchpad van 2020 en was klimaat-ambassadeur bij het Global Youth Climate Network. Hij heeft gewerkt aan accu-wisselstations voor elektrische tuk-tuks en staat op het punt om de eerste cluster van van benzine tot elektro omgebouwde auto's aan klanten in Bangladesh te leveren. U kunt hier een vraaggesprek met hem lezen [4].

“Ik ben bijzonder vereerd dat ik genomineerd ben, hoewel ik niet weet door wie. De opties voor elektrisch rijden die in het Westen beschikbaar zijn, zijn voor ontwikkelingslanden te duur. Daarom werk ik aan een duurzame en betaalbare optie om het vervoer hier te elektrificeren. We moeten beter zorgen voor de dingen die we al hebben en ze koesteren. We moeten zorgen voor alle materialen die we hebben gedolven. We moeten voor Moeder Aarde zorgen.”

## WEEF-winnaar Frank Stührenberg

Frank Stührenberg [5] is CEO van Phoenix Contact en bestuurslid van de Stiftung KlimaWirtschaft. Hij levert een belangrijke bijdrage aan de All Electric Society, een wereld waarin regeneratief opgewekte elektrische energie alom beschikbaar is als primaire energievorm, in voldoende hoeveelheden voor iedereen.

“Ik weet niet wie ons genomineerd heeft. De prijzen die ons bedrijf krijgt hebben meestal een technische achtergrond, dus ik was verrast door deze prijs. De toon van WEEF past bij ons bedrijf. We zijn een familiebedrijf en hebben nu meer dan 20.000 ‘familieleden’. We moeten eraan werken om ons morele kompas niet te verliezen, en daar werken we actief aan. We hebben onze langetermijn-strategie erop gericht om voor 100% hernieuwbare elektrische energie te gaan. We vonden dat we dit moesten doen, en dus zijn we begonnen.”



Milda Pladaitė (links) ontvangt haar prijs van Beatriz Souza van Elektor (foto: Messe München).



Sven Krumpel (rechts) ontvangt een prijs uit handen van Johann Wiesböck van ELEKTRONIKPRAXIS (foto: Messe München).



Gopal Kumar Mohoto.



Frank Stührenberg (rechts, met de CEO van Messe München Reinhard Pfeiffer; foto: Messe München).

We hebben energie nodig om de wereld te ontwikkelen, en wat heeft het voor zin om een bedrijf van 10 miljard euro te zijn als we geen wereld hebben waarin we willen leven? Bedankt dat u deze ontwikkeling en dit initiatief steunt.”

### WEEF-winnaar Milda Pladaitė

Milda Pladaitė [6] is een civiel ingenieur uit Litouwen en maakt deel uit van het Young Engineers Future Leaders Committee van de World Federation of Engineering Organizations (WFEO). Zij heeft bij de WFEO een werkgroep opgericht over Sustainable Development Goal 13 met als doel bij te dragen aan de duurzame ontwikkeling van landen door het werk van jonge WFEO-ingenieurs in te zetten voor klimaat-acties. Momenteel werkt ze aan een sociaal initiatief om ingenieurs te helpen ondernemen door hen in contact te brengen met de industrie, sociale investeerders en academische organisaties.

“Ik heb veel mensen genomineerd voor prijzen, maar ik weet niet wie mij hiervoor heeft genomineerd. Een vriendin zei toen

ze hoorde dat ik hiervoor genomineerd was: ‘Ik wist niet dat je een specialist in ethiek was!’ Dat betekent natuurlijk dat iedereen zich met ethiek kan bezighouden. Dit forum is een geweldig initiatief en ik weet zeker dat het in de toekomst nog zal groeien. Dit is een unieke gelegenheid om over ethiek in de elektronica te spreken.”

We zien aan de verrassing van onze WEEF-prijswinnaars dat het niet altijd duidelijk is wat ethisch is in de elektronica. We zijn misschien terughoudend om als industrie over ethiek te praten, terwijl dat niet nodig is. Het doel van ons World Ethical Electronics Forum is een platform te bieden om te bespreken wat we doen en waarom we dat doen, om verhalen en handelwijzen met elkaar te delen en om samen het goede te vieren. ◀

220650-03

### Over de auteur

Priscilla Haring-Kuipers schrijft over technologie vanuit sociaal-wetenschappelijk perspectief. Ze is vooral geïnteresseerd in technologie die het goede in de mensheid ondersteunt en gelooft heilig in effect-onderzoek. Ze heeft een MSc in mediapsychologie en is een van de mensen achter This Is Not Rocket Science.

### Het World Ethical Electronics Forum

Het World Ethical Electronics Forum (WEEF) inspireert vernieuwers wereldwijd met open discussies en publicaties over ethiek en duurzame ontwikkelingsdoelen. Bezoek [worldethicalelectronicsforum.com](https://worldethicalelectronicsforum.com) voor inspiratie en om deel te nemen.

### WEBLINKS

- [1] WEEF Index nominatieformulier: <https://worldethicalelectronicsforum.com/nomination-form>
- [2] WEEF, Sven Krumpel: [https://worldethicalelectronicsforum.com/index/184374/Sven\\_\\_Krumpel](https://worldethicalelectronicsforum.com/index/184374/Sven__Krumpel)
- [3] WEEF, Gopal Kumar Mohoto: [https://worldethicalelectronicsforum.com/index/182949/Gopal\\_Kumar\\_Mohoto](https://worldethicalelectronicsforum.com/index/182949/Gopal_Kumar_Mohoto)
- [4] P. Haring-Kuipers, “Retrofitting and Upcycling: Interview with Gopal Kumar Mohoto,” ElektorMagazine.com, 2022: <https://www.elektormagazine.com/gopal-interview-for-next-eia>
- [5] WEEF, Frank Stührenberg: [https://worldethicalelectronicsforum.com/index/184378/Frank\\_\\_St%C3%BChrenberg](https://worldethicalelectronicsforum.com/index/184378/Frank__St%C3%BChrenberg)
- [6] WEEF, Milda Pladaitė: [https://worldethicalelectronicsforum.com/index/182955/Milda\\_\\_Pladait%C4%97](https://worldethicalelectronicsforum.com/index/182955/Milda__Pladait%C4%97)

# 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.nl](mailto:sale@elektor.nl)).  
Ons motto: **nooit duur, altijd verrassend!**

## Raspberry Pi High Quality Camera Module (M12 Mount)

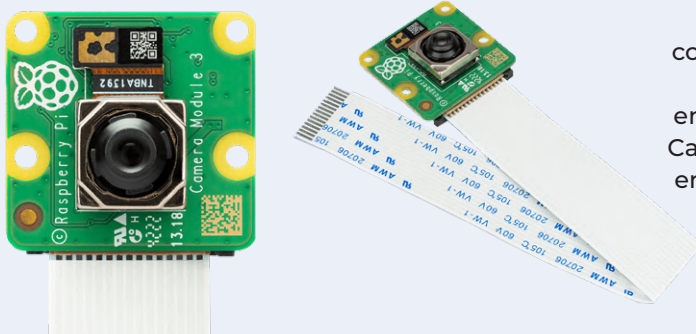
De Raspberry Pi High Quality Camera biedt een resolutie van 12 megapixels en een sensor met een diagonaal van 7,9 mm voor indrukwekkende prestaties bij weinig licht.

**Prijs: € 59,95**



 [www.elektor.nl/20366](http://www.elektor.nl/20366)

## Raspberry Pi Camera Module 3



De Raspberry Pi Camera Module 3 is een compacte camera van Raspberry Pi. Hij heeft een IMX708 12-megapixelsensor met HDR, en beschikt over autofocus met fasedetectie. Camera Module 3 is verkrijgbaar in standaard en groothoek varianten, beide met of zonder infrarood filter.

**Prijs: € 32,95**

 [www.elektor.nl/20362](http://www.elektor.nl/20362)





## YDLIDAR OS30A 3D-dieptecamera

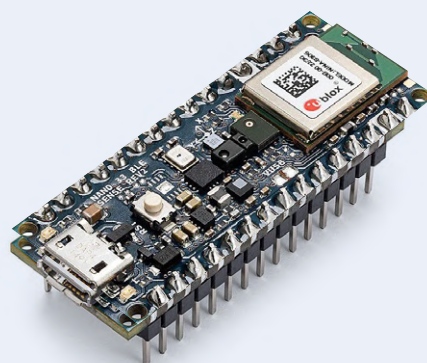


Prijs: € 139,95

**Ledenprijs: € 125,96**

[www.elektor.nl/20350](http://www.elektor.nl/20350)

## Arduino Nano 33 BLE Sense Rev2 met headers



Prijs: € 54,95

**Ledenprijs: € 49,46**

[www.elektor.nl/20404](http://www.elektor.nl/20404)

## Arduino Student Kit

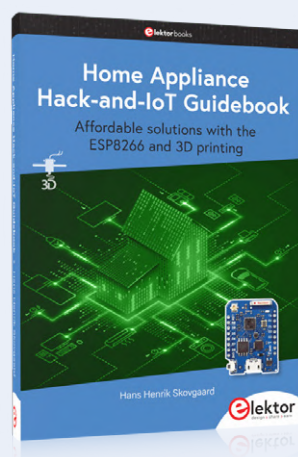


Prijs: € 74,95

**Ledenprijs: € 67,46**

[www.elektor.nl/20329](http://www.elektor.nl/20329)

## Home Appliance Hack-and-IoT Guidebook



+  
FREE ESP8266  
Board



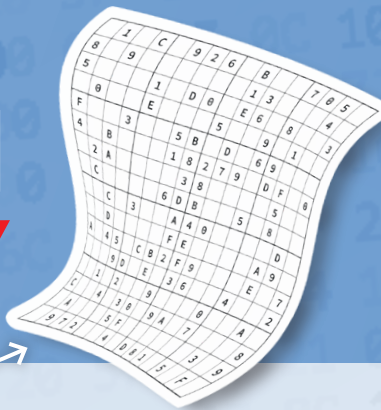
Prijs: ~~€ 54,90~~

**Actieprijs: € 39,95**

[www.elektor.nl/20370](http://www.elektor.nl/20370)

# 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 **15 april 2023** naar:

[hexadoku@elektor.nl](mailto:hexadoku@elektor.nl)

## DE PRIJSWINNAARS

De juiste oplossing van de Hexadoku uit het januari/februari-nummer 2023 is: **AEF1C**.

Oplossingen die ons vóór 15 december 2022 hebben bereikt, deden mee aan de trekking van vijf Elektor-waardebonnen.

De winnaars zijn bekendgemaakt op [www.elektormagazine.com/hexadoku](http://www.elektormagazine.com/hexadoku).

Allemaal van harte gefeliciteerd!

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|   | 1 | 3 | F |   |   |   |   |   |   |   |   | E | 2 | 7 |   |
| 0 | 6 | 5 |   |   |   |   |   |   |   |   |   |   | 8 | C | D |
|   | 8 |   |   |   | 2 |   |   |   |   |   | E |   |   |   | 6 |
| 9 | B | D | E | 4 |   | 7 |   |   |   | C |   | 2 | F | 3 | A |
| F | 7 | E | 8 |   |   | D |   |   |   | B |   |   | A | 6 | 5 |
|   | 9 | 0 |   | 6 | 1 |   |   |   |   |   | 3 | F |   | 4 | B |
|   | D | 4 |   |   | F |   |   |   |   |   | 9 |   |   | 7 | 8 |
| 1 |   | 6 | 3 | 2 |   |   |   | B | D |   |   | 8 | 9 | E | F |
|   |   |   | 6 |   | 8 | 0 |   |   | D | 2 |   | 5 |   |   |   |
|   | E | A |   |   | D | 9 | 2 | 8 | 5 | F |   |   | 0 | 1 |   |
| 5 |   |   |   | 1 |   |   | E | 4 |   |   | 9 |   |   |   | 8 |
|   | 0 |   | 9 |   | 5 |   | 4 | A |   | 1 |   | C |   | F |   |
| D |   |   |   |   | 1 |   |   | 8 |   |   |   |   |   |   | E |
|   |   | 9 |   | 8 | 7 |   |   |   |   | D | A |   | 5 |   |   |
|   | A |   | 1 |   | 9 | 3 |   |   | 2 | 7 |   | 8 |   | D |   |
| 7 | 5 |   |   |   |   | 4 | F | E | 3 |   |   |   |   | 9 | 2 |

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| E | 5 | 2 | A | 4 | B | 8 | C | 6 | 3 | F | 0 | D | 7 | 9 | 1 |
| 0 | 3 | B | 4 | 6 | F | 2 | 7 | D | 9 | 1 | A | E | 5 | 8 | C |
| C | F | 1 | 7 | D | E | 9 | A | B | 4 | 5 | 8 | 2 | 0 | 3 | 6 |
| 6 | D | 8 | 9 | 0 | 1 | 3 | 5 | C | 7 | E | 2 | B | A | F | 4 |
| 1 | 8 | F | 3 | 5 | 7 | 4 | 0 | E | B | 2 | 6 | 9 | D | C | A |
| B | 4 | C | D | 3 | 2 | 6 | 9 | 8 | 0 | A | 5 | F | E | 1 | 7 |
| 5 | A | E | 2 | 8 | C | D | 1 | 7 | F | 9 | 4 | 6 | B | 0 | 3 |
| 7 | 0 | 9 | 6 | B | A | E | F | 1 | C | 3 | D | 4 | 2 | 5 | 8 |
| F | 9 | 7 | B | E | 6 | 5 | 4 | 0 | 8 | D | C | 1 | 3 | A | 2 |
| 3 | E | 5 | C | 7 | 8 | A | 2 | F | 1 | B | 9 | 0 | 4 | 6 | D |
| D | 2 | 0 | 8 | 9 | 3 | 1 | B | 4 | A | 6 | E | C | F | 7 | 5 |
| 4 | 6 | A | 1 | C | 0 | F | D | 5 | 2 | 7 | 3 | 8 | 9 | B | E |
| 8 | C | 3 | F | 1 | 5 | B | E | 9 | D | 4 | 7 | A | 6 | 2 | 0 |
| 9 | 1 | 4 | 0 | 2 | D | 7 | 6 | A | 5 | C | B | 3 | 8 | E | F |
| A | B | 6 | 5 | F | 4 | 0 | 3 | 2 | E | 8 | 1 | 7 | C | D | 9 |
| 2 | 7 | D | E | A | 9 | C | 8 | 3 | 6 | 0 | F | 5 | 1 | 4 | B |

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!
- ✓ 8x Elektor Magazine (Print)
- ✓ 8x digitaal (PDF)
- ✓ 10% korting in onze webshop, en exclusieve aanbiedingen
- ✓ Toegang tot meer dan 5000 Gerberfiles



Ook verkrijgbaar

Het digitale  
lidmaatschap!



- ✓ Toegang tot ons web-archief
- ✓ 10% korting in onze webshop
- ✓ 8x Elektor Magazine (PDF)
- ✓ Toegang tot meer dan 5000 Gerberfiles



[www.elektormagazine.nl/Abonnement](http://www.elektormagazine.nl/Abonnement)



# Levert meer

De breedste selectie halfgeleiders en elektronische componenten op voorraad, klaar voor verzending™

---

