





Aan de slag met Zephyr RTOS krachtig – maar lastig...



FPGA's voor beginners van MCU naar FPGA-programmering







Spanningsreferentie met Arduino Pro Mini lineariseer en kalibreer analoge ingangen

ONS ASSORTIMENT: VAN TECHNICI VOOR TECHNICI

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

Voor u alleen het beste - van meer dan 1.500 merkfabrikanten

Onze productmanagers zijn al vele jaren bij reichelt actief en kennen de eisen van onze klanten. Zij stellen een breed scala aan kwaliteitsproducten samen, optimaal afgestemd op de behoefte bij onderzoek en ontwikkeling, onderhoud, IT-infrastructuur, kleine serieproducties en makers.

Robotica en machine learning

De combinatie van robotica, kunstmatige intelligentie en machine learning stelt robots in staat om complexere taken autonoom uit te voeren en zich aan te passen aan nieuwe uitdagingen. Vind de juiste technologie in ons continu groeiende assortiment.

Arduino Pro Nicla Vision bewaken - analyseren - herkennen

KI-cameramodule met Dual ARM CortexR M7 & M4

Nicla Vision is een gebruiksklare, stand-alone camera voor de analyse en verwerking van beelden on the edge en is geschikt voor Asset Tracking, objectherkenning en voorspellend onderhoud.

De nieuwe generatie - Go2

Nu standaard met 4D ultra-wide LIDAR

Duik in de fascinerende wereld van de robotica. Deze geavanceerde vierbenige robot combineert de modernste technologie met een elegante en wendbare constructie en brengt de toekomst van de robotica direct naar u toe.

- 30% verbeterd motorvermogen
- 150% langere levensduur/batterijcapaciteit
- ISS2.0 (vanaf Pro) intelligent zijdelings geleidingssysteem





www.reichelt.com Bestelservice: +31 85 208 62 94

Preichelt

elektronik – The best part of your project

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.lt./AV-NL of op aanvraag) zijn van toepassing. Afbeeldingen kunnen afwijken. Drukfouten, vergissingen en prijswijzigingen voorbehouden. reichelt elektronik GmbH, Elektronikring 1, D-26452 Sande, Tel.: +31 85 208 62 94

COLOFON

64º jaargang nr. 686 maart/april 2024 ISSN 2590-0765

Elektor verschijnt acht keer per jaar en is een uitgave van Elektor International Media B.V.

Postbus 11, 6114 ZG Susteren (Nederland) Tel.: +31 (0)46 4389444

www.elektor.nl | www.elektormagazine.nl

Voor al uw vragen: service@elektor.nl

Lid worden: www.elektormagazine.nl/abo

Advertenties

Raoul Morreau Tel. +31 (0)6 4403 9907 raoul.morreau@elektor.com www.elektormagazine.nl/adverteren

Auteursrecht

© Elektor International Media B.V. - 2024

Niets uit deze uitgave mag verveelvoudigd en/ of openbaar gemaakt worden door middel van druk, fotokopie, microfilm of op welke wijze dan ook, zonder voorafgaande schriftelijke toestemming van de uitgever. De auteursrechtelijke bescherming van Elektor strekt zich mede uit tot de illustraties met inbegrip van de printed circuits, evenals de ontwerpen daarvoor. In verband met artikel 30 van de Rijksoctrooiwet mogen 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.

Druk

Senefelder Misset, Mercuriusstraat 35, 7006 RK Doetinchem (Nederland)

Distributie

Betapress, Nederland - AMP, België



VOORWOORD

Jens Nickel

Hoofdredacteur Elektor Magazine



Stel jezelf doelen!

Al langer geleden schreef ik in een artikel dat het één specifiek doel was dat me ertoe had gebracht een ambitieuze programmeur te worden. Destijds moest ik voor ons interne artikelbeheer vertrouwd raken met VBA voor Excel, MS Access, C#, HTML + JavaScript en nog veel meer. Ik heb ongelooflijk veel geleerd, wat ook nuttig was voor mijn werk als redacteur en verschillende andere projecten. Ik zou nooit zover zijn gekomen door alleen tutorials te bestuderen en een paar kleine oefenprojecten uit te voeren. Een paar professionals die mijn systeem evalueerden waren ook verbaasd. Ze hadden zelf vaak overwogen om hun eigen redactioneel systeem te maken, maar hadden het nooit voltooid.

En dit principe geldt ook voor hardware. Zoals velen van jullie weten, ben ik via een omweg bij elektronica terechtgekomen. Ik vond het leuk om te knutselen en te programmeren, maar het printontwerp liet ik altijd aan mijn collega's over. Onlangs begon ik samen met een vriend aan een nieuw, groter elektronicaproject - afstandsbediening (en meten op afstand) van gemodificeerde audioversterkers. Plotseling 'moest' ik vertrouwd raken met KiCad, verschillende connectoren, programmeren met touch-display en verschillende opties van GitHub. En dat is zeker niet het einde van het verhaal – ik zal hier binnenkort over berichten op elektor-labs.com. Wat ik hiermee bedoel? Stel jezelf doelen en durf dingen aan te pakken die op het eerste gezicht (veel te) ontmoedigend lijken. Je zult gedreven worden door de ambitie om je idee werkelijkheid te laten worden en dit zal je genoeg enthousiasme geven om je op nieuwe terreinen te wagen. Rond je project zo af en toe ook af, hoe ruw het er ook uitziet en hoe ongelijkmatig de voortgang ook is. Het gevoel iets gepresteerd te hebben zal je motiveren om alles te willen verbeteren. En juist in dit nummer kun je meteen aan de slag met AI-ondersteunde beeldverwerking. Ben je dat al lang van plan? Geweldig! Het is niet zo moeilijk om je eigen presentabele toepassing te ontwikkelen (pagina's 6 en 86). Geniet van deze deur naar een wereld vol nieuwe ideeën die we voor je op een kier hebben gezet!



Deel met Elektor!

Je elektronica-expertise is welkom! Heb je een voorstel voor een artikel of een elektronica-tutorial op video, of een idee voor een boek? Bekijk onze gids voor auteurs en inzendingen op

www.elektormagazine.com/submissions



Elektor Labs: ideeën & projecten Het Elektor Labs-platform is voor iedereen toegankelijk. Post elektronica-ideeën en -projecten, bespreek technische uitdagingen en werk samen met anderen QО

www.elektormagazine.com/labs

Ons team

Internationaal hoofdredacteur: Jens Nickel | Content Director: C.J. Abate | Internationale redactie: Asma Adhimi, Roberto Armani, Eric Bogers, Jan Buiting, Stuart Cording, Rolf Gerstendorf (RG), Ton Giesberts, Ouafae Hassani, Hedwig Hennekens, Saad Imtiaz, Alina Neacsu, Dr. Thomas Scherer, Jean-Francois Simon, Clemens Valens, Brian Tristam Williams | Vaste medewerkers: David Ashton, Tam Hanna, Ilse Joostens, Prof. Dr. Martin Ossmann, Alfred Rosenkränzer | Vormgeving & Layout: Harmen Heida, Sylvia Sopamena, Patrick Wielders | Directeur: Erik Jansen | Technische vragen: redactie@elektor.com



IN DIT NUMMER



Rubrieken

- 3 Voorwoord
- 29 STM32 Wireless Innovation Design Contest 2024 een update
- 52 2024: een Al-odyssee een eerste verkenning van TensorFlow
- 56 Lezersproject in het kort 262.144 manieren om Game of Life te spelen
- 62 Uit het leven gegrepen de Chinese draak
- 77 Alle begin... ...gaat verder met de opamp
- 84 Vreemde onderdelen piëzo-elektrische componenten
- 112 Project 2.0 correcties, updates en brieven van lezers

Achtergrond en info

- 22 FPGA's voor beginners van MCU naar FPGA-programmering
- **30 Bluetooth LE met MAUI** besturingsapps voor Android en co.
- 64 Draaien met die (DC-borstel)motor! voorbeeldprojecten uit de Elektor Motor Control Development Bundle
- 80 Aanbevolen ESP-bibliotheken
- 95 ESP32 Terminal handheld met aanraakscherm
- 98 Aan de slag met Zephyr RTOS krachtig – maar lastig...

Industry

- 92 Oplossingen voor de lastigste uitdagingen bij embedded ontwikkeling
- **107 Ethiek in actie** een vraaggesprek met Alexander Gerfer, CTO van Würth Elektronik eiSos

CaptureCount

objectdetector en -teller op de Raspberry Pi 5

4 maart/april 2024 www.elektormagazine.nl

ESP32/RS-232-adapter draadloze verbinding voor klassieke meetinstrumenten

Bluetooth LE met MAUI

besturingsapps voorAndroid en co.

Projecten

- 6 CaptureCount objectdetector en -teller op de Raspberry Pi 5
- 14 Spanningsreferentie met Arduino Pro Mini lineariseer en kalibreer analoge ingangen
- **38 Port Expander breakout-board** meer I/O's op je development board
- 44 Al-specialist machine learning met de Jetson Nano
- 70 ESP32/RS-232-adapter draadloze verbinding voor klassieke meetinstrumenten
- 86 Slimme objectteller eenvoudige beeldherkenning met Edge Impulse

Binnenkort

Elektor mei/juni 2024

We hebben weer een opwindende mix voor u samengesteld van projecten, schakelingen, principes en tips en trucs van en voor engineers en makers. De focus ligt op testen en meten.

- > Personendetector
- > ECG-monitor
- > Versterking/fase-analyzer met geluidsinterface
- > Reparatie van elektronische apparatuur
- > DDS-signaalgenerator
- > In-circuit LC-meter
- > Interface voor vermogensmeter (5 A/60 V)
- > Actieve stroboscoop voor draaitafels

... en nog veel meer!

Aankondigingen onder voorbehoud. Elektor mei/juni 2024 verschijnt omstreeks 15 mei 2024.



CaptureCount

objectdetector en -teller op de Raspberry Pi 5

Saad Imtiaz (Elektor)

Machine vision is een fascinerende technologie die het mogelijk maakt om een breed scala aan objecten in onze omgeving te herkennen en te classificeren. Dit artikel presenteert een intrigerend Raspberry Piproject dat gebruik maakt van het YOLOobjectdetectiemodel. Het project is in staat talloze objecten te identificeren, variërend van gewone zoals auto's en fietsen tot allerhande dieren zoals katten, honden en vogels.

Het project begon met een duidelijk doel: de bouw van een veelzijdig systeem dat objecten herkent en vervolgens telt hoeveel objecten van elke categorie gedetecteerd worden. De Raspberry Pi was een voor de hand liggende keuze vanwege zijn mogelijkheden en ondersteuning in AI-projecten. Aangezien dit een van mijn eerste computer visionprojecten was, kostte het me wat tijd om Python te leren en de juiste tools en bibliotheken te gebruiken om dit project te ontwikkelen. Omdat er zoveel projecten, bronnen [1][2] en onze geliefde ChatGPT zijn, was het niet moeilijk om te leren hoe zo'n project ontwikkeld kan worden.

Objectdetectie

In de zoektocht naar het maken van dit project begon de reis met het vinden van het juiste objectdetectie-model, bij voorkeur een model dat een grote verscheidenheid aan objecten kan detecteren en herkennen met minimale trainingstijd. Maar voordat we op zoek gaan naar zo'n objectdetectie-model, doen we eerst een stapje terug en bespreken we hoe objectdetectie werkt.

Objectdetectie is een technologie waarmee computers objecten in een afbeelding of video kunnen identificeren en lokaliseren. In tegenstelling tot beeldherkenning, dat je vertelt wat er in een afbeelding staat, gaat objectdetectie verder door precies aan te geven waar die objecten zich bevinden en ze te omlijnen. Dit wordt meestal bereikt via twee belangrijke processen:

> Objectlokalisatie: bepaalt de locatie van een enkel object in een afbeelding. Het model levert is de coördinaten van het kader rond het object.



Figuur 1. CaptureCount draait op een Raspberry Pi 5 met de Raspberry Pi Camera Module 3 (wide).

> Objectclassificatie: identificeert wat het object in het kader is uit een set bekende categorieën.

Geavanceerde objectdetectie-modellen integreren deze twee stappen, waarbij een afbeelding efficiënt wordt verwerkt om zowel de locatie als de classificatie van meerdere objecten in de opname te verkrijgen.

Verschillende objectdetectie-modellen

Bij de zoektocht naar het ideale objectdetectie-model werden verschillende opties geëvalueerd, elk met zijn sterke en zwakkere punten. Modellen zoals R-CNN en de afgeleiden daarvan boden een hoge nauwkeurigheid maar met een lage verwerkingssnelheid, ongeschikt voor realtime-toepassingen. Single Shot Multibox Detector (SSD) bood een sneller alternatief, maar met een compromis bij de nauwkeurigheid. Mask R-CNN bood nauwkeurige detectie, maar was te complex voor wat we nodig hebben. Uiteindelijk werd YOLO (*You Only Look Once*) geselecteerd vanwege de optimale balans tussen snelheid, efficiëntie en acceptabele nauwkeurigheid. De mogelijkheid om opnames in realtime te verwerken en de eenvoud, in combinatie met de sterke ondersteuning van de Al-community, maakten YOLO de ideale keuze voor wat we met dit project wilden bereiken.

De selectie van het YOLO-model was cruciaal voor dit project.

YOLO van Joseph Redmon [3] staat bekend om zijn vermogen om objecten in realtime te detecteren – een essentiële eigenschap voor elk veelzijdig detectiesysteem. Wat me aantrok in YOLO was de unieke benadering van objectdetectie.

In tegenstelling tot traditionele methoden die een afbeelding in meerdere stappen verwerken, doet YOLO dit in één keer. Dit versnelt het proces en verbetert het vermogen van het model om te vanuit zijn training te generaliseren, waardoor het effectiever is in het detecteren van objecten in verschillende en onvoorspelbare omgevingen in de echte wereld. Deze mogelijkheid was cruciaal voor dit project dat als doel had een grote verscheidenheid aan objecten te herkennen.

Integratie en installatie van hardware en software

Voor het project waren de Raspberry Pi 5, bekend om zijn uitstekende verwerkingskracht, en een compatibele cameramodule nodig. Het integratieproces bestond uit de inrichting van het Raspberry Pi-besturingssysteem, het aansluiten van de camera en het installeren van YOLO. Om met de Raspberry Pi aan de slag te kunnen gaan, moet er eerst een besturingssysteem op de microSD-kaart worden geïnstalleerd. Dat kan eenvoudig gedaan worden door de Raspberry Pi Imager te downloaden van de officiële website van Raspberry Pi [4]. Start vervolgens de imager, selecteer het juiste device, kies het nieuwste Raspberry Pi OS (64 bit), selecteer de microSD-kaart en klik op Next. Na een paar minuten wordt het Raspberry Pi OS op de microSD-kaart geïnstalleerd. Plaats daarna de SD-kaart in de microSD-kaartsleuf van de Raspberry Pi en zet hem aan. Je hebt ook een monitor, een mini HDMI/HDMI-kabel, een toetsenbord en een muis nodig om voor de eerste keer met de Raspberry Pi te kunnen werken. Nadat je VNC Server of SSH hebt ingesteld, kun je de Raspberry Pi op afstand bedienen met je computer via een WiFi- of Ethernetverbinding.

Nu is het tijd om de vereiste bibliotheken voor dit project te installeren. Dat is eenvoudig; het wordt gedaan met de Terminal. Voordat we nieuwe bibliotheken installeren, is het aan te raden om de systeem-packages te actualiseren. Dit doen we met dit commando:

sudo apt-get update && sudo apt-get upgrade

Dit kan even duren; daarna kunnen de bibliotheken die we nodig hebben voor het project geïnstalleerd worden, wat gedaan wordt met de volgende commando's:

> Image I/O-packages installeren

sudo apt-get install libjpeg-dev libtiff5-dev libjasperdev libpng12-dev

> Video I/O-packages en GTK development-bibliotheek installeren

sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev sudo apt-get install libxvidcore-dev libx264-dev sudo apt-get install libgtk2.0-dev

> Extra afhankelijkheden voor OpenCV

```
sudo apt-get install libatlas-base-dev gfortran
```

> Pip en pipx (package management tool) installeren

```
sudo apt-get install python3-pip
sudo apt install pipx -y
```

> Numpy, Pandas en OpenCV installeren

```
pip install numpy
pip install pandas
sudo apt install python3-opencv
```

Na het installeren van alle bibliotheken is het belangrijk om te zorgen voor naadloze communicatie tussen de hardware en de software, zodat de Raspberry Pi de live camerafeeds effectief kan verwerken. Om alles compact te houden, werd de Raspberry Pi Camera aangesloten op de MIPI-camerapoort van de Raspberry Pi 5. Zodra alles is geïnstalleerd, kunnen we ons met de code van het project gaan bezighouden.

Een duik in de code

Het hart van het project wordt gevormd door de Python-code, die beschikbaar is op GitHub [4]. Het script begint met het importeren van essentiële bibliotheken zoals OpenCV voor beeldverwerking, Pandas voor dataverwerking en Numpy voor numerieke bewerkingen.

```
import cv2
import pandas as pd
import numpy as np
import subprocess
import os
from datetime import datetime
```

In het begin importeert het script essentiële bibliotheken. *cv2* (OpenCV) is cruciaal voor beeldverwerkingstaken. *pandas* en *numpy* nemen respectievelijk datamanipulatie en numerieke berekeningen voor hun rekening. *subprocess* en *os* zijn standaard Python-bibliotheken voor interactie met het systeem, zoals het uitvoeren van externe commando's en het afhandelen van bestandspaden. *datetime* wordt gebruikt voor tijdstempels.

```
model = './yolo/yolov3.weights'
config = './yolo/yolov3.cfg'
net = cv2.dnn.readNetFromDarknet(config, model)
```

Het script specificeert de paden naar YOLO's voorgetrainde gewichtsfactoren en configuratiebestand en laadt deze vervolgens met behulp van OpenCV's deep neural network (DNN)-module. Deze stap initialiseert het YOLO-model voor objectdetectie.

```
classes = []
with open("./yolo/coco.names", "r") as f:
    classes = [line.strip() for line in f.readlines()]
```

Het bestand *coco.names*, met namen van objecten die YOLO kan detecteren, wordt regel voor regel gelezen om een lijst met klassenamen te maken. Er worden verschillende functies gedefinieerd om



het detectieproces te stroomlijnen. Werpen we in **listing 1** een blik op de initiële parameters en functies voor de hoofdlus.

De functie get_output_layers(net) extraheert de namen van de uitvoerlagen van het YOLO-netwerk. De architectuur van YOLO heeft meerdere uitvoerlagen en deze functie helpt ze te identificeren voor het verwerken van de detecties.

draw_bounding_box(...) tekent rechthoeken (bounding boxes) rond gedetecteerde objecten en labelt ze met de naam van het object en de betrouwbaarheidsscore.

calculate_centroid(x, y, w, h) berekent het middelpunt van de bounding box van het gedetecteerde object, een cruciale stap voor het volgen van objecten in verschillende frames.

def capture_image(image_path):

subprocess.run(["libcamera-still", "-o", image_path, "--width", "1920", "--height", "1080", "-n", "-t", "1"], stdout=subprocess.DEVNULL, stderr=subprocess.STDOUT)

Deze functie gebruikt de subprocess-module om *libcamera-still* uit te voeren, een commandoregel-tool op Raspberry Pi OS die een opname van de camera vastlegt en opslaat op het aangegeven pad. De resolutie is ingesteld op 1920x1080. Een lagere resolutie kan ook worden gebruikt om het CPU- en GPU-gebruik te minimaliseren. Hiervoor kan het bovenstaande codefragment als volgt worden aangepast:

def capture_image(image_path):

subprocess.run(["libcamera-still", "-o", image_path, "--width", "1280", "--height", "720", "-n", "-t", "1"], stdout=subprocess.DEVNULL, stderr=subprocess.STDOUT)

Voordat de hoofddetectielus wordt gestart, initialiseert het script een Pandas DataFrame om detecties te voorzien van tijdstempels en een woordenboek om het aantal van elk objecttype bij te houden. centroid_ tracking wordt gebruikt om de beweging van objecten te volgen.

```
data_frame = pd.DataFrame(columns=['Timestamp', 'Type',
'Count'])
object_counts = {cls: 0 for cls in classes}
centroid_tracking = {}
```

Het volgende deel van de code (zie **listing 2**) controleert of het gedetecteerde object nieuw is of al eerder is geïdentificeerd. Het gebruikt centroid_tracking om dit te bepalen. Als een object als nieuw wordt geïdentificeerd, wordt het aantal voor dat objecttype verhoogd, de detectietijd geregistreerd en een Pandas DataFrame bijgewerkt. Dit DataFrame kan later worden geëxporteerd als een .csv-bestand voor verdere analyse.

Setup van de hoofddetectielus

De hoofddetectielus van de Raspberry Pi 5 en het YOLO-systeem is een onmisbaar onderdeel van de functionaliteit. Het begint met het vastleggen van een opname van de Raspberry Pi-camera en deze te converteren naar een formaat dat geschikt is voor YOLO. Het systeem verwerkt deze afbeelding vervolgens door YOLO en extraheert essentiële informatie zoals klasse-ID's, betrouwbaarheidsscores en bounding box-coördinaten. Om de detecties te verfijnen wordt Non-Maximum Suppression (NMS) toegepast, waarbij minder zekere en overlappende detecties worden uitgefilterd. Het systeem tekent vervolgens kaders rond de gedetecteerde objecten en slaat de afbeeldingen op als er nieuwe objecten zijn geïdentificeerd. Gedurende dit proces werkt het systeem de trackingen registratie-datastructuren bij, waardoor een continu overzicht van de detecties wordt bijgehouden. Deze lus staat centraal in de realtime objectdetectie-mogelijkheden van de Raspberry Pi 5 en het YOLOsysteem. Zie **listing 3** voor de hoofdlus van de volledige code.

Na het verwerken van de detecties compileert het script de data in een DataFrame en slaat het op als een .csv-bestand. Dit biedt een gedetailleerd overzicht van elke detectie-gebeurtenis.

CaptureCount in actie

Toen de CaptureCount in gebruik werd genomen, bleek de detectie van objecten goed te zijn, zij het met enkele intrigerende eigenaardigheden. De Raspberry Pi was gemonteerd op het statief van de camera en het zicht van de camera was naar buiten door het raam, zoals in **figuur 2**. Zoals je ziet, woon ik in een buitenwijk van de stad, en de enige dingen die de CaptureCount detecteert zijn auto's, vrachtwagens, mensen en motorfietsen. Het zou veel interessanter zijn geweest als het een landelijke omgeving was geweest, want dan had ik er wilde dieren mee kunnen detecteren.

Het merendeel van de detecties werd gekenmerkt door een bewonderenswaardige nauwkeurigheid van 80% bij het identificeren



Figuur 2. Dit ziet de camera.



Figuur 3:. Twee auto's zijn gedetecteerd door CaptureCount.



Figuur 4. Foto van de auto's die door het systeem zijn gedetecteerd.



Figuur 5. Door het systeem gedetecteerde vrachtwagen.

en categoriseren van verschillende objecten. Dit niveau van nauwkeurigheid was vooral opmerkelijk gezien de plaatsing van de camera, die zich relatief ver van de objecten bevond, in combinatie met het gebruik van een groothoeklens. In **figuur 3**, **figuur 4** en **figuur 5** worden de opnamen van de gedetecteerde objecten getoond. In **figuur 6** en **figuur 7** wordt het volledige logboek als .csv weergegeven: wanneer welk object werd gedetecteerd en hoeveel objecten van dezelfde categorie werden gedetecteerd tijdens verschillende tijdsintervallen.

Uitdagingen en opmerkelijke zaken

Ondanks de successen stond het systeem voor een aantal interessante uitdagingen:

- Selectieve detectie in dynamische omgevingen: het is een paar maal gebeurd het systeem een persoon op een motorfiets identificeerde, maar de motorfiets zelf niet. Deze selectieve detectie benadrukte een uitdaging bij het onderscheiden van nauw verbonden objecten, vooral wanneer ze in beweging zijn.
- Incidentele verkeerde classificaties: in zeldzame gevallen werden voetgangers ten onrechte geclassificeerd als fietsers. Deze misclassificatie wijst op de complexiteit van objectclassificatie, vooral wanneer objecten vergelijkbare ruimtelijke kenmerken hebben.

Hoe verder met CaptureCount?

Samengevat liet dit project, waarbij gebruik werd gemaakt van de Raspberry Pi 5 en YOLO, opmerkelijke prestaties zien, wat deuren opent voor diverse verbeteringen en toepassingen. De incidentele afwijkingen in de detectie en verkeerde classificaties van het systeem benadrukken niet alleen gebieden voor toekomstige verbeteringen, maar openen ook mogelijkheden voor verder onderzoek.

Toekomstige verbeteringen zijn bijvoorbeeld de integratie van servomotoren voor het gericht volgen van objecten en IoT-connectiviteit

۲		elektor			5
File	Edit View Insert Fo	ormat	Styles	Sheet	D
	• 🖻 • 🗟 • 🗋 🖷	۵.	X 🗈	-	Ê
Libe	eration Sans	-	10 pt	•	В
A1		fx Σ •	= Tim	nestamp)
	А	В	C	D	
1	Timestamp	Туре	Count		-
2	2023-12-09 12:32:13.076846	car	1		
3	2023-12-09 12:32:15.389343	car	1		
4	2023-12-09 12:32:22.641547	car	1		
5	2023-12-09 12:32:51.806253	car	1		
7	2023-12-09 12:33:15.493817	car	1		
0	2023-12-09 12:33:37:409009	car	1		
0	2023-12-09 12:33:49.290102	car	1		
10	2023-12-09 12:33:53 289954	car	1		-
11	2023-12-09 12:33:55.269961	car	1		_
12	2023-12-09 12:33:57.246952	car	1		
13	2023-12-09 12:34:01.136601	car	1		_
14	2023-12-09 12:34:18.922022	person	1		
15	2023-12-09 12:34:22.908511	person	1		
16	2023-12-09 12:34:32.785654	person	1		
17	2023-12-09 12:34:54.565159	truck	1		

Figuur 6. Screenshot van het .csv-bestand dat de gedetecteerde objecten met de bijbehorende tijdstempels toont.

8		>_	elektor		L.	
File	Edit View	Insert Fo	ormat S	tyles S	heet D	
	• 🗁 • 🔚 •	• 🗋 🖨	a X		6 - 6	
Lib	Liberation Sans 🔹 10 pt 💌 🖪					
A1		▼ J	x Σ - =	Туре		
	A	В	С	[D	
1	Туре	Total Count				
2	person	3				
3	bicycle	0				
4	car	12				
5	motorbike	0				
6	aeroplane	0				
7	bus	0				
8	train	0				
9	truck	1				
10	boat	0				
11	traffic light	0				
12	fire hydrant	0				
13	stop sign	0				
14	parking meter	0				

Figuur 7. Screenshot van het .csv-bestand met het totaal aantal objecten die in elke categorie zijn gedetecteerd.





voor geautomatiseerde reacties. Om een voorbeeld te geven: een RGB-LED laten knipperen bij een specifiek type object. Om een indruk te krijgen hoe je je project kunt uitbreiden met deze of een andere hardware-besturing, zie **listing 4**. Deze functie schakelt de rode LED in voor een auto, de groene voor een persoon en rood plus groen (waardoor geel ontstaat) voor een vrachtwagen. Na een bepaalde tijd worden alle LED's weer uitgeschakeld.

Bovendien kunnen aanpassingen worden gemaakt voor bredere toepassingen in geavanceerde bewaking, verkeers- en crowd-management, monitoring van wilde dieren, retail-analyse en gezondheidszorg. De vaardigheid van dit systeem om in realtime nauwkeurige objectdetectie uit te voeren, legt de basis voor innovatieve oplossingen in verschillende industrieën en toont de uitgebreide mogelijkheden op het gebied van AI en computervisie.

vertaling: Willem den Hollander – 230749-03

Vragen of opmerkingen?

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

Over de auteur

Saad Imtiaz (Senior Engineer, Elektor) is een mechatronica-ingenieur met ervaring in embedded systemen, mechatronische systemen en productontwikkeling. Hij heeft samengewerkt met talloze bedrijven, variërend van start-ups tot globale concerns, aan product-prototypes en -ontwikkeling. Saad heeft ook een tijd in de luchtvaartindustrie gewerkt en heeft een technologisch start-up-bedrijf geleid. Bij Elektor is hij verantwoordelijk voor projectontwikkeling op het gebied van software en hardware.



- Raspberry Pi 5 Ultimate Starter Kit (8 GB) www.elektor.nl/20721
- Raspberry Pi High Quality Camera Module www.elektor.nl/19279
- Raspberry Pi Camera Module 3 Wide www.elektor.nl/20364

- [1] (Vereenvoudigde) implementatie van YOLOv3: https://analyticsvidhya.com/blog/2021/06/implementation-of-yolov3-simplified
- [2] Uitleg van de YOLOv3-code: https://pylessons.com/YOLOv3-code-explanation
- [3] YOLO: objectdetectie in realtime: https://pjreddie.com/darknet/yolo
- [4] CaptureCount Github-repository: https://github.com/ElektorLabs/CaptureCount

......

Listing 1. Detectiefuncties.

```
import cv2
import pandas as pd
import numpy as np
import subprocess
import os
from datetime import datetime
# Load pre-trained model and configuration
model = './yolo/yolov3.weights' # Update this path to your model's path
                              # Update this path to your config file's path
config = './yolo/yolov3.cfg'
net = cv2.dnn.readNetFromDarknet(config, model)
# Load class names
classes = []
with open("./yolo/coco.names", "r") as f: # Update to the path of your coco.names file
    classes = [line.strip() for line in f.readlines()]
# Function to get output layers
def get_output_layers(net):
    layer_names = net.getLayerNames()
    return [layer_names[i - 1] for i in net.getUnconnectedOutLayers().flatten()]
```

```
# Function to draw bounding box
def draw_bounding_box(img, class_id, confidence, x, y, x_plus_w, y_plus_h):
   label = str(classes[class_id])
   color = np.random.uniform(0, 255, size=(3,))
   cv2.rectangle(img, (x, y), (x_plus_w, y_plus_h), color, 2)
   cv2.putText(img, label, (x-10, y-10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, color, 2)
# Function to calculate centroid of a bounding box
def calculate_centroid(x, y, w, h):
    return (int(x + w/2), int(y + h/2))
# Function to capture image using libcamera-still
def capture_image(image_path):
   subprocess.run(["libcamera-still", "-o", image_path, "--width", "1920", "--height", "1080", "-n", "-t",
                                                 "1"], stdout=subprocess.DEVNULL, stderr=subprocess.STDOUT)
# Initialize a DataFrame to store counts
data_frame = pd.DataFrame(columns=['Timestamp', 'Type', 'Count'])
object_counts = # Object count per category
centroid_tracking = {} # Tracks centroids of detected objects
# Ensure output directory exists
output_dir = "output"
if not os.path.exists(output_dir):
   os.makedirs(output_dir)
```


Listing 2. Centroid Tracking - detectie of een object nieuw of al eerder gedetecteerd is.

```
# Check if this object was already detected
if class_id not in centroid_tracking or not any(np.linalg.norm(np.array(centroid) - np.array(old_centroid))
                                                    < 50 for old_centroid in centroid_tracking[class_id]):
   object_counts[classes[class_id]] += 1
   centroid_tracking.setdefault(class_id, []).append(centroid)
   new_row = pd.DataFrame([{'Timestamp': datetime.now(), 'Type': classes[class_id], 'Count': 1}])
   data_frame = pd.concat([data_frame, new_row], ignore_index=True)
```


Listing 3. Hoofdlus.

```
# Main loop
image_path = "temp.jpg"
object_id = 0 # Unique identifier for each object
try:
   while True:
       capture_image(image_path)
       frame = cv2.imread(image_path)
       if frame is None:
           continue
       Width = frame.shape[1]
       Height = frame.shape[0]
       scale = 0.00392
```

(Vervolg op de volgende pagina)



```
# Create a blob and pass it through the model
blob = cv2.dnn.blobFromImage(frame, scale, (416, 416), (0, 0, 0), True, crop=False)
net.setInput(blob)
outs = net.forward(get_output_layers(net))
# Process the outputs
class_ids = []
confidences = []
boxes = []
centroids = []
conf_threshold = 0.5
nms_threshold = 0.4
for out in outs:
    for detection in out:
       scores = detection[5:]
       class_id = np.argmax(scores)
       confidence = scores[class_id]
        if confidence > conf_threshold:
            center_x = int(detection[0] * Width)
           center_y = int(detection[1] * Height)
           w = int(detection[2] * Width)
           h = int(detection[3] * Height)
            x = center_x - w / 2
            y = center_y - h / 2
            class_ids.append(class_id)
            confidences.append(float(confidence))
            boxes.append([x, y, w, h])
            centroids.append(calculate_centroid(x, y, w, h))
indices = cv2.dnn.NMSBoxes(boxes, confidences, conf_threshold, nms_threshold)
for i in indices:
    box = boxes[i]
    x, y, w, h = box
    x, y, w, h = int(x), int(y), int(w), int(h) # Ensure the values are integers
    centroid = centroids[i]
    class_id = class_ids[i]
    # Check if this object was already detected
    if class_id not in centroid_tracking or not any(np.linalg.norm(np.array(centroid) -
                     np.array(old_centroid)) < 50 for old_centroid in centroid_tracking[class_id]):</pre>
        object_counts[classes[class_id]] += 1
        centroid_tracking.setdefault(class_id, []).append(centroid)
        # Update data_frame using pandas.concat
        new_row = pd.DataFrame([{'Timestamp': datetime.now(), 'Type': classes[class_id], 'Count': 1}])
        data_frame = pd.concat([data_frame, new_row], ignore_index=True)
        # Save the entire frame when an object is detected
        frame_filename = os.path.join(output_dir, f"frame__.jpg")
        cv2.imwrite(frame_filename, frame)
       object_id += 1
    draw_bounding_box(frame, class_id, confidences[i], round(x), round(y), round(x+w), round(y+h))
# Display the frame
cv2.imshow("Object Detection", frame)
# Break loop with 'q' key
```

```
if cv2.waitKey(1) & 0xFF == ord('q'):
           break
finally:
   cv2.destroyAllWindows()
   if os.path.exists(image_path):
       os.remove(image_path)
# Print and save the total count of objects detected per category
total_counts = pd.DataFrame(object_counts.items(), columns=['Type', 'Total Count'])
print(total_counts)
total_counts.to_csv("total_object_counts.csv", index=False)
# Save detailed counts to CSV
data_frame.to_csv("object_counts.csv", index=False)
# Write the total count to a text log
with open("total_counts_log.txt", "w") as log_file:
   log_file.write(str(total_counts))
```

.....

Listing 4. RGB-LED die knippert bij objectdetectie.

```
#include these libraries in the shared code
import RPi.GPIO as GPIO
import time
# GPIO pin setup
RED_PIN, GREEN_PIN, BLUE_PIN = 17, 27, 22
# Update with your GPIO pin numbers
# add this part in the initial part of the code
GPI0.setmode(GPI0.BCM)
GPIO.setup([RED_PIN, GREEN_PIN, BLUE_PIN], GPIO.OUT)
def blink_rgb_led(object_type):
   GPI0.output(RED_PIN, object_type == 'car' or object_type == 'truck')
   GPI0.output(GREEN_PIN, object_type == 'person' or object_type == 'truck')
   GPI0.output(BLUE_PIN, False)
   time.sleep(1)
   GPI0.output([RED_PIN, GREEN_PIN, BLUE_PIN], False)
# Usage of this function:
# ... [Previous code] ...
for out in outs:
   for detection in out:
# ... [add the following lines to the code to this 'for' loop]
       scores = detection[5:]
       class_id = np.argmax(scores)
       confidence = scores[class_id]
       if confidence > conf_threshold:
           detected_object = classes[class_id]
            blink_rgb_led(detected_object) # Blink the LED based on the detected object
# ... [Rest of your detection and saving logic] ...
# ... [Remaining code] ...
```



Spanningsreferentie met Arduino Pro Mini

lineariseer en kalibreer analoge ingangen

Giovanni Carrera (Italië)

Als je de analoge ingangen van je favoriete microcontroller gebruikt, moet je vroeg of laat hun lineariteit en nauwkeurigheid over het hele ingangsbereik controleren. Deze nauwkeurige en compacte op Arduino gebaseerde spanningskalibrator, die stabiele en nauwkeurige referentiespanningen genereert in een bereik van 0...5 V kan dit probleem oplossen.



Misschien vraag je je af wat een spanningskalibrator is en waar hij voor wordt gebruikt. Een spanningskalibrator is een instrument dat uiterst nauwkeurige en bekende spanningen genereert en dat wordt gebruikt om voltmeters, AD- en DA-converters, acquisitiesystemen en dergelijke te kalibreren of te verifiëren.

Bij spanningsmetingen die worden uitgevoerd met microcontrollers is het vaak nodig om de conversieconstante te kennen of de lineariteit van de ingebouwde analoog/digitaal-omzetter te controleren. Als we genoegen nemen met een nauwkeurigheid van zo'n 5% kunnen we volstaan met V_{ref} te delen door de maximale numerieke waarde die de uitgang levert: bij Arduino UNO is dat bijvoorbeels 2¹⁰ – 1 = 1023. Als we de voedingsspanning (5 V) als V_{ref} gebruiken, dan kan deze variëren, afhankelijk van of we het bord via USB of op een andere manier voeden, en ook vertoont de voedingsspanning (veel) ruis. Als we de interne V_{ref} gebruiken (wat zeker de voorkeur verdient), bijvoorbeeld 1,1 V, dan kennen we die waarde niet exact. Andere MCU's zoals de ESP32 hebben niet eens analoge ingangen vanaf 0 V, maar vanaf 80...150 mV, terwijl de volle-schaal-spanning tussen 950 mV en 1100 mV ligt.

Om deze problemen op te lossen moeten we een kalibratie uitvoeren, en dat betekent dat we een stabiele en ruisarme spanningsbron moeten gebruiken in combinatie met een digitale voltmeter die nauwkeuriger moet zijn dan de converter van de MCU. Vervolgens doen we een tiental metingen van verschillende waarden binnen het ingangsbereik van de ADC, nemen die waarden in een Excel-spreadsheet op en passen we lineaire regressie toe om de helling en het snijpunt met de x-as te bepalen. We hebben deze waarden nodig om een waarde in V of mV te krijgen. Dit project biedt het beste van twee werelden: een zeer stabiele, ruisarme spanningsgenerator en een extreem nauwkeurige digitale voltmeter met een meetbereik van 0...5 V en een resolutie in de orde van grootte van 0,2 mV.

De spanningsbron

Voor een stabiele referentiespanning (dat wil zeggen met een geringe temperatuurdrift) is het niet voldoende om een gewone zenerdiode of spanningsregelaar te gebruiken, maar hebben we een schakeling nodig die is ontworpen om een uitgangsspanning te leveren die zo constant mogelijk blijft als de voedingsspanning en de temperatuur in de tijd veranderen. Voor dit project is een LM236H-2.5 gebruikt, die een temperatuurdrift vertoont van slechts 3,5 mV tussen –25 °C en +85 °C wanneer hij (zoals hier) gevoed wordt met een sperstroom van 1 mA. Bij lineair gedrag zou er een verloop moeten zijn van ongeveer $32 \,\mu$ V/°C (ongeveer 12,8 ppm/°C). Als we de temperatuur en stroom constant houden, bedraagt de spanningsverandering ongeveer 20 ppm over een periode van 1000 uur.

Als iets betere karakteristieken gewenst zijn, kan met kleine aanpassingen aan de schakeling een AD680 worden gebruikt. Als daarentegen chips met iets minder hoogwaardige karakteristieken volstaan, kan dit IC worden vervangen door de gangbaarder TL431 zonder dat er iets aan de schakeling moet worden veranderd. **Figuur 1** toont het schema. De twee diodes D2 en D3 worden gebruikt om de thermische drift nog meer te reduceren, en trimmer Rp1 moet worden ingesteld op een uitgangsspanning van 2,49 V, de waarde waarbij de beste



Figuur 1. Schema van de variabele spanningsbron.

eigenschappen worden bereikt. De opamp (verderop meer over diens functie) die wordt gebruikt is een MCP6002, een dual rail-to-rail opamp, die kan werken bij spanningsniveaus die zeer dicht bij de voedingsspanningen liggen (enkele tientallen mV verschil), in tegenstelling tot normale opamps die al in verzadiging gaan bij een paar volt verschil met de voedingsspanningen.

De thermische drift van deze chip is erg gering ($\pm 2 \mu V/C^{\circ}$) en de offset bedraagt een paar mV. Om nog dichter bij nul te komen is schottkydiode D1 gebruikt. Hierdoor ligt de negatieve spanning van de opamps ongeveer 100...200 mV beneden nul (spanningsval van deze diodes) en wordt de minimale uitgangsspanning alleen beïnvloed door de offset van de opamps. Zoals te zien in het schema is de nulpotentiaal of GND niet de negatieve pool van de batterij maar de anode van diode D1, wat precies de reden is waarom de DVM een differentiële ingang nodig heeft.

Hoewel de eigenschappen van dit ontwerp daardoor verslechteren, kan ook een gewone LM358 worden gebruikt in plaats van de MCP6002. Opamp U1A fungeert als een buffer met hoge ingangsimpedantie; op diens uitgang is de potentiometer aangesloten waarmee de spanning kan worden gevarieerd tussen 0 V en 2,5 V. Hier is een tienslagenpotmeter gebruikt; de waarde is niet erg belangrijk; die mag tussen 5 k Ω en 100 k Ω liggen. De tweede opamp U1B is geschakeld als niet-inverterende versterker met een versterking die gelijk is aan

 $G=(1{+}R3/R2)=2$

De uitgangsspanning zal dus tussen ergens tussen 0 V en 5 V liggen. Condensator C3 dient als een laagdoorlaatfilter om thermische halfgeleiderruis te reduceren. Weerstand R4 begrenst de uitgangsstroom in het geval van een kortsluiting. Een uitgangsweerstand van 100 Ω levert geen problemen op omdat deze verwaarloosbaar is vergeleken met de ingangsweerstand van de te kalibreren systemen. De voeding werkt op batterijen om netruis te voorkomen, en een schakelende voeding met een uitgangsruis van een paar honderd kHz met pieken tot 100 mV moet koste wat kost worden vermeden.

De digitale precisie-voltmeter

Om een kalibrator te maken heb je een digitale voltmeter nodig met een precisie die een paar orden van grootte beter is dan die van het systeem dat we moeten kalibreren, en met voldoende nauwkeurigheid. De hier voorgestelde DVM (digitale voltmeter) heeft een 16-bit A/D-omzetter; voor uitsluitend positieve waarden zijn er $2^{15} = 32.768$ waarden vanaf nul mogelijk. De maximale volle-schaal-waarde bedraagt dus 32.767. In dit systeem is de ADC geprogrammeerd voor een volle schaal van 6144 mV, dus de resolutie bedraagt 6.144/32.767 = 0,1875 mV. De converter verdraagt geen ingangsspanningen die 0,3 V hoger zijn dan de voedingsspanning V_{CC}, dus 5,3 V; de spanningsbron is ontworpen voor een maximale spanning van ongeveer 5 V. De maximale waarde, corresponderend met 5 V,is dus 5.000/0,1875 = 26.666.

Dit is zeer hoog vergeleken met dat van een normale 3,5-cijferige paneel-DVM, waar dat 1.999 is. Na toevoeging van een schakelaar kan de voltmeter worden losgekoppeld en apart worden gebruikt. Wat niet gaat,



Onderdelenlijst spanningsbron Weerstanden:

 $\begin{aligned} \text{R1} &= 3 \text{ k, 1\%, 0,25 W} \\ \text{R2,R3} &= 10 \text{ k, 1\%, 0,25 W} \\ \text{R4} &= 100 \ \Omega, 5\% \text{ 3 W, draadgewonden} \\ \text{Rp1} &= 10 \text{ k meerslagen-instelpotmeter} \\ \text{Pot} &= 50 \text{ k tienslagen-potentiometer, lineair} \end{aligned}$

Condensatoren:

 $C1 = 10 \mu/25 V$, tantaal C2 = 100 n keramisch C3 = 10 n keramisch

Halfgeleiders:

U1 = MCP6002, opamp U2 = LM236H-2,5, 2,49V-spanningsreferentie D1 = 1N5819, schottky-diode D2,D3 = 1N4148, diode

Diversen:

SW1 = enkelpolige schakelaar, SP B1...B4 = 4 x AA 1,5V-alkalinebatterijen, met houder



Figuur 2. Belangrijkste delen van de ADS1115-converter (bron: https://ti.com/lit/ds/symlink/ads1115.pdf).



Figuur 3. De ADS1115-module.



Figuur 4. Schema van de ADS1115-module.



Figuur 5. Het 16×2 LC-display.

is toepassing van een hoogohmige spanningsdeler aan de ingang om hogere spanningen te meten; we hebben dat bij andere toepassingen geprobeerd en ondervonden daarbij ruisproblemen waardoor bij een stabiele ingangsspanning maar liefst twee cijfers bleken te variëren.

De ADS1115-converter

Het hart van de DVM is de ADS1115, een uiterst precieze en nauwkeurige 16-bit analoog/digitaal-converter van het delta/sigma-type. Hij heeft maximaal vier ingangen. Van de 16 bits is de meest significante gereserveerd voor het teken; in feite kan de chip negatieve waarden meten, vooral wanneer deze is geconfigureerd in differentiële modus, maar met enkele beperkingen gerelateerd aan de voedingsspanning. Deze interessante chip, gemaakt door Texas Instruments, heeft opmerkelijke eigenschappen:

- > een resolutie van 16 bit, vergeleken met de 10 bit van de Arduino;
- vier single-ended (ten opzichte van massa) of twee differentiële kanalen;
- > bemonsteringsfrequentie van 8...860 monsters/seconde;
- gebruikt een programmeerbare versterker (PGA) waarmee zelfs kleine spanningen gemeten kunnen worden;
- heeft een interne referentie-spanningsbron met lage thermische drift;
- voedingsspanning 2...5,5 V met een stroomverbruik van slechts 150 µA (nog minder in single-shot modus);
- > I²C-interface met verschillende selecteerbare adressen;
- heeft een waarschuwingsfunctie (ALERT/RDY) om spanningen efficiënt te controleren. Dit kan het stroomverbruik verminderen, de microcontroller wekken en aan het werk zetten wanneer bepaalde gebeurtenissen optreden, of interrupts genereren.

Figuur 2 toont het blokschema van de ADS1115-chip, die twee conversiemodi kent:

- > continue conversie: de ADS1115 voert continu omzettingen uit. Zodra een omzetting is voltooid, zet de ADS1115 het resultaat in het conversieregister en begint hij direct met een volgende conversie;
- > single-shot conversie: de ADS1115 wacht tot het OS-bit hoog wordt gemaakt. Zodra dit is bevestigd, wordt het bit op 0 gezet, wat aangeeft dat er momenteel een conversie wordt uitgevoerd. Zodra de conversiegegevens klaar zijn, wordt het OS-bit opnieuw bevestigd en schakelt de chip uit. Het schrijven van een 1 naar het OS-bit tijdens een conversie heeft geen effect.

In dit project wordt de single-shot modus gebruikt waarbij elke halve seconde een conversie wordt uitgevoerd.

ADS1115-module

Modules als die van **figuur 3** (die we in dit project gebruikt hebben) zijn goed verkrijgbaar. Zo'n module heeft niet veel componenten, zoals te zien is in het schema van **figuur 4**. Om deze chip met de Arduino IDE te gebruiken, hebben we de *ADS1115_WE*-bibliotheek van Wolfgang Ewald gebruikt, die kan worden gedownload van de IDE zelf (Library Manager) of van het web [1]. Deze bibliotheek heeft talloze functies waarmee alle functies van de chip kunnen worden benut.



Figuur 6. Schema van de DVM.



Figuur 7. Bedrading tussen de DVM en de spanningsbron.

LCD

We hebben gekozen voor een LC-display dat compatibel is met de Hitachi HD44780-controller. Omwille van duidelijk leesbare karakters hebben we een klassiek tweeregelig display met 16 karakters gebruikt, zoals in **figuur 5**. De reden voor deze keuze is dat deze displays goed leesbaar zijn, een gering stroomverbruik hebben (zonder achtergrondverlichting) bij 5V-voeding en dito logische niveaus, die perfect compatibel zijn met Arduino Pro Mini. De benodigde Arduinobibliotheek is hier te vinden [2].

Bedrading van de DVM

Figuur 6 toont de bedrading van de DVM. Als microcontroller hebben we een uiterst compacte Arduino Pro Mini gebruikt; deze levert ook de 5 V om het LC-display en de converter van stroom te voorzien. In dit project worden de ingangen A0 en A1 van de ADC-module gebruikt die geconfigureerd zijn als differentiële ingang. Het verbruik van het hele systeem bedraagt ongeveer 20...25 mA als de achtergrondverlichting van het display niet wordt gebruikt. De ideale spanning is 6 V; het systeem accepteert ook hogere spanningen (maximaal 12 V), maar de kleine ingebouwde regelaar kan oververhit raken. De voeding voor de achtergrondverlichting mag niet door de interne regelaar worden verzorgd, maar moet worden geleverd door de externe regelaar met weerstand R1 in serie, zoals in figuur 6,. De waarde van R1 moet worden berekend op basis van de gewenste stroom.

In het prototype is een weerstand van 82 Ω gebruikt, maar een waarde van 100 Ω is ook geschikt en bespaart nog een paar mA. Oudere displays trokken zelfs een stroom tot 100 mA, omdat ze technologisch oudere LED's gebruikten. Recente displays hebben een maximale stroom in de buurt van 20 mA omdat ze LED's met een hoge helderheid hebben. Een 4-polige connector verbindt de DVM met de print van de variabele spanningsbron. Condensator C5 en weerstanden R2 en R3 (510 Ω) vormen een eerste-orde laagdoorlaatfilter en reduceren ruis. De tijdconstante bedraagt $\tau = 47,94 \,\mu$ s, wat overeenkomt met een kantelfrequentie van

 $f = 1/(\tau * 2*\pi) = 3,320 \text{ kHz}$

Bedrading

In het prototype zijn twee printen aanwezig, een voor de DVM en een voor de spanningsbron (**figuur 7**), maar niets belet je om alle componenten op één print te monteren. De voeding voor de digitale schakelingen produceert ruis als gevolg van geschakelde stromen, zodat ontkoppelcondensatoren werden gebruikt om ruispieken te reduceren (bijvoorbeeld tantaalcondensator C2).

Het prototype

Figuur 8 toont het prototype. Je ziet twee stukken gaatjesprint; het display en de ADC-omzetter zijn aangesloten met 16- en 10-polige busstrips (raster 2,54 mm), en de Arduino Pro-module is verbonden met twee 12-polige busstrips.

Figuur 9 toont de DVM-print. Helaas zijn de pinnen A4 en A5 van de l²C-bus van de Arduino Pro Mini niet beschikbaar op de 12-polige busstrips en passen ze ook niet in een 2,54mm-raster, dus werden er twee draden (wit en bruin, afgebeeld) en een 2-pins connector gemonteerd om de signalen naar de ADS1115-module te brengen. De 16-polige busstrip linksboven is voor het LC-display, dat onder een hoek van 90° is gemonteerd. De compacte gaatjesprint is aan het display bevestigd met een kleine koperen beugel, die linksboven op de foto te zien is.



Figuur 8. Het voltooide prototype van binnen.



Figuur 9. De opgebouwde DVM-print.

Programmeren

Voor het programmeren hebben we een seriële USB/TTL-adapter nodig – zoiets als in **figuur 10**. Natuurlijk moeten we ook de adapterdriver laden die, naast de Rx- en Tx-signalen, ook DTR moet ondersteunen die wordt gebruikt om het systeem te resetten. Hij dient ook als een seriële USB-interface, maar in dit geval gaat het vooral om het overbrengen van de sketch, gecompileerd vanuit de Arduino IDE, naar het systeem. In de Arduino IDE moet je het board instellen als *Arduino Pro* op



Figuur 10. Aansluitingen tussen de seriële adapter en de Arduino Pro Mini.

16 MHz en 5 V. Als het programmeren is gelukt, kun je de programmer verwijderen en het board via de batterijen voeden.

De connector is een 6-polige busstrip in 2,54mm-raster, die direct op de male header van de adapterprint wordt gesoldeerd; natuurlijk kunnen ook female/female jumperkabels worden gebruikt. Niet alle seriële adapters hebben dezelfde pinning als in de figuur, of zelfs dezelfde chip. Bij sommige seriële adapterchips, zoals de Prolific PL2303, zijn problemen opgetreden met de driver onder Windows 10.

Kalibratie van de DVM

Hoewel de ADS1115-metingen al erg nauwkeurig waren, werd de kalibratie uitgevoerd met een 5,5-digit HP3478A benchtop-multimeter en een resolutie van 0,1 µV. Hiertoe moeten we deze regel in de functie printData()

// calibrated and result in millivolts voltage = Ch0*1.000515+0.022;

vervangen door deze (we verwijderen de correctie):

// not calibrated and result in millivolts voltage = Ch0;

Er werden ongeveer tien waarden gegenereerd, de meetwaarden van de referentie-multimeter en de meetwaarden van de kalibrator werden in Excel ingevoerd en als scatterplot (corelatiediagram) weergegeven met de trendline (lineaire regressie) plus de opties Display equation on graph en Display R square value on graph (determinatiecoëfficiënt) aangevinkt. Het resultaat was uitstekend, zoals je ziet in figuur 11. De waarde van de determinatiecoëfficiënt was 1 met maar liefst zeven

Onderdelenlijst DVM

Weerstanden:

 $R1 = 82 \Omega$, 5%, 1 W , zie tekst $R_{2}R_{3} = 510 \Omega$, 1%, 0,25 W Rp1 = 22 k instelpotmeter

Condensatoren:

C1,C4 = 100 n/50 V, keramisch $C2 = 10 \mu/25 V$, tantaal $C3 = 1 \,\mu/63$ V, keramisch C5 = 47 n/63 V, polyester

Modules:

1 x Arduino Pro Mini (of compatibel) 1 x 16x2 LC-display met achtergrondverlichting 1 x ADS1115 A/D-omzetter

Diversen: 2x 12-polige busstrip 10-polige busstrip 16-polige busstrip



Figuur 11. DVM-kalibratiegrafiek (linearisatie).



cijfers achter de komma gelijk aan nul: dat betekent een extreem goede lineariteit. Om de gegevens te corrigeren, moeten we de inverse formule toepassen, dat wil zeggen de assen omkeren. Dan krijgen we:

- > helling = 1,000515286
- snijpunt x-as = 0,022110099
- > fout = 0,051941236

Je ziet dat de fout minimaal is. Als we geen zeer nauwkeurig instrument tot onze beschikking hebben, kunnen we de correctie achterwege laten en genoegen nemen met een standaardfout van 0,051941236; anders corrigeren we de lijn met de waarden die onze kalibratie heeft opgeleverd. In het prototype was de minimaal gegenereerde spanning 1,7 mV en het maximum 4.997,03. Het minimum is niet gelijk aan nul vanwege de offset van de opamps.

Programmabeschrijving

Het programma is relatief eenvoudig; na initialisatie van de I/O, het display en de ADC-omzetter wordt elke 500 ms een meting uitgevoerd. De functie readChannel(ADS1115_MUX channel) is de functie die wordt gesuggereerd voor single shot-modus en levert een meetwaarde in millivolt. De functie LCDprintLine(String text, byte line) drukt een string af op regel 0 of regel 1. De functie printData() corrigeert de meting met de resultaten van de initiële kalibratie en drukt deze af. 230715-03

Vragen of opmerkingen?

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

Listing 1

```
/* program ArduCalibrator.ino for the ADS1115, 16 bit 4 ch ADC
a single differential input (A0,A1), Ch0 6144 mV (5000 generated)
read the precision voltage source
uses an Arduino Pro Mini, LCD 16x2 display
Giovanni Carrera 23/11/2022 with calibration
*/
#include <Wire.h>
#include<ADS1115_WE.h>
#include <LiquidCrystal.h>
ADS1115_WE adc = ADS1115_WE();// uses Wire / ADC Address = 0x48
// LCD pins
#define rs 7
#define en 6
#define d4 5
#define d5 4
#define d6 A2
#define d7 A3
// initialize the library by associating any needed LCD interface pin
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
#define Alert 2 // AD Alert pin
const unsigned long deltat_ms = 500;
unsigned long cms, pms;
float Ch0,Ch1;
void setup(){
  Wire.begin();
  //Serial.begin(115200);
   pinMode(Alert,INPUT);
   // set up the LCD's number of columns and rows:
   lcd.begin(16, 2);
   lcd.print(F("ArduCalibrator"));
   lcd.setCursor(0, 1);// print on the second row
   lcd.print(F("GCar V231122"));
   delay(2000);
   if(!adc.init()){
```

```
lcd.setCursor(0, 1);
    lcd.print(F("No ADS1115 found"));
    while( true );// ends here
   }
   adc.setVoltageRange_mV(ADS1115_RANGE_6144); // 6144 mV input range
  LCDprintLine("PRECISION SOURCE", 0);
}
void loop() {
  cms = millis();
  // checks if passed deltat milliseconds
  if(cms - pms > deltat_ms) { // period timebase
      pms = cms;// update pms
    Ch0 = readChannel(ADS1115_COMP_0_1);// Ch0 differential A0(+) with A1(-)
    Ch0 = readChannel(ADS1115_COMP_0_1);// repeating helps
     printData();
}
```



Over de auteur

Giovanni Carrera is afgestudeerd elektronicus. Als professor aan de faculteit Scheepsbouwkunde in Genua, Italië, doceerde hij talloze cursussen, zoals scheepsautomatisering en de simulatie van scheepsvoortstuwingssystemen. Carrera ging eind jaren 1970 aan de slag met de 6502 CPU en stapte daarna over op andere processoren. Tegenwoordig houdt hij zich bezig met het ontwerpen en ontwikkelen van analoge en digitale elektronische schakelingen, waarover hij veel heeft geschreven op zijn blogs (ArduPicLab en GnssRtkLab) en in verschillende tijdschriften.



Gerelateerde producten

- > SparkFun Arduino Pro Mini 328 (5 V, 16 MHz) www.elektor.nl/20091
- Standard 2x16 Character back-lit LCD www.elektor.nl/16414



WEBLINKS

- [1] Wolfgang Ewald's ADS1115-bibliotheek: https://github.com/wollewald/ADS1115_WE
- [2] LiquidCrystal-bibliotheek: https://github.com/arduino-libraries/LiquidCrystal
- [3] Downloads voor dit project: https://elektormagazine.com/Arducalibrator
- [4] G. Carrera, "Ruisarme ADC-kalibrator voor moderne microcontrollers", Elektor Summer Circuits 2022: http://www.elektormagazine.nl/magazine/elektor-263/60814
- [5] Ultra-Small, Low-Power, 16-Bit Analog-to-Digital Converter with Internal Reference: https://ti.com/lit/ds/symlink/ads1114.pdf



FPGAS Soor beginners

van MCU naar FPGA-programmering

Theo Mulder (Nederland)

Er zijn veel goedkope microcontroller development boards beschikbaar, maar FPGA's blijven duur. Voor toepassingen die hogere rekenprestaties vereisen (zoals AI) of snellere I/O (bijvoorbeeld high-speed camera's en dito displays, snelle ADC of DAC) is het wenselijk om kleine en betaalbare FPGAboards te vinden. Laten we daarom in de opwindende wereld van FPGA's voor beginners duiken!

Wanneer je uit enkele goedkope FPGA's wilt kunnen kiezen, moet je de huidige zakelijke trends enigszins begrijpen. De FPGA-markt werd in 2022 geschat op 8 miljard USD en zal naar verwachting groeien tot 15,5 miljard USD in 2027. Deze sector is competitief en werd gekenmerkt door veel overnames en consolidaties. Momenteel zijn als belangrijkste spelers overgeleven AMD/Xilinx, Intel/Altera, Lattice, Microchip, QuickLogic, Efinix, Flex Logix, GOWIN, Achronix, S2C en Renesas. Door overnames blijft het landschap veranderen. Al in 1999 werd Vantis door Lattice overgenomen, terwijl Acree in 2001 hetzelfde lot onderging. Microsemi nam in 2010 Actel over. In 2013 waren de belangrijkste spelers Xilinx, Altera, Actel, Vantis, Lattice, Lucent, QuickLogic en Cypress. Het spel van fusies en overnames is echter nooit gestopt; zelfs de "Grote Twee" (Xilinx en Altera) werden op hun beurt weer overgenomen door respectievelijk AMD en Intel. In 2019 was Xilinx marktleider met een aandeel van 52%, gevolgd door Altera met 35% terwijl Microchip, Lattice en anderen kleinere marktaandelen hadden. Ondanks

de heerschappij van deze grotere bedrijven blijven kleinere FPGA-leveranciers actief, wat uitstekend is voor de ondersteuning van verschillende soorten bedrijven en waardoor innovatie wordt gestimuleerd. Het toenemende gebruik van FPGA's in de auto-industrie biedt kansen voor leveranciers zoals Lattice, omdat ontwerpers in de industrie op zoek zijn naar kosteneffectieve alternatieven voor het duurdere aanbod van de grootste bedrijven. Renesas is een belangrijke concurrent aan het worden dankzij zijn gevestigde aanwezigheid in de automotive sector.

De dynamiek van de FPGA-markt met voortdurende verschuivingen in de concurrentieverhoudingen, overnames en het verschijnen van nieuwe spelers kan gunstig zijn voor wie op zoek is naar goedkope FPGAopties voor zijn projecten.

Betaalbare FPGA-boards voor beginners

Omdat de toolchains van de verschillende FPGA-fabrikanten nogal van elkaar verschillen en het leren gebruiken van nieuwe software tijd kost, is het zinvol om boards te onderscheiden naar FPGA-fabrikant. Op die manier is de leertijd goed geïnvesteerd als je in de toekomst besluit te upgraden naar een hoogwaardiger model van dezelfde FPGA-fabrikant.

De drie belangrijkste fabrikanten zijn Altera/Intel, Xilinx/AMD en Lattice. Die laatste pleegt minder krachtige maar betaalbaarder modellen aan te bieden dan de beide eerstgenoemden.

Beginnend met boards op basis van FPGA's van Lattice Semiconductor, biedt het Olimex iCE40HX1K-EVB evaluatieboard met een prijs van € 15 een uiterst betaalbare instap, hoewel er een externe programmer voor nodig is. Dit board bevat een iCE40HX1K met 1280 logische elementen (LE's). Interessant is dat deze FPGA een intern niet-vluchtig configuratiegeheugen heeft dat kan worden geprogrammeerd via de SPI-interface van een Arduino-board dat je misschien hebt liggen, dankzij de sketch die Olimex ter beschikking stelt. De officiële programmer van Lattice gebruikt een FT232H van FTDI, en het is ook mogelijk om hiervoor een FT232H breakout-board te gebruiken. Een van de spannende eigenschappen van deze FPGA-familie is dat die compatibel is met de open-source grafische tool Icestudio, zodat gebruikers kunnen experimenteren met grafisch programmeren op basis van functieblokken.

Voor diegenen die op zoek zijn naar andere compacte, op breadboard bruikbare en budgetvriendelijke opties, zijn er interessante producten in de TinyFPGA-familie. De TinyFPGA BX kost \in 40 en bevat een Lattice iCE40LP8K, die ook compatibel is met Icestudio. De AX1- en AX2-modellen zijn verkrijgbaar voor respectievelijk \in 13





en \in 17 en vormen kleinere, goedkopere alternatieven, zij het met minder mogelijkheden. Helaas zijn deze drie modellen vaak niet op voorraad. Van een andere fabrikant biedt de Upduino v3.1 voor \in 30 een iCE40UP5K FPGA (5,3 kLE).

De Lattice iCEstick is een compacte optie voor € 48 met de vormfactor van een USB-stick, uitgerust met een iCE40HX1K FPGA en een onboard-programmer. Dit is de officiële tool van Lattice en werkt dus goed met de software van Lattice. Dit tool wordt in veel tutorials gebruikt.

Om de Lattice-opties af te ronden, noemen we ook een open-source platform met de vormfactor van een Arduino UNO: de Alhambra II (iCE40HX4K, 3,52 kLE, \in 60) en ook het Nandland.com Go Board (iCE40HX1K, \in 70) dat de uitstekende tutorials op Nandland.com ondersteunt. Overstappend op Altera (nu eigendom van Intel) biedt het MAX1000-board van Trenz Electronic een MAX10 FPGA, verkrijgbaar met 8 of 16 kLE (\in 34 tot \in 49), met een breadboard-vriendelijk ontwerp. De CYC1000 [1] (**figuur 1**), ook van Trenz en met een prijskaartje van \in 40 legt de lat iets hoger met een Cyclone 10 FPGA (25 kLE). Voor traditionelere boards met meer toeters en bellen zijn er de opties van Terasic. Sommige van hun producten zijn duur, maar andere zijn erg interessant voor beginners, zoals de DEO-nano die een Cyclone IV FPGA met 22 kLE en een heleboel onboard functionaliteit biedt voor \in 110. De DE10-Lite (\in 140) is een board met veel mogelijkheden, een MAX10 FPGA (50 kLE), verschillende I/O-opties, zeven-segment displays, LED's en schakelaars, plus ondersteuning voor Arduino-shields.

Last but not least bieden onder de paraplu van AMD (voorheen Xilinx) de Digilent Cmod S7 en Cmod A7-35T op breadboard bruikbare modellen met Spartan 7 en Artix 7 FPGA's (respectievelijk 23,4 en 33,3 kLE) met een prijskaartje van € 90.

Voor een rijke leerervaring op een groter board biedt de Digilent Basys 3 een Artix 7 FPGA met 33,3 kLE, uitgebreide I/O opties inclusief VGA out en USB host, voor een prijs van \in 155.

In dit artikel concentreer ik me op producten van Intel/Altera. Ik kan de CYC1000 aanbevelen. In de volgende paragrafen gaan we experimenteren met Quartus Prime Lite van Intel, de gratis versie van het officiële ontwikkel-tool voor de FPGA's van Altera/ Intel.

Installatie

Op de Intel Quartus Prime Design Softwarepagina [2] vind je download-links voor de laatste versie, voor Windows of Linux. Een goede internetverbinding is een vereiste, want het bestand is ongeveer 5,5 GB groot. Het installatieprogramma heeft 15 GB schijfruimte nodig. Sommige delen van Quartus Prime Lite gebruiken code van vroege versies van Quartus. Dat heeft tot gevolg dat spaties in bestandsnamen moeten worden vermeden, hoe verrassend dat naar hedendaagse maatstaven ook moge klinken.

We hebben alleen de Low Level Designtool nodig. Er zijn aanvullende pakketten die buiten het bestek van deze inleiding vallen, zoals DSP Builder (voor signaalverwerkingstoepassingen), High Level Synthesis Compiler (gebruikt met C++ als invoer, voor geavanceerde toepassingen) en de Nios II Embedded Design Suite (gebruikt om een Nios II soft processor in een FPGA te implementeren).

Een nieuw project aanmaken

Maak eerst een map aan, bijvoorbeeld C:\ Projects\IntelFpga\Elektor\BeginExample. Open dan Quartus Prime Lite. Je ziet het hoofdscherm van **figuur 2**. Ga naar File en selecteer New Project Wizard. Nadat je het



Figuur 2. Quartus Lite Edition, met ElektorFirstTop.bdf.

Logische geschiedenis: van TTL tot FPGA

Voor de komst van FPGA's waren digitale schakelingen vooral gebaseerd op TTL-chips en programmeerbare IC's zoals FPLD's, PLA's en PAL's. De TTL SN7400-serie werkte tot 20 MHz. Altera introduceerde 40 jaar geleden de EP300, en snel daarna kwam de XC2064 van Xilinx. De EP300 werd gefabriceerd een 5000nm-proces en bood snelheden tot ongeveer 10 MHz, terwijl moderne mid-range FPGA's een 20nm-proces gebruiken en kloksnelheden van minstens 250 MHz bereiken.

De EP300 was een 20-pins herprogrammeerbaar IC dat gebruik maakte van de A+PLUS software die draaide op IBM-PC's onder DOS. De XC2064 van Xilinx had logische cellen met elk 1200 poorten en gebruikte een low-power CMOS-proces, met ontwerpsoftware zoals Future-Net, VIEW-logic en XACT-Design-Editor. De XC2064 werd geadverteerd met een interne schakelsnelheid van 100 MHz, maar haalde meestal ongeveer 70 MHz.

Aanvankelijk maakten FPGA-softwaretools gebruik van schematic capture, maar dit werd later aangevuld met tekstgebaseerde hardware-beschrijvingstalen (Hardware Description Language, HDL) zoals AHDL voor Altera en VHDL en Verilog voor Xilinx. Tegenwoordig ondersteunen beide bedrijven VHDL en Verilog. Oudere versies van Altera's Quartus-tool gebruikten AHDL met ingebouwde simulatiefuncties, en vereisten later het gebruik van ModelSim voor simulatie vanwege het gebrek aan AHDL-ondersteuning. Tegenwoordig bieden alle fabrikanten een gratis versie van hun tools aan.



welkomstscherm hebt bewonderd, klik je op Next. Wanneer naar de werkdirectory wordt gevraagd, kies je de directory die je zojuist hebt gemaakt. Er moet een projectnaam worden gekozen, evenals een naam voor de top-level ontwerpentiteit, die de compiler zal gebruiken als de root van het ontwerp. In dit voorbeeld hebben we ElektorFirst en ElektorFirstTop gebruikt. Kies in de volgende stap Empty Project. Dan kom je bij de stap Add files. Aangezien we geen bestanden hebben, klik je gewoon weer op Next.

Je komt dan op een pagina waar je een apparaat kunt selecteren, dit kan via een van de tabbladen *Device* of *Board*. Op die laatste staan verschillende development boards in de MAX10- en Cyclone V-families; later kunnen nog meer boards worden geïnstalleerd. Op het tabblad *Device* kunt je de exacte familie en het typenummer kiezen van de FPGA die je gebruikt, zoals de 10LC025YU256C8G uit de Cyclone 10 LP-familie als je het CYC1000 bord hebt. De Cyclone 10-familie bestaat uit goedkope en energiezuinige apparaten die zijn afgeleid van de Cyclone V.

Als je geen development board hebt, kun je toch dat model selecteren om verder te

gaan met de tutorial en te experimenteren met het compilatieproces. Klik op Next, nogmaals Next en dan Finish.

Structuur van de projectmap

Neem even de tijd om je werkdirectory te bekijken. Je zult merken dat er een db-directory is die Quartus later zal gebruiken. Het bestand *ElektorFirst.qpf* is de Quartus Project File (QPF). Als je later naar dat project wilt terugkeren, kun je gewoon op dit bestand dubbelklikken om de Quartus-software opnieuw te starten. Er is ook een bestand *ElektorFirst.qsf*, dat het Quartus Settings File (QSF) is. Dit zijn allebei tekstbestanden, dus jr kunt ze bekijken als je nieuwsgierig bent – klik met de rechter muisknop en kies Open with om een teksteditor te vinden.

Het project openen

In de linkerbovenhoek zie je de Project Navigator met Hierarchy geselecteerd. Zoek naar ElektorFirstTop, die we eerder hebben genoemd. Als je erop klikt, verschijnt er een pop-up met de tekst "Can't find design entity ElectorFirstTop" omdat we nog geen bestand hebben gemaakt. Klik op OK en ga dan naar het menu File. Hier zie je verschillende bestandstypen die je kunt aanmaken. Kies *Block Diagram/Schematic File* en klik op OK.

Nu krijg je een wit blad (canvas) te zien in het middelste deelvenster, met bovenaan het label *Block1.bdf*. Ga naar *File*, selecteer *Save as* en er wordt voorgesteld om het bestand op te slaan als *ElektorFirstTop.bdf*. Dat is perfect – klik op *Save*. Als je nu in de *Project Navigator* en *Hierarchy* op *Elektor-FirstTop* klikt, kom je op dit lege schemablad terecht.

Onderdelen toevoegen aan ons eerste schema

Om ervoor te zorgen dat al onze lezers, zelfs degenen die niet bekend zijn met de talen Verilog en VHDL, deze tutorial kunnen volgen, gaan we schematisch programmeren. Klik op de *ElektorFirstTop.bdf* tab om het blad te openen, klik met de rechtermuisknop ergens in het blad en kies *Insert Symbol*. Hierdoor verschijnt het menu *Libraries*. Klik op het kleine driehoekje naast *Libraries* om de lijst uit te klappen; je ziet nu categorieën als *Megafunctions*, *Other*, *Primitives* en meer. Er valt hier veel te ontdekken, dus neem de tijd om wat rond te neuzen. Scroll naar de onderkant van het *Libraries*venster en typ "Input" in het veld *Name*. Er verschijnt een ingangssymbool; klik op *OK*. Zet dit symbool ergens op de pagina, bij voorkeur linksboven. Voeg vervolgens een tweede ingang toe door met de rechtermuisknop te klikken en *Insert Symbol* te kiezen. Aangezien "Input" nog steeds in het veld *Name* zou moeten staan, klik je gewoon weer op *OK* en plaats je deze nieuwe ingang onder de eerste op je blad.

Vervolgens voegen we een D-flipflop toe. Klik met de rechtermuisknop, selecteer Insert Symbol invoegen en typ "dff" in het veld Name. Kies de dff uit de beschikbare opties en klik op OK, plaats het dan op het blad. Om deze stap af te ronden klik je nogmaals met de rechtermuisknop, selecteer je Insert Symbol en vul je in het veld Name "Output" in en klik je op OK. Zet deze uitgang ook op het blad.

Aansluitingen herbenoemen

We moeten aansluitpinnen betekenisvolle namen geven. Het is mogelijk om voor- of achtervoegsels toe te voegen ter verduidelijking, zoals *i* voor input, o voor output, sync of async voor respectievelijk synchroon of asynchroon enzovoort. En dus geven we de ingang *pin_name1* de nieuwe naam *iDinpAsync*. Dubbelklik hiervoor op *pin_name1* en voer de nieuwe naam in. Geef ook *pin_name2* de nieuwe naam *iClk* en de uitgang *pin_name3* de naam *oDout_sync*. Klik dan op *File* en *Save*.

Verbindingen tekenen

Klik op de pin *iDinpAsync*. Hermee wordt automatisch de *Orthogonal Node Tool* geactiveerd om verbindingen te tekenen. Begin vanaf deze pin te tekenen en verbind hem met de D-ingang van de D-flipflop (DFF). Verbind vervolgens *iClk* met de klokingang van de DFF, die wordt aangegeven door een driehoekje in het symbool. Maak ten slotte een verbinding van de Q-uitgang van de DFF naar het *oDout_sync* uitgangssymbool. Sla het geheel vervolgens weer op.

Compileren

Klik op het blauwe driehoek-symbool in de werkbalk om het ontwerp samen te stellen. Houd het deelvenster *Task* aan de linkerkant in de gaten om de voortgang te volgen en let op of er meldingen onder aan het scherm verschijnen. Het is best wel een interessant proces.

Na de compilatie wordt in het middelste deelvenster de *Flow Summary* getoond. Deze samenvatting geeft aan dat slechts één van de in totaal 24.624 logische elementen is gebruikt. Het komt zelden voor dat alle logica-elementen van een IC gebruikt worden; in grotere, goed geoptimaliseerde ontwerpen kan ongeveer 80% van de LE's gebruikt worden. Voorbij dat punt kun je routeringsproblemen tegenkomen of kan het IC niet voldoen aan de timingvereisten vanwege 'congestie'. Het overzicht laat ook zien dat er één register is gebruikt en dat er drie van de 151 beschikbare pinnen in gebruik zijn, wat neerkomt op 2% van het totaal. Houd er echter rekening mee dat sommige pinnen gereserveerd kunnen zijn en niet beschikbaar voor algemeen gebruik.

Werp nog een keer een blik op je projectmap. Je ziet daar een aantal nieuwe submappen. Neem een kijkje in de map *db* om een idee te krijgen van het vele werk dat op de achtergrond is gedaan. In de map *output_ files* staat het bestand *ElektorFirstTop.sof* waarmee je FPGA geprogrammeerd wordt.

Kies de ideale FPGA voor je project

Bij het selecteren van FPGA's en boards zijn er verschillende parameters waar rekening mee moet worden gehouden. Een daarvan is het aantal logische elementen, wat een idee geeft van de grootte van de FPGA. Ook moet rekening worden gehouden met de noodzaak van low- of high speed-I/O's. High-end FPGA's met snel geheugen en transceivers zijn duur, net als de bijbehorende softwaretools. Natuurlijk zijn de classificaties high-, mid- en low-end subjectief en kunnen ze per leverancier verschillen. Voor de twee grootste FPGA-verkopers geldt dat wat wordt beschouwd als mid-range prestaties, door Lattice als high-end kan worden bestempeld.

Als het gaat om I/O's wordt vaak over transceivers gesproken. Dit zijn seriële verbindingen die een hoge overdrachtssnelheid bieden (bijvoorbeeld 4...32 Gbps), maar die de kosten sterk beïnvloeden. Ze zijn lang niet voor alle toepassingen noodzakelijk. In plaats van snelle PCIe- of COM Express-verbindingen is het gebruik van USB een kosteneffectievere en handiger optie, eventueel met een FTDI-chip voor interfacing.

Voor snelle ADC- of DAC-verbindingen met een FPGA is het mogelijk om de JESD204B-interface te gebruiken, maar dat vereist transceivers aan zowel de FPGAals de ADC/DAC-kant, wat duur kan worden. Aandacht voor signaalintegriteit is nodig bij het routeren van de print, maar de layout wordt enigszins vereenvoudigd door het geringere aantal sporen. Een andere optie als er grote doorvoersnelheden nodig zijn, is om meerdere langzamere LVDS-paren (Low Voltage Differential Signaling) parallel te gebruiken. In elk geval is het nuttig om een redelijk nauwkeurige berekening te maken van je doorvoervereisten als hulp bij het kiezen van een geschikte component.

Ik ben zelf geneigd om interfaces als low-speed te beschouwen als ze beneden 1000 Mbps werken, en als high-speed als ze daarboven komen. De kosten van FPGA's nemen toe met het aantal snelle interfaces. Toch zijn er ook veel momenten waarop zelfs LVDS met doorsnee-snelheid niet nodig is: I/O-pinnen voor algemeen gebruik kunnen volstaan.

Op FPGA's zijn de general purpose I/O-pinnen de langzaamste van de beschikbare interfaces, maar ze zijn nog sneller dan de GPIO-pinnen van microcontrollers. Ze zijn meestal compatibel met 3,3V-, 2,5V- of 1,8V-logica. Als je voor je volgende project een microcontroller wilt gebruiken, maar snellere I/O's wilt dan wat typische micro-controllers aan boord hebben, kunnen FPGA's een compacte oplossing vormen, omdat het mogelijk is een processor (RISC-V of ander) in de FPGA-logica te implementeren.

VHDL of Verilog?

De vraag of een beginner Verilog dan wel VHDL moet leren tijdens zijn FPGA-leertraject, wordt beïnvloed door verschillende factoren. Voor degenen die in de defensieof luchtvaartsector willen werken, is VHDL de overwegende taal, vooral in Europese landen zoals Duitsland en Frankrijk. Aan de andere kant wordt in de Verenigde Staten en in veel sectoren buiten defensie vaker Verilog gebruikt.

Maar iedereen die een carrière in digitaal ontwerpen ambieert, zal waarschijnlijk beide talen moeten kunnen gebruiken. Het is belangrijk om je aan te passen aan de context; als het voor academische doeleinden is, sluit je dan aan bij de taal die in je cursussen wordt onderwezen, terwijl je voor professionele omgevingen het beste kunt gebruiken wat je collega's ook gebruiken.

Vanuit een technisch perspectief is Verilog compacter en lijkt het in sommige opzichten op C, wat niet altijd voordelig is omdat het kan leiden tot een softwaregeoriënteerde denkwijze in plaats van de vereiste focus op hardware. VHDL's uitgebreide en sterk typegebaseerde aard kan helpen bij het vermijden van bepaalde fouten die Verilog's zwakke typologie zou kunnen missen. Dit onderscheid kan VHDL een beter startpunt maken voor diegenen die de voorkeur geven aan een taal die een hardware-georiënteerd denkproces afdwingt, wat cruciaal is voor FPGA-ontwerp.

Bestudeer tenslotte het bestand *Elektor-FirstTop.pin* (open het in een teksteditor). Dit bestand toont de pintoewijzingen die de compiler heeft gemaakt. Omdat we geen specifieke beperkingen hebben opgegeven, heeft Quartus willekeurige pinnen gekozen. Normaal gesproken zou je als ontwerper zelf je pintoewijzingen definiëren. Zoek naar de pinnen *iDinpAsync*, *iClk* en *oDout_sync*. In mijn geval was *iDinpAsync* verbonden met pin T4, werkte deze met een logisch niveau van 2,5 V en bevond deze zich in bank 3 van de FPGA.

De FPGA programmeren

Natuurlijk zijn we benieuwd of dit bij een echte FPGA werkt; laten we het daarom proberen! Sluit eerst je board aan op een vrije USB-poort van je computer en volg de stappen om het configuratiebestand te uploaden naar het SRAM van de FPGA. Een alternatief is uploaden naar flash, waardoor het ontwerp ook na uitschakelen in de FPGA bewaard blijft.

Ga hiervoor naar Tools en dan Programmer, of dubbelklik gewoon op Program Device in de taaklijst. Klik op Hardware Setup en selecteer USB-Blaster [USB-0]. Klik dan op Add File en kies het SRAM-objectbestand ElektorFirstTop.sof dat na de compilatie is gegenereerd in de map output_files. Klik op Start. Vergeet niet de configuratie van de programmer op te slaan, bijvoorbeeld als example1.cdf, zodat hij de volgende keer met deze instellingen wordt geopend. Na een paar seconden is de overdracht voltooid! We hebben met succes het ontwerp in de FPGA geladen, waar het blijft totdat de stroom wordt uitgeschakeld. Gefeliciteerd, je bent nu de bezitter van een uiterst fraaie D flip-flop, die je kunt uitproberen door draden en schakelaars aan te sluiten op de juiste pinnen van het CYC1000-board!

Hiërarchisch ontwerp

Het kan handig zijn om een deel van een systeem één keer te ontwerpen en het dan steeds opnieuw te gebruiken op verschillende plaatsen in het systeem. Hiertoe kun je een deel van het ontwerp 'verpakken' in wat een symbool wordt genoemd. Laten we als voorbeeld een symbool maken voor onze schakeling.

Maak eerst een nieuw blokschema/schemabestand aan zoals in de paragraaf **Het project openen** is beschreven: *File, New* enzovoort. Geef het de naam *Synchronizer. bdf*. Klik vervolgens in het venster *Elektor*-*FirstTop.bdf* met de muis en houd de muisknop ingedrukt om alle componenten in een rechthoek te selecteren. Kopieer en plak dan de inhoud in het lege *Synchronizer. bdf* en sla dat op. Ga vervolgens, terwijl het venster *Synchronizer.bdf* nog steeds geopend is, naar File, kies Create/Update, selecteer Create Symbol Files for Current File en klik op Save (met de door het programma gesuggereerde naam Synchronizer.bsf). Klik op OK in het pop-up venster.

Er is nu een Symbool voor een DFF aangemaakt met de naam Synchronizer.bsf. Je kunt het Synchronizer-venster nu sluiten. In ElektorFirstTop.bdf kun je een beetje experimenteren en dit nieuw gemaakte symbool invoegen. Verwijder de D-flipflop, verplaats de ingangen en uitgangen naar de zijkant van het blad om ruimte te maken en klik met de rechtermuisknop in de lege ruimte om het symbool in te voegen zoals we hebben gedaan in de paragraaf **Onderdelen toevoegen aan** ons eerste schema. Het nieuw aangemaakte Synchronizer-symbool is beschikbaar in het menu Insert Symbol. Nu kun je het symbool verbinden met de juiste inen uitgangen en het geheel weer opslaan. Dit hiërarchische ontwerp kan nu opnieuw worden gecompileerd.

Hardware-beschrijvingstalen gebruiken

De meeste mensen gebruiken liever een HDL, zoals Verilog of VHDL, dan een blokschema. Er zijn veel hulpmiddelen beschikbaar ter ondersteuning van deze talen. Bovendien wordt het onderhouden en bijwerken van schematische ontwerpen een ware uitdaging naarmate ze groter worden.

Laten we daarom Synchronizer.bdf omzetten in een Verilog-bestand. Klik op Synchronizer. *bdf*, ga naar File en dan naar Create/Update; kies daar Create HDL Design File from Current File. Selecteer Verilog HDL en klik op OK. Nu zou er een Synchronizer.v bestand in je ontwerpdirectory moeten verschijnen. Je kunt dit bestand openen met elke teksteditor of met Quartus zelf. Ga naar File, kies Open, zoek Synchronizer.v en open het. Voilà, je hebt nu de Verilog-beschrijving van de synchroniserschakeling voor je. Dat kan heel leerzaam zijn! Natuurlijk is het ook mogelijk om een VHDL-bestand te maken. Je moet nu weer een symbool maken voor dit nieuwe Synchronizer.v-bestand. Volg dezelfde stappen die zijn beschreven in de paragraaf **Hiërarchisch ontwerp** en gebruik de opdracht Create Symbol Files for Current File.



Figuur 3. Quartus Lite Edition, met het voorbeeld running_led_with74XXs.



Figuur 4. Quartus Lite Edition, met enkele lower-level implementaties van het running_led_with74XXs voorbeeld.



Eigen experimenten

Voel je vrij om nu zelf te gaan experimenteren en onderzoeken. Klik in *ElectorFirst-Top.bdf* met de rechter muisknop op het schemablad, kies *Insert Symbol* en klik op het kleine driehoekje naast het onderste item. Duik in de primitieven, dan in de logica en je zult een reeks logische basiscomponenten ontdekken zoals AND, OR, XOR enzovoort. Ga je gang en ontwerp je eigen schakeling!

Om de logische circuits van de 7400-serie in de bibliotheek te vinden, kun je ook met de rechtermuisknop op het schemablad in ElectorFirstTop.bdf klikken, selecteer je Insert Symbol en typ je "7400" in het Name-veld. Er is een brede selectie IC's uit de 7400-serie te vinden. Blader een beetje rond en bekijk een voorbeeld van de symbolen in het rechter venster. Er zijn veel intrigerende logische schakelingen die gebruik maken van 7400-serie online beschikbaar, die je kunt emuleren op een FPGA. Het is misschien niet de meest efficiënte methode, maar het kan zeker een interessante en leerzame ervaring zijn! In **figuur 3** zie je een voorbeeld van een ontwerp dat gebruik maakt van 74xx-logica. Je kunt ook het complete ontwerp in de FPGA bekijken. Kies in de bovenste balk Tools, Netlist viewers en RTL viewer. Het resultaat zie je in **figuur 4**. De schema's zijn linksboven zichtbaar. Om de geplaatste elementen binnen de FPGA te onderzoeken, kies je Tools en Chip planner. Hier kun je inzoomen op het gebied met de logische cellen van het ontwerp. Bovendien is in de afbeelding rechtsboven een kleine paarse cel gemarkeerd. Door een logische cel te selecteren en erop te klikken, kun je de inhoud bekijken, zoals linksonder.

En verder...

We hebben slechts een tipje van de sluier opgelicht over FPGA-ontwerp met de gratis-versie Quartus Prime Lite. Deze software biedt een complex scala aan mogelijkheden, waarvan we er veel niet konden behandelen in deze inleiding, zoals simulatie, de Signal Tap logic analyzer, Platform Designer en meer. Er is geen tekort aan geschikte FPGA-boards voor beginners om mee te experimenteren. Voor liefhebbers die naar open-source neigen, bieden het Icestorm-project en de grafische Icestudio-IDE hiertoe opmerkelijke instapmogelijkheden.

Bovendien is er een overvloed aan online-materiaal om verder te leren. Websites zoals fpga4fun [3], NandLand [4] en VHDLwhiz [5] zijn uitstekende uitgangspunten. Joel's overzicht van betaalbare FPGA-boards [6] kan ook helpen bij het kiezen van de juiste hardware.

En wat boeken betreft, de digitale versie van Free Range VHDL van Bryan Mealy en Fabrizio Tappero wordt volledig gratis aangeboden door de auteurs en *Getting Started with FPGA* van Russel Merrick (de auteur van de NandLand-tutorials) is ook een goede aanvulling voor je boekenplank. De FPGA-wereld ligt binnen handbereik, met meer tutorials en communities dan ooit tevoren. Dit is een opwindende tijd om je met digitaal ontwerpen te gaan bezighouden!

230067-03



Over de auteur

Theo Mulder is elektrotechnisch ingenieur. Hij is bekend met signaalverwerking en heeft gewerkt met analoge en digitale elektronica, vooral in de medische elektronica voor echo-apparatuur. Hij is vertrouwd met C++, DSP's en FPGA's. Hij zet onderzoeksconcepten om in werkende elektronica. Sommige van zijn eigen concepten zijn gepatenteerd. Hij is een systeemdenker, op print-niveau, om een goede balans te vinden tussen analoge/ digitale hardware en software, om goed werkende systemen te realiseren met goede prijs/prestatie-verhoudingen.

Vragen of opmerkingen?

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



- M. Dalrymple, Microprocessor Design Using Verilog HDL (E-book, Elektor) www.elektor.nl/18518
- > Alchitry Au FPGA Development Board (Xilinx Artix 7) www.elektor.nl/19641

WEBLINKS

- [1] Trenz Electronic CYC1000 board: https://tinyurl.com/trenzcyc1000
- [2] Intel Quartus Prime Design Software: https://tinyurl.com/downloadquartus
- [3] fpga4fun: http://fpga4fun.com
- [4] NandLand: https://nandland.com
- [5] VHDLwhiz: https://vhdlwhiz.com
- [6] Joel's list of FPGA boards: https://joelw.id.au/FPGA/CheapFPGADevelopmentBoards

UPDATE: STM32 Wireless Innovation Design Contest 2024

Het Elektor Content Team

In de STM32 Wireless Innovation Design Contest 2024 ligt € 5.000 aan geldprijzen voor het grijpen. Vernieuwers van over de hele wereld hebben de afgelopen maanden hard gewerkt met STM32-oplossingen om een verscheidenheid aan creatieve draadloze toepassingen te ontwikkelen. De genomineerden voor de prijzen worden in maart 2024 bekendgemaakt. De uiteindelijke bekendmaking van de winnaars vindt plaats op embedded world 2024.

De STM32 Wireless Innovation Design Contest biedt engineers en makers een unieke kans om hun ontwerpvaardigheden te tonen door draadloze toepassingen te creëren met behulp van de krachtige ontwikkel- en evaluatieboards van STMicroelectronics. Of je nu gepassioneerd bent door IoT, robotica, gaming, domotica of AI, de mogelijkheden zijn eindeloos. € 5.000 aan geldprijzen ligt voor het grijpen!

Blijf op de hoogte!

De uiterste inzenddatum voor de wedstrijd was 1 maart 2024. Op het moment van publicatie van dit nummer is de jury druk bezig met de beoordeling van de inzendingen. De winnaars worden live bekendgemaakt tijdens de embedded world 2024-conferentie in Neurenberg (Duitsland) om 17:00 CEST op woensdag 10 april, en ook online op *elektormagazine.com/st-contest*. Als je van plan bent de embedded world 2024 bij te wonen, ga dan zeker langs bij de Elektor- en STMicroelectronics-stand (embedded-world.de).

Beoordeling

Een panel van onafhankelijke juryleden zal de top-drie winnaars kiezen op basis van de volgende criteria:

- Creativiteit en innovatie: uniciteit en originaliteit van het ontwerp van de draadloze toepassing.
- Technische uitmuntendheid: technische bekwaamheid en vaardigheid bij het gebruik van het gekozen development board.
- Functionaliteit en bruikbaarheid: effectiviteit en bruikbaarheid van de draadloze toepassing bij het oplossen van een echt probleem of het verbeteren van de gebruikerservaring.
- Esthetiek en gebruikerservaring: de visuele aantrekkingskracht, het ontwerp van de gebruikersinterface en de algemene gebruikerservaring van de applicatie.

 Documentatie en presentatie: de duidelijkheid, volledigheid en kwaliteit van de projectdocumentatie en -presentatie.

We wensen alle deelnemers veel succes!

STM32-technologie

Om mee te doen aan de wedstrijd moeten deelnemers een project maken met een van de aangeboden borden: NUCLEO-WBA52CG, STM32WB5MM-DK of NUCLEO-WL55JC. Het idee is om de mogelijkheden van een bord te benutten om draadloze toepassingen op elk gebied te ontwerpen en te ontwikkelen. Deelnemers kunnen gestandaardiseerde protocollen verkennen, zoals LoRaWAN, Sigfox en Bluetooth Low Energy (BLE), of ze kunnen hun eigen protocol maken.

De NUCLEO-WBA52CG is een draadloos Bluetooth Low Energy en ultra-low-power board met een krachtige en ultra-low-power radio die voldoet aan de Bluetooth Low Energy SIG specificatie v5.3. De ondersteuning voor ARDUINO Uno V3 connectiviteit en de ST



morpho headers maken eenvoudige uitbreiding van de functionaliteit van het STM32 Nucleo open development platform mogelijk met een ruime keuze aan gespecialiseerde shields.

De STM32WB5MM-DK Discovery Kit is een demo- en development platform voor de STM32W5MMG-module van STMicro-

electronics. Dit dual-core 32-bit Arm Cortex-M4/M0+ board integreert een ultralow-power radio (compatibel met Bluetooth Low Energy (BLE) 5.2, 802.15.4) met Zigbee, Thread en eigen protocollen.



Het Nucleo-WL55JC board is een evaluatieboard voor de STM32WL-serie microcontrollers, en in het bijzonder de STM32WL55. Deze zogenaamde sub-GHz draadloze microcontroller is gebaseerd op een dual-core 32-bit Arm Cortex-M4/ M0+ met een klokfrequentie van 48 MHz. Hij heeft een ultralaag stroomverbruik, een RF-transceiver met een frequentiebereik van 150 MHz tot 960 MHz, 256 KB flashgeheugen en 64 KB SRAM.

Op zoek naar meer informatie? Bezoek de STM32 Wireless Innovation Design Contest website – *elektormagazine.com/st-contest* – voor details over de winnende projecten en nog veel meer.

240010-03



Bluetooth LE met MAUI

besturingsapps voor Android en co.

Tam Hanna (Hongarije)

Een smartphone-app is ideaal voor het besturen van je eigen elektronica. Communicatie via Bluetooth-LE bespaart daarbij energie. Maar het ontwikkelen en onderhouden van software voor Android en iOS kost wat inspanning. Hier komt het MAUI-framework, dat we al eerder hebben gepresenteerd in Elektor, in het spel. Daarmee kunnen we apps onafhankelijk van het platform ontwikkelen. In dit artikel laten we zien hoe je de BLE-interface van een smartphone kunt gebruiken.

Vooral wie veel ervaring heeft met .NET-ontwikkeling, zal in MAUI een gemakkelijke manier vinden om software te ontwikkelen voor Android en iOS. Een algemene inleiding is te vinden in een ander artikel [1]. In dit artikel gaan we wat dieper op de zaak in, door een Bluetooth-interface op Android te programmeren. De hier gebruikte library werkt ook op iOS, maar de aanpassingen die daarvoor nodig zijn vallen buiten het kader van dit artikel, daarom gaan we daar niet nader op in.

Opzetten van de werkomgeving

Dit artikel is gebaseerd op een consultancy-project uit de praktijk van de auteur, die als freelance-ontwikkelaar werkt. Een ESP32 dient als het 'andere station' en voert een besturingsprogramma uit dat is gebaseerd op het codevoorbeeld [2]. We gebruiken Visual Studio 2022 voor het ontwikkelen van de smartphone-toepassing; het eigenlijke voorbeeldproject is gebaseerd op het .NET MAUI app-sjabloon. Microsoft heeft lang geleden besloten om geen 'directe' Bluetooth-API te bieden in MAUI (of in Xamarin, waar MAUI op is gebaseerd). Dat was een wijs besluit; waarschijnlijk had Microsoft al ervaring met het ontwik-



De BopSync-asbak, ontwikkeld door Icy Beats LLC, gebruikt een Bluetooth LE-interface voor de besturing.

kelen van Bluetooth API's onder Android, wat bepaald niet 'gladjes' verloopt. Maar omdat er een interface bestaat waarmee Xamarin native elementen van het besturingssysteem op de host kan benaderen, kunnen verschillende NuGet-packages worden gebruikt met libraries die dat gemis proberen op te lossen. Hieronder maken we gebruik van de *Plugin.Ble*-library die te vinden is op [3]. Eerst openen we het NuGet-console, om de ontbrekende library te downloaden.

Aanpassen van het manifest-bestand

Cross-platform-omgevingen kunnen de ontwikkelaar natuurlijk maar ten dele isoleren van het onderliggende platform. Bij de hier gebruikte library zien we dat aan het feit dat aanpassingen aan de respectieve manifest-bestanden nodig zijn voor zowel Android als iOS (waar we hier niet verder op in gaan). Die vertellen de besturingssystemen welke mogelijkheden de toepassing wil gaan gebruiken.

Bij Visual Studio versie 17.6.0 Preview, dat we hier gebruiken, wordt normaal gesproken een grafische editor geopend als we op *AndroidManifest.xml* klikken, maar die is nog niet volledig uitontwikkeld. Daarom klikken we er gemakshalve met de rechter muisknop op en openen we het bestand in een teksteditor.

Vervolgens heeft de auteur enkele structuren vervangen en/of toegevoegd in de bestaande declaraties (**listing 1**).

Vanwege het 'dynamic permission system' in Android 6.0 volstaat het niet langer om het manifest-bestand hier te veranderen. In plaats



daarvan moet de toepassing de gebruiker actief om toestemming vragen voor het gebruik van de gevoelige resources.

Om conflicten tussen de MAUI-runtime en Google's beheer van nieuwe permissies te voorkomen, implementeert Microsoft een 'algemene' interface voor het verkrijgen van permissies. In principe is dat een class-structuur die een lijst van permissie-constanten bevat. Om een nieuw permissie-attribuut te gebruiken, moet de ontwikkelaar de constante inpakken in een instantie van Permissions.

BasePlatformPermission. De rest van de interactie met de gebruikersinterface volgt een gestandaardiseerde procedure die wordt verzorgd door de runtime in MAUI.

We moeten dus nu een nieuwe klasse maken van het type Permissions.BasePlatformPermission, die de benodigde permissies bevat (listing 2).

Opmerking: deze permissieselectie werkt met Android 13. Oudere versies en de Kindle Fire gebruiken een ander protocol.

.....

Listing 1. Declaraties.

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android">
              <application android:allowBackup="true" android:icon="@mipmap/appicon" android:roundIcon</pre>
               ="@mipmap/appicon_round" android:supportsRtl="true"></application>
              <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
              <uses-permission android:name="android.permission.INTERNET" />
              <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
              <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
              <uses-permission android:name="android.permission.BLUETOOTH" />
              <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
              <uses-permission android:name="android.permission.BLUETOOTH_SCAN" />
              <uses-permission android:name="android.permission.BLUETOOTH_CONNECT" />
              <uses-permission android:name="android.permission.BLUETOOTH ADVERTISE" />
              <uses-feature android:name="android.hardware.bluetooth_le" android:required="true"/>
```

</manifest>

Listing 2. Een nieuwe klasse van het Type Permissions.BasePlatformPermission maken.

```
public class BluetoothLEPermissions : Permissions.BasePlatformPermission
 public override (string androidPermission, bool isRuntime)[] RequiredPermissions
  {
   get
   {
    return new List<(string androidPermission, bool isRuntime)>
   {
     (Manifest.Permission.Bluetooth, true),
     (Manifest.Permission.BluetoothAdmin, true),
     (Manifest.Permission.BluetoothScan, true),
    (Manifest.Permission.BluetoothConnect, true),
    (Manifest.Permission.AccessFineLocation, true),
    (Manifest.Permission.AccessCoarseLocation, true),
        //(Manifest.Permission.AccessBackgroundLocation, true),
   }.ToArray();
}
```



Scannen naar BLE-apparaten

Nu kunnen we gaan zoeken naar bereikbare Bluetooth-apparaten. De auteur gaat er van uit dat de lezer min of meer bekend is met het gebruik van Bluetooth-LE. Een korte introductie vind je bij [4]. Om alle elementen in de omgeving weer te geven, gebruikt de auteur een ListView in de volgende stappen: open *MainPage.xaml* en voeg de volgende *ListView* toe aan de layout:

```
<ListView x:Name="deviceList">
```

```
<ListView.ItemTemplate>

<DataTemplate>

<ViewCell>

<StackLayout Margin="20,0,0,0"

Orientation="Horizontal"

HorizontalOptions="FillAndExpand">

<Label Text="{Binding}"

VerticalOptions="Start"

TextColor="White"/>

</StackLayout>

</DataTemplate>

</ListView.ItemTemplate>

</ListView>
```

Ontwikkelaars die zijn opgegroeid met Visual Basic 6 en co. zullen even moeten wennen aan het werken met MAUI list boxes. Een MAUI list box bestaat niet alleen uit de verzameling van weer te geven elementen: er moet ook extra informatie in de vorm van een ItemTemplate worden meegegeven. Dat is een stukje XAML-markup dat de GUI-stack gebruikt voor elke rij data die moet worden weergegeven in de ListView. Onze instantie is eenvoudig, want hij plaatst eigenlijk alleen een label op het scherm. De string Text="" legt een relatie vast waarmee de XAML-parser rechtstreeks de in het item weer te geven elementen kan ophalen.

Er moet ook een button in de XAML-markup komen, waarmee de gebruiker het scannen kan starten.

Bij de volgende stap kunnen we terugkeren naar de Code Behind, waar we enkele ondersteunende klassen maken volgens dit schema:

```
public partial class MainPage : ContentPage
{
    int count = 0;
    IBluetoothLE myBLE = CrossBluetoothLE.Current;
    IAdapter myAdapter =
        CrossBluetoothLE.Current.Adapter;
    public ObservableCollection<String> BTLEDevices;
```

Naast de variabelen voor de library is ook een ObservableCollection nodig. Die werkt als een 'databron' voor het vullen van de ListView. Er is niet zomaar gekozen voor het gebruik van een ObservableCollection: het is een Collection-klasse die notificaties kan zenden als er veranderingen optreden in de gekoppelde database.

Om de datarelatie compleet te maken, zijn nog wat organisatorische werkzaamheden in de constructor van de hoofdpagina nodig:

```
public MainPage() {
    InitializeComponent();
    BTLEDevices = new ObservableCollection<String>();
    deviceList.ItemsSource = BTLEDevices;
}
```

Nu zijn we toe aan de Code Behind-methode, die het klikken op de scan-knop afhandelt. Android werkt met een permissie-cache. Daarom beginnen we met controleren of onze toepassing al de nodige autorisaties heeft voor het werken met de Bluetooth-zender. Zo ja, dan roepen we de goScan-methode aan, die het feitelijke scannen afhandelt.

```
private async void OnScanClicked
  (object sender, EventArgs e) {
   PermissionStatus status = await
    Permissions.CheckStatusAsync
    <BopSyncNetPOC.Platforms.
    Android.BluetoothLEPermissions>();
   if (status == PermissionStatus.Granted){
     goScan();
   }
```

Als uit de retourwaarde van CheckStatusAsync blijkt dat we nog niet de benodigde permissies hebben, vragen we de gebruiker om toestemming:

```
else {
 await DisplayAlert("Permission required",
         "Google requires the declaration
         of this permission to perform a BTLE scan.
         Please grant it!", "OK");
 status = await
       Permissions.RequestAsync
          <BopSyncNetPOC.Platforms.
            Android.BluetoothLEPermissions>();
 if (status == PermissionStatus.Granted) {
     goScan();
 }
 else {
   await DisplayAlert("Alert",
         "Permission denied.
          Please reinstall application!", "OK");
  3
}
```

De uitgebreide toelichting bij de vraag om toestemming moet de gebruiker overtuigen om akkoord te gaan. Praktische ervaring heeft aangetoond dat vooral 'technisch minder onderlegde' gebruikers meestal negatief reageren op onverwacht opduikende verzoeken om toestemming, omdat ze afschrikwekkende verhalen hebben gehoord in de media. Bovendien slaan moderne Android-versies zo'n weigering soms 'permanent' op: als de gebruiker één keer op No heeft geklikt,

}

kan het dan nodig zijn om de app opnieuw te installeren, voordat alsnog toestemming kan worden gegeven. Hoe dan ook, de volgende functie is de goScan-methode:

```
private void goScan() {
    if (myBLE.State == BluetoothState.On) {
        myAdapter.ScanMode = ScanMode.LowLatency;
        myAdapter.DeviceDiscovered += FoundDevice;
        myAdapter.ScanTimeout = 12000;
        //MUST be called last or ignores settings
        myAdapter.StartScanningForDevicesAsync();
    }
```

Als de Bluetooth-zender al actief is, parameteriseren we het adapter-object en starten een scan met een aanroep van StartScanningForDevicesAsync. Zorg ervoor dat alle nodige instellingen voor de scan naar de class moeten zijn geschreven voordat de method wordt geroepen; latere veranderingen hebben geen invloed meer.

Als de Bluetooth-zender niet actief is, tonen we de gebruiker een venster dat hem oproept om de zender in te schakelen:

```
else {
  DisplayAlert("Alert",
        "Please switch Bluetooth on!", "OK");
  }
}
```

De delegate zorgt voor de 'lokalisatie' van apparaten. Wij concentreren ons op het vullen van de lijst:

De meeste Bluetooth LE-apparaten hebben geen naam. De hier getoonde routine gebruikt een try/catch-blok om dan een 'primitievere' default-string in de ListView te schrijven.

Het programma is nu klaar voor een eerste test. In **figuur 1** zie je hoe een scan eruit ziet.

Overigens leidt de rechtstreekse plaatsing van native elementen in het algemene deel van de Solution tot vreemde effecten: er zullen specifiek fouten optreden die wijzen op het niet bestaan van de namespace



Figuur 1. De Bluetooth LE-detectie werkt.

Platforms. Android. Dat komt doordat sommige acties in Visual Studio proberen om niet alleen een Android-variant te compileren, maar ook een versie voor verschillende andere doelplatformen in het MAUI-projectframework. We kunnen dat probleem omzeilen door een preprocessor-bescherming rond de platformspecifieke code te plaatsen:

```
private async void OnScanClicked
        (object sender, EventArgs e) {
    #if ANDROID
    PermissionStatus status . . .
    #endif
}
```



}

3

Figuur 2. Het eeuwige gevecht: mens tegen Visual Studio.

Het is wel hinderlijk dat Visual Studio moeite heeft met zulke elementen en de syntax-completion uitschakelt, zoals in **figuur 2**.

Identificatie en opbouw van de verbinding

Nu kunnen we de besturing van ons apparaat ter hand gaan nemen. De auteur heeft de hoofdpagina uitgebreid met een extra lijst met de individuele Bluetooth-apparaatinstanties. Dat zijn de stations die gevonden zijn door de Bluetooth LE-scan.

Bij de toepassing van de auteur hoeft slechts één apparaat tegelijk geadresseerd te worden. Daarom is het zinvol om de instantie vast te leggen in de klasse Application. Open *App.xaml.cs* en voeg een globale member toe:

```
public partial class App : Application
{
    public Plugin.BLE.Abstractions.
```

Contracts.IService myBTLEService;

De kwalificatie 'complete' is redelijk omdat er vaak meer dan één klasse met de naam IService is in de .NET-wereld. Door 'compleet' te declareren, blijf je aan de veilige kant.

Het eigenlijke opzetten van de verbinding begint door de IDeviceklasse uit de hiervoor genoemde lijst van apparaten te halen:

```
private async void deviceList_ItemSelected
  (object sender, SelectedItemChangedEventArgs e)
{
    IDevice myDevice =
    BTLEDeviceClasses[e.SelectedItemIndex];
```

Daarna moeten we een device-instantie en de lijst van alle services verkrijgen:

try {

```
await myAdapter.ConnectToDeviceAsync(myDevice);
var services = await myDevice.GetServicesAsync();
```

```
services = services;
foreach (IService serv in services) {
  String aString = serv.Id.ToString();
  if (aString.CompareTo
    ("000000ff-0000-1000-8000-00805f9b34fb") == 0)
  { //Swap view
    App curApp = (App) Application.Current;
    curApp.myBTLEService = serv;
    await Navigation.
    PushModalAsync(new MainWorkPage()) ;
  }
}
```

In dit geval heeft de auteur besloten het apparaat te identificeren door de beschikbaarheid van een specifieke service te controleren. Als een service met de vereiste GUI is gevonden, wordt de weergegeven huidige activiteit veranderd.

Het is belangrijk de methode Navigation.Pushmodalasync te gebruiken. Dat komt doordat de auteur in zijn commerciële toepassing gebruik maakt van een van [5] afgeleide pagina. Gebruik van de normale method PushAsync, zoals aanbevolen door Microsoft, leidt hier tot vreemde crashes bij een venster dat op deze manier is opgebouwd. Het try/catch-blok is nodig omdat het beschermt tegen communicatieproblemen:

```
catch (DeviceConnectionException ex)
{
   // ... could not connect to device
}
```

Nu moet de toepassing gaan communiceren met de karakteristieken. Om de code in de views zoveel mogelijk te beperken, maakt de auteur gebruik van de structuur in de flowchart in **figuur 3**.

Als je de communicatiefunctie in een (bij voorkeur statische) klasse opneemt, hoef je hem niet te definiëren in alle individuele pagina's.



Figuur 3. Isoleren van de communicatielogica om doublures in de code te voorkomen.

De auteur heeft weer besloten om gebruik te maken van de klasse App in zijn toepassing. Hier volgt de declaratie:

```
public async void setLightMode(int _what)
{
    var x = await myBTLEService.
    GetCharacteristicAsync(Guid.Parse
        ("0000ff01-0000-1000-8000-00805f9b34fb"));
```

Om te beginnen gebruiken we de methode GetCharacteristicAsync, die een specifieke karakteristiek selecteert op basis van zijn GUID. Omdat het hier alleen gaat om een 'identificatie', heeft de auteur de waarde van x hier niet beschermd; in de praktijk zou het natuurlijk redelijk zijn om dat te doen.

De library zou trouwens ook een 'globale' zoekactie kunnen gebruiken. De code daarvoor zou er als volgt uitzien:

```
public async void setLightMode(int _what) {
  var ibx = await myBTLEService.GetCharacteristicsAsync();
  foreach (var chara in ibx) {
    var str = chara.Id.ToString();
    str = str;
  }
....
```

De tautologische instructie in deze code is nuttig omdat we daarmee GUI's van alle individuele eigenschappen kunnen 'oogsten'. Als je de GUI van de karakteristiek niet uit de ESP32-code kunt halen in de vorm van een string, kun je hier een breakpoint zetten bij de tautologische instructie en het programma daarachter uitvoeren.

Bij elke run kun je dan de respectieve karakteristieken in de debugger extraheren, zoals in **figuur 4**, en die (indien nodig) vergelijken met de informatie in de Bluetooth-scanner.

		4					
	1	_	_ 19	Verweise			
	0	5	Epu	IDLIC PARTIAL CLASS APP : Application			
		6	1	nublic Divers DLE Abstractions Cont	Text-Schnellansicht		×
		.7		public Plugin.BLE.Abstractions.com	rext-sermenaristene		
		8		private bool myControlFlag = false;	Ausdruck:	str.str	
		0		verweise	Zeichenfolgenmanipulation:	Keine	 -
		9	T:	s	Wert:		
		10			0000ff01-0000-1000-800	0-00805f9b34fb	^
		11		InitializeComponent();			
		12					
		13		<pre>MainPage = new AppSnell();</pre>			
		14		3			
		15	L	5 verweise			
		10		s section as a section of the sectio			
		10		l			
		17		Var IDX - await mybileService.			
		18	무글	foreach (Var chara in iDX) {			
		19		var str = chara.ld.lostring			
		20 🖞		str = str;			
		21		}			
		22		<pre>var x = await myBTLEService.Get</pre>)-80
		23		<pre>byte[] myChar = new byte[] { (t</pre>			
		24	Ē.	switch (_what)			
		25		ł			
150.00	- @	26		case 0:			
150 %	* *	0	A 2		Zeilenumbruch		
Auto						- T A Autoniste	

Figuur 4. Tautologische instructies besparen typewerk!



Bluetooth-scanners zijn handig

ledereen die een Bluetooth LE-toepassing wil ontwikkelen, zou een Bluetooth LE-scanner op zijn Android moeten installeren. Dat is een app die verbinding maakt met alle beschikbare apparaten in de buurt en al hun diensten en karakteristieken in een lijst weergeeft. De auteur gebruikt de nRF Connect-app, die gratis kan worden gedownload van de Play Store [7].

De scanner moet draaien op Android of op een oude iPhone met iOS 15. iOS16 heeft een gewijzigde Bluetooth-stack, waardoor minder devices zichtbaar zijn en de functionaliteit van de scanner beperkt wordt!

Nu moeten we de payload gaan samenstellen. Bij de auteur wordt een communicatieprotocol gebruikt op basis van streams van ASCII-waarden. De payload wordt dan opgebouwd met arrayconstanten op basis van de numerieke waarde:

```
byte[] myChar = new byte[] { (byte)'0' };
switch (_what)
  {
  case 0:
    myChar= new byte[] { (byte)'0' };
    break:
  case 1:
    myChar = new byte[] { (byte)'1' };
    break;
  case 2:
    myChar = new byte[] { (byte)'2' };
    break:
  case 3:
    myChar = new byte[] { (byte)'3' };
    break:
  case 4:
    myChar = new byte[] { (byte)'4' };
    break;
}
```

En tenslotte is een commando nodig om de verbinding op te bouwen, en dat heeft deze structuur:

var y = await x.WriteAsync(myChar);
}

Data van de ESP32 naar de Android-telefoon sturen

Tenslotte willen we data van de ESP32 via de draadloze interface naar de telefoon sturen. De auteur wil hier iets dieper ingaan op de ESP-IDFcode. Hij vindt de voorbeelden gebaseerd op het GATT-tabelsysteem erbarmelijk gedocumenteerd en op het forum zien we verwarde en maar ten dele beantwoorde vragen.

De belangrijkste vraag in dit verband is hoe de karakteristieken

worden geïnitialiseerd. Als de constante ESP_GATT_AUTO_RSP wordt doorgegeven, zoals standaard gespecificeerd in het voorbeeld, handelt de stack de waarden in de karakteristiek af.

De applicatie ontvangt ook een read-event volgens het onderstaande schema, maar de Bluetooth-stack haalt de retourwaarden uit een intern geheugen, en stuurt die normaal gesproken zelfs op, vóór de activering van het event:

case ESP_GATTS_READ_EVT: ESP_LOGI(GATTS_TABLE_TAG, "ESP_GATTS_READ_EVT"); break;

In theorie bevat de methode esp_ble_gatts_set_attr_value een functie waarmee de geheugenwaarden in de Bluetooth-stack kunnen worden opgeslagen om ze aan te passen. Een naieve implementatie zou er zo uitzien:

```
void updateBTLECache() {
    esp_err_t ret;
    ret = esp_ble_gatts_set_attr_value
        (heart_rate_handle_table[IDX_CHAR_VAL_A],
            1, (const uint8_t *)"1");
}
```

Het probleem met deze methode is dat hij een nulwaarde retourneert, zelfs als de meegegeven handle ongeldig is of niet is geïnitialiseerd door de Bluetooth-stack.

En dat is meteen het probleem: het invullen van het handle-array gebeurt tamelijk laat. Een manier die volgens het bedrijf van de auteur goed werkt, is de waarden in de BTLE-cache telkens bij te werken als een nieuw apparaat met de ESP32 verbinding maakt. De volgende code is daar ideaal voor:

Ter afsluiting volgt hier een codefragment voor het inlezen van de waarden aan de MAUI-kant:

De waarden, die binnenkomen in de vorm van een byte-array, kunnen dan naar eigen inzicht verder worden verwerkt.
Sneller ontwikkelen

Onze experimenten tonen aan dat MAUI's Bluetooth LE-API probleemloos met andere doelplatforms kan communiceren. Als je een besturingsapp voor je embedded systeem nodig hebt en die bouwt met MAUI, kun je programmeren in C# of in Visual Basic zonder teveel gedoe met afwijkende platforms. Zeker als je bedrijf al ervaring heeft met het .NET-framework, kan dat een aanzienlijke versnelling van het ontwikkelproces opleveren. Natuurlijk is een Bluetooth LE-programma op een smartphone maar de helft van de oplossing. In een eerder gepubliceerd artikel [6] laat de auteur zien hoe je een STM32 kunt programmeren met een BLE-interface.

vertaling: Evelien Snel — 230381-03

Vragen of opmerkingen?

Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een email naar de auteur via tamhan@tamoggemon.com of neem contact op met Elektor via editor@elektor.com.

Over de auteur

Tam Hanna werkt al meer dan 20 jaar als ingenieur met elektronica, computers en software. Hij is een zelfstandig ontwerper, auteur en journalist (@tam.hanna op Instagram). In zijn vrije tijd ontwerpt en produceert hij 3D-geprinte oplossingen en hij is een liefhebber van goede sigaren.



WEBLINKS

- [1] V. Krypczyk, "MAUI: programmeren voor PC, tablet en smartphone," Elektor januari/februari 2024: http://www.elektormagazine.nl/magazine/elektor-326/62589
- [2] GATT Server Service Table: https://bit.ly/45KuplU
- [3] Plugin.Ble-library: https://github.com/dotnet-bluetooth-le/dotnet-bluetooth-le
- [4] Een korte introductie tot Bluetooth Low Energy (BLE): https://developer.android.com/develop/connectivity/bluetooth/ble/ble-overview
- [5] Navigation.PushModalAsync-methode: https://github.com/dotnet/maui-samples/tree/main/8.0/Navigation/FlyoutPageSample
- [6] T. Hanna, "Bluetooth LE oP De STM32," Elektor januari/februari 2024: http://www.elektormagazine.nl/magazine/elektor-326/62607
- [7] nRF Connect-applicatie: https://play.google.com/store/apps/details?id=no.nordicsemi.android.mcp

Ontmoet Elektor op embeddedworld 2024 Exhibition&Conference

Praat met onze redacteuren, het client team en marketingexperts. Studenten, kom langs voor een praatje en blijf voor de goodies.

Laat ons dit bericht zien als je ons bezoekt en ontvang een leuke verrassing!



Port Expander **breakout-board**

meer I/O's op je development board

Alessandro Sottocornola (Italië)



Ben je het zat met compromissen te moeten leven omdat je bij het ontwikkelen van je ontwerpen met het platform van je voorkeur I/Opoorten te kort komt? Wil je er meer hebben om een groot aantal signalen en actuatoren te kunnen verwerken? Met dit breakout-board, dat via de I²C-bus wordt aangestuurd, kun je tot 16 I/O's per unit toevoegen tot je een maximum van 128 I/O's op je bestaande systeem hebt.

Er zijn verschillende situaties denkbaar waarin het nodig is om meerdere signalen en actuatoren in te lezen of aan te sturen met een microcontroller die niet genoeg I/O-pinnen heeft. In zulke gevallen kan het nuttig zijn om gebruik te maken van IC's die bekend staan als 'I/O expanders' en die als functie hebben om een aantal inen uitgangslijnen toe te voegen en deze aan te sturen via een seriële I²C- of SPI-verbinding.

Om die behoefte aan meer I/O-lijnen te bevredigen, hebben we een breakout-board ontworpen op basis van de Microchip MCP23017 [1], een 16-bit I/O-expander. Daarnaast hebben we de minimale hardware op het board gezet die nodig is om het ook te kunnen gebruiken: in dit geval een DIP-schakelaar en wat pull-up-weerstanden. Om je een idee te geven van wat je kunt bereiken met een I/O-expander, presenteren we een toepassingsvoorbeeld waarbij het breakout-board een reeks van acht relais aanstuurt die op een print zijn gemonteerd en die kunnen worden aangestuurd via TTL-niveaus of door op knoppen te drukken. Maar alles op zijn tijd – eerst kijken we hoe deze chip werkt.

De MCP23X17

Deze veelzijdige geïntegreerde schakeling biedt een generieke 16-bit seriële/parallelle I/O-uitbreiding en is verkrijgbaar in twee versies: de hier gebruikte MCP23017, uitgerust met een I²C-interface, en de MCP23S17, een SPI-variant. De chip is een 16-bit I/O-expander, verdeeld in twee poorten van elk 8 bit, met een interface via de I²C-bus. Dit betekent dat het met slechts twee lijnen (ten opzichte van massa) mogelijk is om de status van maar liefst 16 lijnen op te vragen (ingangsmodus) of de logische status van elk daarvan in te stellen (uitgangsmodus). De I/O-lijnen werken standaard als ingangen. De MCP23017 bestaat uit meerdere 8-bit-registers voor ingangs-, uitgangs- en polariteitsselectie. De systeem-master kan de I/O's activeren als in- of uitgangen door de corresponderende I/O-configuratiebits (IODIRA/B) te schrijven. Data voor elke ingang of uitgang wordt opgeslagen in het corresponderende ingangs- of uitgangsregister. De polariteit van het Input Port-register kan worden omgekeerd met behulp van het Polarity Inversion-register. Alle registers kunnen worden gelezen vanuit de systeemmaster. De 16-bit I/O-poort is structureel samengesteld uit twee 8-bit-poorten - namelijk poort A en poort B met hun respectieve pinnen 21...28 en 1...8. De MCP23X17 kan worden geconfigureerd in 8-bitof 16-bit-modus. Verder heeft hij twee interrupt-pinnen, INTA en INTB, die op twee manieren kunnen worden toegewezen:

- de interruptpinnen werken onafhankelijk van elkaar. INTA weerspiegelt interruptcondities op poort A en INTB doet hetzelfde voor poort B;
- > beide interruptpinnen worden actief wanneer een interrupt optreedt op een van beide poorten.

Schema

Zoals je in het schema van **figuur 1** ziet, is het breakoutboard uiterst basaal gehouden, met aan boord de MCP23017-chip in DIP-versie; alle pennen zijn toegankelijk via pinheaders (in een 2,54mm-raster om plaatsing op andere printen of breadboards mogelijk te maken). De drie pinnen voor het instellen van de minst significante bits van het perifere I²C-busadres zijn verbonden met een driepolige DIP-schakelaar (SW1 in het schema) om het adres op de I²C-bus in te stellen. De pinnen A0, A1 en A2 worden door weerstanden R1 tot R3 hoog gehouden als de schakelaars open zijn.

De I/O-gerelateerde pinnen van de registers A en B zijn aan weerszijden van het IC naar buiten gevoerd, zodat we het board gemakkelijk kunnen layouten en aansluiten. Op deze digitale I/O-pinnen kun je alles aansluiten wat je maar wilt, binnen de stroom- en spanninggrenzen zoals die voor de MCP23017 gelden. Naast de 2,54mm-raster pinheaders aan de zijkanten zijn er nog vier pinnen op een header met de naam *I2C* naar buiten gevoerd (ook met 2,54mm-raster), zodat we de bus aan de kopse kant van de print kunnen aansluiten; dit biedt meer flexibiliteit. Deze pinnen, waaronder de +5V-voeding en massa, zijn doorverbonden met de corresponderende pinnen op de headers aan de zijkant, die als alternatief kunnen worden gebruikt. Zowel SDA als SCL zijn van pull-up-weerstanden voorzien.

Merk op dat op de print voor het gemak de pinnen van het A-register aan de ene kant zijn geplaatst en de pinnen van het B-register aan de andere kant, om het aansluiten te vereenvoudigen. Zoals gezegd zijn de pinnen die bij de I²C-bus horen ook op de *I*2*C*-pinheader naar buiten gevoerd (voorzien van pull-up-weerstanden), samen met +5 V en massa. De reset van de I/O-expander wordt in ons breakout-board niet gebruikt; om deze uit te schakelen, hebben we de betreffende pin (pin 18, RST) via weerstand R4 hoog getrokken.

Ook de _5V- en masa-aansluitingen zijn op de headers aan de zijkant naar buiten gevoerd. De schakeling als geheel wordt gevoed via een 5V-pin (we hebben eigenlijk twee contacten: pinnen 1 en 24 van de pinheaders aan de lange zijkanten van de print) ten opzichte van massa (GND, de pinnen 2 en 23 van diezelfde pinheaders).

De I/O-expander

Het belangrijkste element van de schakeling is natuurlijk de MCP23017, die we kunnen beschouwen als een I²C/ parallel-converter, vervaardigd door Microchip (gemarkeerd met U1). De geïntegreerde schakeling, waarvan **figuur 2** het interne blokschema toont, functioneert als periferie (Slave) van de I²C-bus en ondersteunt twee ingangs- en uitgangsmodi. In de eerste modus kunnen de I/O-toestanden van de A- en B-registers in serieel formaat (één byte voor elk register) op verzoek van de I²C-busmaster op de bus worden gezet; in de tweede



Figuur 1. Het schema van het breakout-board.



Figuur 2. Blokschema van de MCP23017. De SPI-variant met het typenummer MCP23S17 is ook afgebeeld (bron: Microchip [4]).

De MCP23017 in het kort

De chip idie het hart vormt van het hier beschreven breakout-board is een 16-bit I/O-expander, opgesplitst in twee poorten van elk 8 bit, met I²C-interface. Dit betekent dat met slechts twee lijnen plus gemeenschappelijke massa de status van 16 lijnen kan worden opgevraagd of kan worden ingesteld. De specificaties luiden:

- snelle I²C data-interface (100 kHz, 400 kHz of 1,7 MHz)
- acht verschillende I²C-busadressen instelbaar
- configureerbare interruptpinnen (niveau en logische functie)
- configureerbare interruptbron
- · register voor omgekeerde polariteit

Voor de ingangen geldt:

- externe reset-ingang
- stand-by stroom 1 µA max.
- voedingsspanning 1,8...5,5 V

modus worden de I/O-lijnen ingesteld door de binnenkomende data op de I²C-bus te converteren naar de overeenkomstige toestand van de A- en B-registerlijnen. De interruptuitgang kan worden geconfigureerd om aan te spreken onder twee (elkaar uitsluitende) voorwaarden:

- > wanneer de toestand van een ingang verschilt van de overeenkomstige toestand van het register van de ingangspoort. Deze toestand wordt gebruikt om de systeemmaster aan te geven dat een ingangstoestand is gewijzigd;
- wanneer de toestand van een ingang verschilt van de vooraf geconfigureerde waarde van het register (DEFVAL-register).

De INTA- en INTB-interruptlijnen kunnen worden geconfigureerd als actief-hoog, actief-laag of open-drain. Het Interrupt Capture-register legt de waarden van de poorten vast op het moment dat de interrupt wordt getriggerd, waardoor de toestand die de interrupt veroorzaakte wordt opgeslagen. De Power-On-Reset (POR) zet de registers op hun standaardwaarden en initialiseert de IC-toestandsmachine. De noodzaak voor bidirectionele werking komt voort uit het feit dat elke l²C-busdeelnemer zowel moet kunnen lezen (bijvoorbeeld commando's) als vastgelegde 8+8 bit data over de bus moet kunnen verzenden.

Zoals bij alle units voor de I²C-bus het geval is, kan het adres van de MCP23017 worden ingesteld binnen een bereik van acht adressen; voor dit doel heeft hij pinnen A0, A1, A2, waarmee de adressen van de slaveunit kunnen worden ingesteld, als deze direct vanaf de I²C-bus worden aangesproken. Elk van deze adresbits wordt ingesteld met een een van de schakelaars van SW1: elke gesloten schakelaar maakt de corresponderende bit van het adres nul, terwijl omgekeerd een open schakelaar voor een één staat. De mogelijkheid om acht adressen te definiëren maakt het mogelijk om tot acht I/O-expanders op dezelfde bus aan te sluiten en zo een maximum van 128 I/O's aan te sturen met slechts drie lijnen. Voor alle toepassingen waarvoor je het breakoutboard wilt gebruiken, geven we in tabel 1 het adres voor elke schakelaarcombinatie.

Met deze hardware is de logica van de werking als volgt: iedere keer dat het board een string ontvangt op de SDA-lijn van de I²C-bus (geklokt door het signaal op de SCL-lijn), voert het MCP23017-IC het betreffende commando uit (in dit geval het commando om een databyte te laden) en worden de acht uitgangslijnen IOA0...IOA7 en IOB0...IOB7 corresponderend ingesteld. IOA0 zal bijvoorbeeld de toestand van het eerste bit van byte 1 aannemen, IOA1 die van het tweede bit enzovoort. Hetzelfde gebeurt met IOB0...IOB7, die exact de bits van de tweede databyte weerspiegelen.

Natuurlijk vinden conversie en uitvoer alleen plaats als de ontvangen string het I²C-busadres bevat dat overeenkomt met het adres dat via de schakelaars van SW1 is ingesteld voor U1. Wanneer een string wordt ontvangen, werkt het IC de status van zijn uitgangen bij en de respectievelijke logische niveaus bepalen of de schakelingen 'stroomafwaarts' (bijvoorbeeld LED's of displaysegmenten) worden ingeschakeld of uitgeschakeld; als er vervolgens geen string wordt verzonden, behoudt de uitgangsstatus het laatste bytepatroon omdat de uitgangen van het MCP23017-IC worden vergrendeld. Het bovenstaande geldt voor de uitgangsmodus, dat wil zeggen het schrijven van de status van de twee via de I²C-bus ontvangen bytes naar de uitgangsregisters A en B. Als daarentegen een leescommando van de bus wordt gelezen, leest de MCP23017 de I/O-status van elk register in en genereert hij twee bytes, de eerste met de status van IOA0...IOA7 en de tweede met de logische toestand van IOB0...IOB7; deze worden vervolgens als antwoord via de I²C-bus verzonden.

Praktische realisatie

Nu we het schema hebben beschreven, kunnen we ons met de bouw bezighouden. Daarna zullen we een toepassingsvoorbeeld geven op basis van interfacing met een Arduino-board, compleet met de bijbehorende sketch. Zoals gebruikelijk hebben we een (dubbelzijdige) print ontworpen waarvan de layout van [2] kan worden gedownload. Aan de hand daarvan kun je de print zelf maken (belichten, ontwikkelen en etsen). Nadat je de print hebt geëtst en geboord, kun je het handjecol componenten monteren; ten behoeve van diegenen die niet van oppervlaktemontage houden, zijn dit allemaal conventionele through-hole-types. Plaats en soldeer eerst de weerstanden, ga dan verder met het IC-voetje (met de inkeping in de richting van C1, zoals te zien in de componentenopstelling) en de drievoudige DIP-schakelaar,



Onderdelenlijst

Weerstanden: R1...R6 = 4k7

Condensatoren: C1 = 100 n keramisch

Halfgeleiders: U1 = MCP23017-E/SP

Diversen:

SW1 = 3-voudige DIP-schakelaar 1x 28-pins IC-voet 2x 12-polige pinheader male 1x 4-polige pinheader, male

Print (zie tekst)



te monteren met schakelaar 1 naar links gericht als we in de lengterichting met C1 boven naar de print kijken. Soldeer tenslotte de vierpolige pinheader gemarkeerd I2C, en daarna aan de lange zijden en aan de onderzijde van de print de beide 12-polige pinheaders die montage op breadboards of plaatsing op andere boards mogelijk maken; de verbinding met een Arduino kan met klassieke male/female jumperkabeltjes worden gemaakt. Zodra alle componenten gesoldeerd zijn, kun je de MCP23017 in zijn voetje prikken, met de inkeping in de richting van C1. Daarna je breakout-board klaar voor experimenten of voor de ontwikkeling van prototypes.

Toepassingsvoorbeeld met Arduino

Het breakout-board is gemaakt om in combinatie met een microcontroller te worden gebruikt, omdat I/O-expanders meestal worden toegepast bij apparaten die zijn uitgerust met een seriële I²C-interface. Aangezien Arduino deze



Figuur 4. De hardware voor het testen van de voorbeeldsketch.



Figuur 3. Bedrading van het toepassingsvoorbeeld (relais met drukknopbedienning).

Listing 1. Demoproject.

```
#include <Adafruit_MCP23X17.h>.
#include <Adafruit_MCP23X17.h>
Adafruit_MCP23X17 mcp;
int i = 0;
int OUT[] = ; //Represents MCP23017 PIN (A7...A0)
int IN[] = ; //Represents MCP23017 PIN (B0...B7)
int STAT0[] = ; //For each output, every toggle the status is being saved
void setup()
{
Serial.begin(9600);
Serial.println("MCP23017 INPUT/OUTPUT");
if (!mcp.begin_I2C(0x20))
                           //0x20 is MCP23017's address with A0=A1=A2 > ON(GND)
{
   Serial.println("MCP Error!"); //If MCP is not found, the error is visualized
   while (1);
}
//bank A Pin set as outputs and B bank as inputs
//The STATO variable to 0 to indicate idling outputs
for (i=0; i<8; i=i+1)</pre>
{
  mcp.pinMode(OUT[i], OUTPUT);
   mcp.pinMode(IN[i], INPUT_PULLUP);
   STATO[i] = 0;
}
}
void loop()
{
String Testo_Debug = "";
for (i=0; i<8; i=i+1)</pre>
{
   //If button pressed or output not activated, I activate it
   if ((mcp.digitalRead(IN[i])==0) && (STATO[i]==0))
  {
    STATO[i] = 1;
    Testo_Debug = "Pulsante " + String(i+1) + " premuto";
    Serial.println(Testo_Debug);
    mcp.digitalWrite(OUT[i], HIGH);
   //If button released and output is active,I de-activate it
   if ((mcp.digitalRead(IN[i])==1) && (STATO[i]==1))
  {
    STATO[i] = 0;
    Testo_Debug = "Pulsante " + String(i+1) + " rilasciato";
    Serial.println(Testo_Debug);
    mcp.digitalWrite(OUT[i], LOW);
  }
}
delay(10);
}
```

bus ondersteunt, hebben we voorbeeldcode geschreven om de I/O's van de MCP23017 via Arduino te lezen en aan te sturen; deze code kan worden gedownload van [3]. De sketch maakt het in principe mogelijk om de toestand van het register van poort A te schrijven met een door Arduino via de bus verzonden byte, waarvan de bits overeenkomen met de toestand die van poort B wordt ingelezen, die hier dus als invoer fungeert. Om het voorbeeld een concrete toepassing te geven, hebben we besloten om de logische toestanden van de I/O's van poort A, die in dit geval dus digitale uitgangen zijn, te gebruiken om een relaisprint aan te sturen. Concreet moeten we de acht relais-stuurlijnen van een 8-kanaals relaisprint verbinden met de I/O-bank van poort A, terwijl acht NO-drukknoppen (normally open, normaal geopend) moeten worden verbonden met poort B, met GND als gemeenschappelijke aansluiting. Om deze toepassing te realiseren, is het nodig om een Arduino UNO, het breakout-board, de relaisprint en de knoppen aan te sluiten zoals in het bedradingsschema van figuur 3. Figuur 4 toont de praktische realisatie van deze toepassing. Aangezien dit heel gebruikelijke drukknoppen zijn die geen externe elektronica nodig hebben, werden de interne pull-up-weerstanden van de MCP ingeschakeld (via de bibliotheek) om ze in te lezen en toestandsveranderingen te herkennen. Op deze manier wordt de corresponderende uitgang geactiveerd wanneer de knop wordt ingedrukt en contact met massa maakt.

Om dit kleine demoproject uit te voeren, werd eenvoudige Arduino UNO-gebaseerde code geschreven, gebruik makend van de Adafruit-bibliotheek die, zoals je in **listing 1** ziet, in de eerste regel van de sketch wordt opgenomen.

Alvorens de code in het programmageheugen van de Arduino te laden, moet je beslist de bibliotheek downloaden van *www.adafruit.com* en deze te installeren met behulp van de Library Manager van de IDE, of de inhoud van het ZIP-bestand uitpakken en vervolgens de volledige *Adafruit_MCP23017_Arduino_Library*-map te kopiëren naar de *libraries*-map die normaal te vinden is binnen het besturingssysteem via het pad *DocumentsArduino\ libraries*. Na het laden van de bibliotheek volstaat het de voorbeeldcode in het board te laden nadat je de juiste COM-poort hebt gekozen in het *Tools*-menu van de IDE. In de code moet een byte van de MCP23017 worden opgevraagd die de status van de knoppen bevat.

Tabel 1. Instelling van het perifere adres van de MCP23017.

ADRES	A2	A1	A0
0x20	ON	ON	ON
0x21	ON	ON	OFF
0x22	ON	OFF	ON
0x23	ON	OFF	OFF
0x24	OFF	ON	ON
0x25	OFF	ON	OFF
0x26	OFF	OFF	ON
0x27	OFF	OFF	OFF

De betreffende gegevens worden verwerkt en vervolgens in een byte geschreven die naar het IC wordt gestuurd en die de status van de lijnen van poort A stabiel instelt tot een refresh komt. Om de interfacing te laten werken, moeten de DIP-schakelaars A0, A1 en A2 correct worden ingesteld, want als het IC niet het adres 0x20 krijgt toegewezen, verschijnt er een foutmelding in seriële monitor; het adres van het breakout-board wordt toegewezen door de drie bits A0, A1, A2 nul te maken (dat wil zeggen alle schakelaars gesloten). Als je het adres wilt wijzigen, raadpleeg dan **tabel 1**. Je moet dan natuurlijk ook in de code het adres navenant wijzigen. **|**

230469-03



WEBLINKS

- [1] Webpagina van de MCP23017 van Microchip: https://microchip.com/en-us/product/mcp23017
- [2] Printlayout voor dit project: https://tinyurl.com/9sh3ct5t
- [3] Sketch voor de voorbeeldapplicatie: https://tinyurl.com/yx3fr4r6
- [4] Microchip-datasheet: https://tinyurl.com/yc39n93v

Al-Specielos machine learning met de Jetson Nano

Tam Hanna (Hongarije)

Kunstmatige intelligentie op GPU's betekent niet noodzakelijkerwijs dure grafische kaarten en een enorm energieverbruik. NVIDIA beweegt zich al een tijdje in de richting van kleinere systemen, zeker ook door de concurrentie van microcontroller-fabrikanten. In dit artikel bekijken we de Jetson Nano en een kleine demotoepassing.

De markt voor AI-versnelling in embedded systemen is in beweging. Bedrijven als Canaan en Maxim Integrated vechten verbitterd om suprematie. ARM waagt zich ook in deze arena met de aankondiging van de Cortex M52. Het doel is om kleine AI-systemen aan te bieden die basale AI-taken uitvoeren op de veelbesproken edge zonder verbinding met het internet. Dit leidt tot een veel stabielere systeemarchitectuur omdat de ML- of AI-taken dan kunnen blijven draaien ook als de verbinding met de bovenliggende server wegvalt. NVIDIA probeert al een tijdje op dit gebied voet aan de grond te krijgen met de Jetson-serie. Eerlijkheidshalve moet worden opgemerkt dat maatgesneden boards met deze systemen niet werkelijk haalbaar zijn voor kleinere bedrijven, al was het maar vanwege de extreme bandbreedte die nodig is om toegang te krijgen tot het DDR-RAM. NVIDIA is zich bewust van dit probleem. Het portfolio-overzicht (beschikbaar op [1]) dat elektronica-ingenieurs een overzicht biedt van het volledige Jetson ecosysteem, bevat daarom ook verschillende 'compute cards', zoals de Jetson TX2 die in **figuur 1** te zien is.

Er moet ook worden opgemerkt dat door de dominantie van NVIDIA in de GPU-sector, het goed ontwikkelde ecosysteem van derden nu ook het Jetson-productassortiment heeft ontdekt. Een overzicht met verschillende producten van derden is te vinden op [2], waarvan de integratie in een interne oplossing veel ontwikkelingstijd kan besparen (denk bijvoorbeeld aan het ontwerp van de behuizing en de selectie van camera's en dergelijke).

De boel aan de praat krijgen

Voor 'zij-instromers' biedt NVIDIA de NVIDIA Jetson Nano Developer Kit die je ziet in **figuur 2** (naast een Raspberry Pi en een Orange Pi 5+). Bij het ter perse gaan van dit artikel bedroeg de beste OEMSecrets-prijs (zie [3]) \in 155. Hoewel dit iets duurder is dan een Raspberry Pi, is de NVIDIA-technologie beter geïntegreerd in het algemene AI-ecosysteem. Als je de NVIDIA Jetson Nano koopt, is het aan te raden om ook een voedingseenheid te bestellen met een voedingsconnector in de gebruikelijke afmetingen van 5,5/2,1 mm. Hieronder gebruikt de auteur een MeanWell GST25E05-P1J.

Jetson TX2 Series

The extended Jetson TX2 family of embedded modules gives you up to 2.5X the performance of Jetson Nano in as little as 7.5W. Jetson TX2 NX offers pin- and form factor compatibility with Jetson Nano, while Jetson TX2, TX2 4GB, and TX2i all share the original Jetson TX2 form factor. The rugged Jetson TX2i is ideal for settings including industrial robots and medical equipment.

Learn More >

Figuur 1. Het gebruik van deze kant-en-klare module maakt het veel eenvoudiger om de Jetson te integreren in eigen schakelingen (bron: NVIDIA).

Een zorgvuldige blik op het systeem (zie ook **figuur 3**) laat zien dat het een tweedelig product is. Naast de carrierprint, die de verschillende interfaces bevat, is er de eigenlijke computermodule met koellichaam. Er moet beslist een microSD-kaart met het besturingssysteem in de computermodule worden geplaatst.

Micro-USB is in theorie ook mogelijk

Als je een heel krachtige micro-USB voeding bij de hand hebt en je gebruikt de micro-USB poort niet om externe hardware aan te sluiten, dan kun je ook voor deze voedingsmethode kiezen. Vanwege 'onaangenaamheden' met de Raspberry Pi wil de auteur zich niet nogmaals de vingers verbranden en vertrouwt hij op een klassieke DC-voeding.

De ingebouwde SoC is een multicore-systeem dat naast de eigenlijke AI-versneller ook vier volwaardige Arm Cortex A57-cores heeft. Dit resulteert in een opstelling die doet denken aan klassieke procescomputers die meestal Embedded Linux draaien. NVIDIA raadt aan om een MicroSD-kaart te gebruiken met minstens 32 GB capaciteit en een minimumsnelheid van UHS1. Het imagebestand is te vinden op [4], dat je zoals gebruikelijk met een kaartlezer kunt uitpakken op je geheugenkaart.

Een USB-muis en een USB-toetsenbord zijn ook nodig voor de eerste start; Jetson ondersteunt zowel HDMI als DisplayPort voor schermuitvoer. We gaan ervan uit dat er in de volgende stappen ook een ethernetkabel is aangesloten – eigenlijk is dat vanzelfsprekend.

Niet voor worstvingers!

De microSD-houder van de Jetson-module is gebaseerd op het 'form-fit' principe. Hij wordt geplaatst en verwijderd door hem 'diep in te duwen' – als je een elektronica-pincet en een fijne schroevendraaier gebruikt, kun je dit voor elkaar krijgen zonder het evaluatieboard te demonteren.

Het ontbreken van een camera blijkt problematisch tijdens de installatie. In plaats van een CCD-sensor gebruikt NVIDIA twee van de connectoren die we kennen van Raspberry Pi 4 en dergelijke. Sommige Raspberry Pi-camera's kunnen rechtstreeks op de Jetson worden aangesloten. In de volgende stappen gebruikt de auteur een Raspberry Pi 2-camera die wordt aangesloten op de CAMO-poort via een FPC-kabel die is gemaakt voor de Raspberry Pi. Er moet beslist worden opgemerkt dat de kabelversie die bedoeld is voor de Raspberry Pi 5 niet compatibel is met de Jetson. Verschillende 3D-modellen voor het positioneren van de camera zijn beschikbaar in ThingiVerse, en het afdrukken ervan zal het leven van de ontwikkelaar een stuk eenvoudiger maken.

Als je de NVIDIA Jetson in gebruik wilt nemen met de eerder genoemde MeanWell-voeding, moet je rekening houden met geniepigheidje waar beginners last van kunnen hebben. De jumper in **figuur 4** is bij levering niet geplaatst: de DC connector is dan



Figuur 2. De Jetson naast een Raspberry Pi 5 en een OrangePi 5 Plus.



Figuur 3. Als je twee schroeven losdraait, kun je de development kit uit elkaar halen.

Figuur 4. Deze jumper is van cruciaal belang.



niet aangesloten op de voeding. Als de voeding is aangesloten, gaat de status-LED niet branden, en dat kan verwarring veroorzaken. Tijdens de eerste keer opstarten schakelt de procescomputer meteen het HDMI-scherm in – de herconfiguratie van de microSD-kaart en enkele andere 'initiatierituelen' nemen enige tijd in beslag. Zodra het werk is gedaan, presenteert het systeem een Ubuntu-desktop en vraagt het je om de welkomstwizard te voltooien.

Over het algemeen is dit de gebruikelijke Ubuntu-installatiewizard, maar NVIDIA heeft onder andere een stap toegevoegd om de interne softwarelicenties te accepteren. De wizard voor het selecteren van het voedingsmodel is belangrijk: het is aan te raden om de standaard-optie aan te houden. Dit garandeert dat de Jetson de maximale hoeveelheid stroom krijgt.

Zodra de wizard met succes is voltooid, is het nog even wachten tot het Jetson-board klaar is voor gebruik – er kunnen ook enkele prompts op het bureaublad verschijnen om het systeem opnieuw op te starten.

Eerste experimenten

NVIDIA vertrouwt op een min of meer volledig standaard-conform Ubuntu 18.04 in de Jetson. De auteur houdt ervan om zulke procescomputers te configureren voor toegang op afstand om zich zo de moeite te besparen om te schakelen tussen de toetsenborden van de PC en de procescomputer.

De Settings-applicatie in Ubuntu is echter ongeschikt voor deze taak, daarom draaien we sudo apt-get update en sudo apt-get upgrade om de package-inventaris bij te werken in de eerste stap. Elke vraag moet worden bevestigd door op *Enter* te drukken; de herstart van het Docker-systeem moet worden toegestaan. De Jetson moet dan opnieuw worden opgestart. Het invoeren van sudo apt install vino zorgt ervoor dat de VNC-server klaar is voor gebruik. Tenslotte zijn de onderstaande commando's nodig om de configuratie in een bruikbare staat te brengen. De string thepassword moet natuurlijk vervangen worden door een wachtwoord dat geschikt is voor jouw installatie.

tamhan@tamhan-desktop:~\$ mkdir -p ~/.config/autostart tamhan@tamhan-desktop:~\$ cp /usr/share/applications/ vino-server.desktop ~/.config/autostart tamhan@tamhan-desktop:~\$ gsettings set org.gnome.Vino prompt-enabled false tamhan@tamhan-desktop:~\$ gsettings set org.gnome.Vino require-encryption false tamhan@tamhan-desktop:~\$ gsettings set org.gnome.Vino authentication-methods "['vnc']"

tamhan@tamhan-desktop:~\$ gsettings set org.gnome.Vino
vnc-password \$(echo -n 'thepassword'|base64)

Na de volgende herstart is de Jetson toegankelijk met de VNC-client Reminna als een gebruiker is ingelogd. Houd er echter rekening mee dat de applet voor externe toegang in de Settings-applicatie nog steeds niet werkt.

Als volgende stap kun je een eerste test van de cameraverbinding uitvoeren door het volgende commando in te typen:

tamhan@tamhan-desktop:~\$ gst-launch-1.0 nvarguscamerasrc
! nvoverlaysink

De camera-preview die met dit basiscommando kan worden geactiveerd, verschijnt alleen op een monitor die fysiek op de Jetson is aangesloten – het systeem dat via VNC met de computer communiceert, ziet in plaats daarvan de terminaluitvoer. Om het hele proces te beëindigen, is het voldoende om een interrupt te versturen met *Control* + *C*.

De feitelijke toegang tot de camera wordt dan – over het algemeen – uitgevoerd met methoden die bekend zijn van de PC of het werkstation. Bijzonder interessant is het bestand [5], dat de opzet van een Open CV-gebaseerde pijplijn demonstreert.

De volgende commando's zijn nodig om het lokaal uitvoerbaar te maken:

tamhan@tamhan-desktop:~\$ git clone https://github.com/ JetsonHacksNano/CSI-Camera tamhan@tamhan-desktop:~\$ cd CSI-Camera/ tamhan@tamhan-desktop:~/CSI-Camera\$ python3 simple_ camera.py

Het geopende cameravenster is dan ook zichtbaar via VNC omdat het de informatie niet rechtstreeks naar de framebuffer van de Tegra GPU stuurt.

Als het camerabeeld ondersteboven staat, helpt het om de parameter flip_method in het bestand aan te passen:

Interactie met GPIO-pinnen (via Python)

Kunstmatig intelligente systemen vereisen compleet andere vaardigheden van de ontwikkelaar; over het algemeen is er weinig overlap met klassieke embedded ontwikkeling. In de praktijk is mij gebleken dat mensen van buiten het vakgebied met een achtergrond in traditionele mainframe-computertechnologie AI vaak beter onder de knie krijgen dan embedded ontwikkelaars. De partner van de auteur, die zeer efficiënt in Java programmeert, lost AI-taken sneller op – maar ze heeft weinig kennis van assembler.

Het doel van deze kleine omweg via software-architectuur is om aan te geven dat we in de volgende stappen met opzet GPIO-toegang op de Jetson in Python illustreren. De reden hiervoor is dat het merendeel van de gebruikersgeoriënteerde ontwikkeling van AI-systemen plaatsvindt in Python: als je hier op C vertrouwt, word je uiteindelijk getrakteerd op een onnodige 'native' interface.

De standaard-distributie van Python bevat geen package management op de Jetson, daarom moeten de commando's sudo apt install python-pip en sudo apt install python3-pip



Figuur 5. De NVIDIA Jetson produceert blokgolven met Python.

worden ingevoerd. De volgende stap is controleren of de GPIOmodules correct zijn geïnstalleerd. Ook hier zijn twee commando's nodig omdat de Python-omgevingen (uiteraard) niet echt in staat zijn om hun bibliotheekrepository te delen:

tamhan@tamhan-desktop:~\$ sudo pip install Jetson.GPIO tamhan@tamhan-desktop:~\$ sudo pip3 install Jetson.GPIO

We kunnen de Jetson GPIO-API hier niet uitputtend bespreken vanwege ruimtegebrek. Onder [6] vind je een groep kant-en-klare voorbeelden die de API volledig uitleggen.

Het onderstaande programma volstaat voor onze kleine test:

```
import RPi.GPIO as GPIO
import time
output_pin = 18 # BCM pin 18, BOARD pin 12
def main():
    GPI0.setmode(GPI0.BCM)
# BCM pin-numbering scheme from Raspberry Pi
   GPI0.setup(output_pin, GPI0.0UT, initial=GPI0.HIGH)
    print("Starting demo now! Press CTRL+C to exit")
    try:
       while True:
           GPI0.output(output_pin, GPI0.HIGH)
            GPI0.output(output_pin, GPI0.LOW)
            GPI0.output(output_pin, GPI0.HIGH)
            GPI0.output(output_pin, GPI0.LOW)
    finally:
       GPI0.cleanup()
```

if __name__ == '__main__': main()

Hier is vooral interessant dat NVIDIA de 'ontluikende' Jetsonontwikkelaar verschillende manieren biedt om de pinnen te adresseren – wij gebruiken hier de GPIO.BCM-optie, die gebaseerd is op de pinout van de Raspberry Pi. De beloning voor onze inspanningen is het screenshot van **figuur 5** – de frequentiestabiliteit is misschien niet geweldig, maar het is meer dan voldoende voor het triggeren van IoT-gebeurtenissen.

Ook moet worden opgemerkt dat het uitvoeren van GPIOprogrammavoorbeelden zonder superuser-rechten soms problemen kan veroorzaken. Zie [7] voor een gedetailleerde bespreking van dit onderwerp.

Experimenten met de ML-functie

In theorie bevordert de beschikbaarheid van een volwaardig Linuxbesturingssysteem en een (zeer snelle) USB3-interface de uitvoering van experimenten. Een mogelijk voorbeeld zou Stable Diffusion zijn: in de praktijk is het mogelijk om de Jetson te gebruiken als beeldgenerator met behulp van een (aangepaste) runner.

In de praktijk is deze aanpak echter niet aan te raden: zowel het relatief kleine grafische geheugen van slechts 4 GB VRAM als het geringe aantal van 'slechts' 128 cores zorgen ervoor dat het genereren van afbeeldingen geduld vereist. Op Reddit staan verslagen van ML-experimenteerders die tot 5 uur rekentijd per afbeelding schatten met een clustergrootte van 512×512 pixels.

Vliegende start voor populaire frameworks

NVIDIA probeert het ontwikkelaars zo gemakkelijk mogelijk te maken om verschillende veelgebruikte ML-frameworks te implementeren. Een lijst met ondersteunde producten inclusief – geoptimaliseerde – installatie-instructies is te vinden op [8].



Figuur 6. Het demoprogramma dat bij de Jetson wordt geleverd is niet erg 'communicatief' op commandoregel-niveau.

Hetzelfde geldt voor de 'end-to-end' oplossingen die vaak worden gedemonstreerd – het trainen van een model vereist zoveel rekenkracht en middelen dat NVIDIA in de meeste tutorials aanbeveelt om het uit te besteden aan een desktop of mainframe – de Jetson wordt dan geparametriseerd met de kant-en-klare modelgewichten. In [9] vind je min of meer gebruiksklare voorbeelden die de prestaties van de Jetson op een inzichtelijke manier illustreren. Om deze model-'powershow' te gebruiken, is het nodig om een NVIDIA-softwarepackage te downloaden en te implementeren. In theorie is het gebruik van een docker-container hier ook mogelijk – voor mensen die niet bekend zijn met containertechnologie is lokaal compileren een alternatief, dat kan worden bereikt door de volgende commando's in te voeren:

```
sudo apt-get update
```

sudo apt-get install git cmake libpython3-dev
python3-numpy
git clone --recursive --depth=1 https://github.com/
dusty-nv/jetson-inference
cd jetson-inference
mkdir build
cd build
cd build
cmake ../
make -j\$(nproc)

sudo make install sudo ldconfig

Voel je vrij om de vraag over het installeren van de trainingsonderdelen met ja of nee te beantwoorden – de auteur heeft in de volgende stappen *No* geantwoord om wat geheugen te besparen op zijn microSD-kaart, die slechts 32 GB groot is.

Nadat het compilatieproces met succes is voltooid, kan een relatief complexe projectstructuur worden gevonden in de map ~/jetson-inference/build/aarch64\$/bin, die naast verschillende binaire bestanden ook Python-bestanden bevat. Het is interessant om op te merken dat NVIDIA hier zelfs enkele kant-en-klare testvoorbeelden zet.

Als eerste willen we de classifier gebruiken – deze analyseert afbeeldingsbestanden en bepaalt wat er te zien is in de geleverde afbeelding:

tamhan@tamhan-desktop:~/jetson-inference/build/aarch64/ bin\$./imagenet.py images/orange_0.jpg images/test/ output_0.jpg

De eerste maal uitvoeren van deze opdracht neemt iets meer tijd in beslag omdat de meegeleverde modules geoptimaliseerd zijn voor de vereisten van het Jetson-systeem. Daarna zien we de timinginformatie van **figuur 6**. Om de resultaten van het ML-proces te visualiseren, moet je het gegenereerde afbeeldingsbestand bekijken – door het volgende commando in te voeren open je de uitvoermap direct in de file manager van Nautilus:

tamhan@tamhan-desktop:~/jetson-inference/build/aarch64/ bin\$ nautilus images/test/output_0.jpg

De beloning voor je inspanningen is het scherm van **figuur 7**.

Analyse van het Python-bestand

Nu werpen we een snelle blik op het voorbeeldprogramma dat we zojuist hebben gebruikt. De broncode kan worden geopend door de volgende opdrachtregel in te voeren:

tamhan@tamhan-desktop:~/jetson-inference/build/aarch64/ bin\$ gedit imagenet.py

Al op het eerste gezicht is het duidelijk dat de hier gebruikte bibliotheek verwant is aan het klassieke ImageNet – NVIDIA verpakt verschillende veelgebruikte kunstmatige intelligentie-systemen in de Jetson-starterkit.

De kern van het programmavoorbeeld is een eindeloze lus die een afbeelding opneemt uit de invoerstroom, deze doorsluist naar het ML-model en uiteindelijk de 'berekende' informatie toont (**listing 1**).

De initialisatie van de gegevensstromen en het te gebruiken model wordt daarvoor uitgevoerd volgens het volgende schema:

```
net = imageNet(args.network, sys.argv)
input = videoSource(args.input, argv=sys.argv)
output = videoOutput(args.output, argv=sys.argv)
font = cudaFont()
```

Een ander interessant aspect is het verkrijgen of het invullen van de network-parameter, die de naam van het te verwerken model levert. Hier vertrouwt NVIDIA op de klasse ArgParser, die – normaal



Figuur 7. Het NVIDIA-voorbeeldprogramma blijkt een voorbeeldige fruitdetector te zijn.

gesproken – gespecialiseerd is in het verwerken van parameters die via de commandoregel worden aangeleverd.

In het geval van deze declaratie wordt een standaardwaarde ingevoerd die normaal gesproken het GoogLeNet activeert en laadt:

parser.add_argument("--network", type=str, default="googlenet", help="pre-trained model to load (see below
for options)")

Terzijde: 'begeleide' online-training inclusief certificering

Het succes van de voormalige Sovjet-Unie in verschillende Arabische en Afrikaanse landen kan deels worden toegeschreven aan het hechte trainingspartnerschap – wat de leerling leert, gebruikt hij later in zijn werk. NVIDIA is zich duidelijk bewust van deze situatie en daarom is Jetson AI Certification – zoals te zien in **figuur 8** – een volledig gratis tweedelige ML-cursus. Ook moet worden benadrukt dat het succesvol afronden van de cursus door NVIDIA wordt beloond met een certificaat.

Als je geïnteresseerd bent, raden we je aan om [10] te bezoeken. Daar vind je meer informatie over hoe je je deelname aan de NVIDIA-training zo efficiënt mogelijk kunt organiseren.



Figuur 8. Tweesporen-onderwijs in NVIDIA-stijl (bron: NVIDIA [10]).

Listing 1. Classificatie.



Voordelen dankzij Linux

Met de Jetson stuurt NVIDIA een hybride in de race die wat kunstmatige intelligentie betreft vlees noch vis is. Aan de ene kant hebben speciale low-power microcontrollers zoals de Maxim MAX78000 een aanzienlijk lager energieverbruik. Aan de andere kant lijden dergelijke controllers onder het feit dat ze geen ondersteuning bieden voor CUDA: een model dat op een PC of mainframecomputer kan draaien, moet dus worden aangepast voordat het met deze chips in IoT-applicaties kan worden gebruikt.

Aan de andere kant is de NVIDIA Jetson geen volwaardige GPU: zowel qua energieverbruik als qua ondersteunde CUDA-variant (CUDA 11 werkt niet) is de module geen volwaardige vervanger voor een RTX 4000.

Het komt erop neer dat een implementatie loont als een model dat probleemloos werkt op een computer of mainframe met weinig moeite moet worden gemobiliseerd en het voldoende heeft de middelen die Jetson biedt. De beschikbaarheid van een Linuxbesturingssysteem betekent dat er relatief weinig conversiewerk nodig is voor een datawetenschapper – vertrouwd raken met de ingebedde API's die Maxim en consorten gebruiken vereist aanzienlijk meer inspanning. De hogere kosten van de hardware kunnen dus snel en efficiënt worden afgeschreven, vooral bij kleinere series.

230740-03

Vragen of opmerkingen?

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

WEBLINKS

- [1] Portfolio-overzicht: https://www.nvidia.com/en-eu/autonomous-machines/embedded-systems/
- [2] Lijst van verschillende producten van derden: https://developer.nvidia.com/embedded/ecosystem
- [3] OEMSecrets koopjes: https://www.oemsecrets.com/compare/%20945-13450-0000-100
- [4] Download van het image-bestand: https://developer.nvidia.com/jetson-nano-sd-card-image
- [5] Open CV-gebaseerd pijplijn-bestand: https://github.com/JetsonHacksNano/CSI-Camera/blob/master/simple_camera.py
- [6] Kant-en-klare voorbeelden: https://github.com/NVIDIA/jetson-gpio/tree/master/samples
- [7] Uitvoering van GPIO-programmavoorbeelden zonder superuser-rechten: https://github.com/NVIDIA/jetson-gpio/issues/20
- [8] Lijst van ondersteunde producten: https://elinux.org/Jetson_Zoo
- [9] Gebruiksklare voorbeelden: https://github.com/dusty-nv/jetson-inference
- [10] Jetson Al-cursussen en certificaten: https://developer.nvidia.com/embedded/learn/jetson-ai-certification-programs

Exhibition&Conference

CONNECTING THE EMBEDDED COMMUNITY 9–11.4.2024

Get your free ticket now!



embedded-world.de/codes Use the voucher code ew24ELE

Media partners

Markt&Technik

Elektronik

automation

Elektronik automotive Elektronik

elektroniknet.de

NÜRNBERGMESSE

202een AI-odyssee

een eerste verkenning van TensorFlow

Brian Tristam Williams (Elektor)

Deze aflevering verkent het potentieel van TensorFlow Lite op de Raspberry Pi en kijkt naar de praktische aspecten van AI in compacte vorm, waarbij we de grenzen zoeken van wat mogelijk is op energiezuinige apparaten.



Wat is TensorFlow?

TensorFlow, ontwikkeld door het Google Brain-team, is een open-source bibliotheek voor numerieke berekeningen en machine learning en speelt een cruciale rol in het bredere AI-ecosysteem, dat generatieve AI-technologieën zoals ChatGPT omvat. Het is een hoeksteen in deep learning, waarmee modellen kunnen worden gemaakt die 'big data' kunnen verwerken en daarvan kunnen leren.

TensorFlow en generatieve AI-modellen vullen elkaar vaak aan, waarbij TensorFlow het basisplatform biedt voor het bouwen en trainen van een verscheidenheid aan AI-modellen, waaronder modellen die generatieve toepassingen kunnen aansturen. De veelzijdigheid maakt het geschikt voor een breed scala van taken, van eenvoudige classificaties tot complexe besluitvormingsprocessen, waardoor het een populaire keuze is in zowel academische als industriële applicaties. Het vermogen om grote datasets te verwerken, patronen te herkennen en ervan te leren maakt het een krachtig hulpmiddel in de bredere context van generatieve AI, die zich richt op het creëren van nieuwe content en datamodellen.



TensorFlow Lite: minder is meer

In deze aflevering van onze AI-reis neem ik een ietwat onconventionele route. Hoewel ik de beschikking heb over een redelijk krachtige PC met een degelijke RTX 4070 GPU en voldoende RAM, voel ik me aangetrokken tot de wereld van TensorFlow Lite [1] op de Raspberry Pi. Er zit een bepaalde charme in de uitdaging om AI-processen te optimaliseren voor zo'n compact en minder krachtig apparaat. Maar het gaat niet alleen om meer doen met minder; het gaat om het maken van slimme, op zichzelf staande oplossingen die draagbaar en efficiënt zijn. Soms heeft het project niet de omvang en kracht van een grote desktop-toren nodig, of kan het die niet aan. In deze scenario's biedt de Raspberry Pi een perfect alternatief. Door TensorFlow Lite uit te voeren op deze kleine maar krachtige machine wordt AI toegankelijker en beter aanpasbaar aan verschillende omgevingen, of het nu gaat om educatieve doeleinden, hobbyprojecten of echte toepassingen waarbij ruimte en stroomverbruik beperkende factoren zijn.

Met de opkomst van edge computing is de behoefte aan krachtige maar efficiënte AI-tools groter geworden. Dit is waar TensorFlow Lite, een lichtere en efficiëntere versie van TensorFlow, om de hoek komt kijken, vooral voor apparaten zoals de Raspberry Pi. De Raspberry Pi is een kleine, betaalbare en toch krachtige computer die het perfecte platform vormt voor het draaien van TensorFlow Lite, waardoor mogelijkheden voor machine learning binnen handbereik komen van hobbyisten, docenten en professionals.

Het installeren van TensorFlow Lite op een kleine single-board computer biedt aanzienlijke voordelen voor AI-toepassingen op grotere machines. TensorFlow Lite is geoptimaliseerd voor prestaties op 'lichtgewicht' hardware, waardoor het geschikt is voor een breed scala aan andere praktische toepassingen naast objectdetectie. Deze omvatten real-time gegevensverwerking, lokale besluitvorming in IoT-apparaten en het uitvoeren van AI-modellen voor taken zoals gebarenherkenning, omgevingssensoren en gezondheidsbewaking. Dit stelt Raspberry Pi-makers in staat om complexe AI-modellen in te zetten op een kosteneffectieve, toegankelijke manier en opent een wereld aan mogelijkheden voor innovatie en verkenning op het gebied van AI, zelfs voor kleinschaligere projecten.

In de volgende paragrafen bekijken we hoe we TensorFlow Lite op een Raspberry Pi kunnen installeren en hoe we ons kunnen voorbereiden op toekomstige innovatieve toepassingen.

Om te beginnen

Ik heb een lade vol met Raspberry Pi's, van de eerste tot de Zero tot de 5, maar hier ga ik een Raspberry Pi 4 gebruiken. Het is een goede middenweg in termen van toegankelijkheid voor onze lezers, netjes tussen de drie-modellen, die ik nog steeds soms te koop zie, en de nieuwkomer – de vijf. (Ik heb trouwens maar één Raspberry Pi 5, dat speelt ook een rol).

Dus waar ik mee begin is:

- > Raspberry Pi 4 Model B: we zullen zien of hij genoeg heeft aan zijn 4 GB RAM. (Ik gebruikte free -h op de commandoregel om me te herinneren hoeveel geheugen er aan boord is).
- 32 GB microSD-kaart: ik weet niet of ik per se zoveel nodig heb, maar het is de eerste die ik vond toen ik er een zocht.



Figuur 1. Raspberry Pi Imager heeft een overzichtelijke, gebruiksvriendelijke interface.

> Raspberry Pi OS Lite (64 bit): ik ben niet echt een fan van de overhead die GUI's met zich meebrengen en ik geef de voorkeur aan de rustgevende ja/nee en aan/uit van de tekstinterface, dus dit is de port van Debian Bookworm zonder desktopomgeving. Ik heb lang met webservers gewerkt en heb nooit echt een GUI in Linux nodig gevonden. Laat me weten of ik met de tijd mee moet gaan!

Ik deed een verse installatie van Raspberry Pi Imager v1.8.5, de versie die op [2] beschikbaar was op het moment van schrijven. Download gewoon de versie voor jouw OS en voer het uit. Het is echt geweldig in het vereenvoudigen van het installatieproces. Kies gewoon de versie van je Raspberry Pi-board, het besturingssysteem dat je wilt en de kaart die je in de kaartlezer van je computer hebt gestoken. De gebruikersinterface is echt schoon en eenvoudig (**figuur 1**). En nog beter: het werkt.

Na de installatie heb ik mijn versienummer gecontroleerd met cat / etc/os-release. De jouwe kan anders zijn, maar dit was mijn resultaat:

briantw@raspberrypi: \$ cat /etc/os-release PRETTY NAME="Debian GNU/Linux 12 (bookworm)"
NAME="Debian GNU/Linux"
VERSION_ID="12"
VERSION="12 (bookworm)"
VERSION_CODENAME=bookworm
ID=debian
HOME_URL="https://www.debian.org/"
SUPPORT_URL="https://www.debian.org/support"
BUG_REPORT_URL="https://bugs.debian.org/"

Qua accessoires heb ik gewoon een toetsenbord (geen muis nodig – jippie!), een Raspberry Pi 4-voeding en beide Micro HDMI-uitgangen aangesloten (**figuur 2**). Op dit moment gaat er een naar een lekker kleine draagbare monitor en de andere naar een capture-apparaat op mijn PC, die ik voorlopig gebruik voor schermafbeeldingen. Het uiteindelijke idee is om mijn code op het ene scherm te laten draaien en de video-uitvoer – bijvoorbeeld van een objectdetectiescript – zichtbaar te maken op het andere scherm.



Figuur 2. De Raspberry Pi-basisopstelling die ik gebruik voor dit onderzoek.

Installatie van TensorFlow Lite

Nu het uitgangspunt vastligt, proberen we TensorFlow Lite aan het draaien te krijgen op de Raspberry Pi.

Allereerst wat huishoudelijke taken. Laten we een update/upgradecyclus uitvoeren op de Raspberry Pi, om er zeker van te zijn dat we de laatste iteraties hebben van alles dat geïnstalleerd is. Begin met:

sudo apt-get update

Hoe lang dit gaat duren, hangt af van hoe ver je achterloopt met de laatste updates. Bij mij zijn er maar 12 items bijgewerkt. Daarna:

sudo apt-get upgrade

Bij mij resulteerde dit in een heel scherm vol upgrades, wat me altijd weer verbaast als ik net het nieuwste officiële OS heb geïnstalleerd. Heb hier wat geduld – dit kan van geval tot geval variëren. Wat betreft de installatie van TensorFlow Lite, spitte ik verschillende

Google-zoekresultaten door, en de meest toegankelijke handleiding die ik vond was van EdjeElectronics op GitHub [3]. Natuurlijk was er enige tijd verstreken sinds Edje's eerste beschrijving van het proces en nieuwe OS-releases en upgrades, dus waren er een paar kleine struikelblokken, die ik hier voor je zal documenteren.

De eerste stap was het klonen van de GitHub-repository met:

git clone https://github.com/EdjeElectronics/TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi.git

De bemoedigende reactie van mijn Pi:

-bash: git: command not found

Oh ja, het is een nieuwe installatie. Oeps. Installeer git met:

sudo apt install git

Dat werkte voor mij, dus nu we *git* geïnstalleerd hebben, terug naar het git clone... commando van hierboven. Als dat goed gaat, word je beloond met een submap in je homedirectory met de naam *TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi*. Dat is een beetje omslachtig, dus noem hem *tflite1* met behulp van het mv (move) commando:

mv TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi tflite1

Ga vervolgens naar die map met:

cd tflite1

EdjeElectronics stelt voor om een virtuele omgeving te creëren, dus dat doen wij ook:

sudo pip3 install virtualenv

De (voorspelbare) reactie van de Pi:

-bash: pip3: command not found

Het installeren van *pip3* is bijna net zo makkelijk opgelost als in het geval van *git*. Ik voerde eerst nog een sudo apt-get update uit, waar negen dingen werden bijgewerkt, en toen dit:

sudo apt-get install python3-pip

Daar werd het nodige geïnstalleerd, wat resulteerde in nog een scherm vol tekst.

Vervolgens probeerde ik het sudo pip3... commando opnieuw. Debian had daar geen zin in. Na wat zoekwerk kwam ik erachter dat ik mijn eigen virtuele omgeving moest maken. Ervan uitgaande dat je nog steeds in de *tflite1*-directory bent, doe je dat met:

python3 -m venv tflite1-env

Activeer vervolgens deze virtuele omgeving:

source tflite1-env/bin/activate

Nu installeren we de TensorFlow Lite-afhankelijkheden en OpenCV. Zoals eerder besproken, heeft TensorFlow Lite vele toepassingsgebieden, maar deze installatie zal het voorbereiden op een zeer populair gebied: machine vision en objectherkenning. OpenCV, een open-source bibliotheek ontworpen voor computer vision- en machine learningtaken, zal daarom van pas komen.

Gelukkig heb je hier geen curl, wget of git clone nodig, want de maker van de repo heeft een mooi 40-regelig shell-script voor ons gemaakt genaamd *get_pi_requirements.sh*. Als je nieuwsgierig bent om te zien wat het script gaat doen voordat je het uitvoert, kun je de inhoud bekijken door cat get_pi_requirements.sh uit te voeren, maar wij voeren het gewoon uit met:

bash get_pi_requirements.sh

Ik maakte de fout om dit te proberen nadat ik de Raspberry Pi opnieuw had opgestart, en toen kreeg ik een aantal foutmeldingen en waarschuwingen. Reden? Dat source-commando dat we eerder hebben uitgevoerd in de *tflite1*-directory moet elke keer worden uitgevoerd als je opnieuw opstart of een nieuwe Terminal-sessie start. Dit nam een paar minuten in beslag, dus heb nog een beetje geduld!

Nu krijgen we de optie om een TensorFlow Lite-voorbeeldmodel van Google te gebruiken, of om er een te gebruiken dat we zelf hebben getraind. Als beginner heb ik gekozen voor dat van Google:

wget https://storage.googleapis.com/download. tensorflow.org/models/tflite/coco_ssd_mobilenet_v1_1.0_ quant_2018_06_29.zip

Dat is een hele mondvol die hier ongelukkig wordt afgebroken; wat je moet weten is dat je alleen na wget een spatie moet intypen. Vanaf https komen er geen spaties meer voor.

Pak vervolgens uit wat zojuist is gedownload:

unzip coco_ssd_mobilenet_v1_1.0_quant_2018_06_29.zip -d Sample_TFLite_model

En dat is het – we hebben nu alles geïnstalleerd wat we nodig hebben om te gaan spelen met objectdetectie en beeldclassificatie met behulp van TensorFlow Lite op de Raspberry Pi.

Nu kunnen we gaan experimenteren.

In de volgende aflevering: als verzamelaar van gegevens, vooral van historische afbeeldingen en video's die vaak nergens anders meer te vinden zijn, wil ik wat ik beschikbaar heb met de wereld delen. Maar ik zal nooit de tijd hebben om elke foto en elk videoframe door te nemen om metadata toe te voegen en de zoekmachines te vertellen wat ik heb. Het is dus mijn bedoeling om deze tools te gaan gebruiken om me te helpen bij het analyseren en classificeren van mijn 'Mount Media'. Die fase begint nu en ik zal je de volgende keer vertellen over mijn successen en mislukkingen!

230181-E-03

Vragen of opmerkingen?

Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar brian.williams@elektor.com.



Over de auteur

Brian Tristam Williams is al gefascineerd door computers en elektronica sinds hij op 10-jarige leeftijd zijn eerste 'microcomputer' kreeg. Zijn reis met Elektor Magazine begon toen hij op 16-jarige leeftijd zijn eerste nummer kocht en sindsdien volgt hij de wereld van elektronica en computers, onderzoekt en leert hij voortdurend. In 2010 begon hij bij Elektor te werken en tegenwoordig houdt hij zich graag bezig met de nieuwste trends op het gebied van technologie, waarbij hij zich vooral richt op kunstmatige intelligentie en singleboard computers zoals de Raspberry Pi.



WEBLINKS

- [1] Officiële website van TensorFlow Lite: https://tensorflow.org/lite
- [2] Raspberry Pi Imager download: https://raspberrypi.com/software
- [3] TensorFlow Lite Object Detection on Android and Raspberry Pi [GitHub]: https://tinyurl.com/edjetflite



262.144 manieren om Game of Life te spelen

lezersproject in het kort

Brian White (Verenigde Staten)

Ik ben al heel lang gefascineerd door Conway's Game of Life. Volg mij bij mijn poging om deze programmeerpuzzel uit de jaren '70 om te zetten in een boeiend visueel spektakel, met behulp van een RGB LED-matrix in combinatie met unieke coderingsmethoden waarmee we elke denkbare groep regels voor het spel kunnen simuleren!

Dit project vertelt het verhaal van vele ideeën, waarvan sommige al 50 jaar in mijn brein rondspoken en die nu eindelijk samenkomen in een zeer bevredigend resultaat. Het begint met Conway's Game of Life, waar ik eind jaren 1970 voor het eerst van hoorde. In die tijd volgde ik mijn eerste (en enige) programmeercursus – BASIC programmeren met behulp van een DECwriter-terminal aangesloten op een PDP-11 op mijn middelbare school – en het schrijven van code voor "The Game of Life" was een van de oefeningen. Het is een leuk voorbeeld van geneste lussen (FOR...NEXT in BASIC), een prima oefening om te beginnen met programmeren en het leverde boeiende veranderende patronen op. In de jaren die volgden, bleef het me fascineren toen computers – en dus ook Conway's spel – zich verplaatsten van printer-terminals naar CRT-monitoren naar laptops en iPads.

Game of Life

In het kort is Conway's Game of Life [1] (GoL) een eenvoudige regelgebaseerde cellulaire automaten-simulatie. Het wordt gespeeld op een rechthoekige matrix van willekeurige grootte. De speelstukken zijn cellen die elk één plaats innemen in de matrix. In het spel kunnen cellen van de ene generatie op de volgende blijven bestaan, sterven of geboren worden. De aan- of afwezigheid van een bepaalde cel in de volgende generatie hangt alleen af van hoeveel



Figuur 1. Een locatie in de 'wereld' met zijn acht buurlocaties.

buren die locatie heeft in de huidige generatie. In een rechthoekige matrix heeft elke locatie tussen nul en acht buren (**figuur 1**). De door Conway opgestelde regels luiden:

- een cel kan in de volgende generatie blijven bestaan als hij twee of drie buren heeft. Cellen met minder buren 'sterven aan eenzaamheid'; cellen met meer buren 'sterven door overbevolking';
- > een cel kan in de volgende generatie geboren worden op een lege plaats als die cel precies drie buren heeft.

Conway selecteerde deze regels na een periode van experimenteren zodat aan de volgende criteria werd voldaan (zoals beschreven in [2]):

- > er mag geen beginpatroon bestaan waarvan eenvoudig kan worden bewezen dat de bevolking onbeperkt kan groeien;
- er mogen geen initiële patronen bestaan zijn die kennelijk onbeperkt groeien;
- > er moeten eenvoudige beginpatronen bestaan die gedurende een aanzienlijke periode groeien en veranderen voordat ze op drie mogelijke manieren eindigen: volledig verdwijnen (door overbevolking of door te eenzaam te worden), zich vestigen in een stabiele configuratie die daarna onveranderd blijft, of een oscillerende fase ingaan waarin ze een eindeloze cyclus van twee of meer perioden doormaken.

Codeer de regels in zes octale cijfers

De oorspronkelijke regels van het Game of Life kunnen in het kort zo worden beschreven:

Een cel overleeft als hij 2 of 3 buren heeft, anders sterft hij. Een cel wordt geboren op een lege locatie met precies drie buren. Deze zelfde regels kunnen ook zo worden weergegeven, waarbij een '1' in de regel een bezette locatie in de volgende generatie (overleving of geboorte) aangeeft bij dat aantal buren en een '0' in de regel een onbezette locatie in de volgende generatie (gestorven of blijft leeg).



Het bitpatroon specificeert de toestand van de cel in de volgende generatie; de index in het bitpatroon is het aantal buren gecombineerd met de vraag of de cel momenteel bezet is of niet. Hieronder wordt de decodering van de regel 256020 als voorbeeld getoond:

Tabel 1. Interpretatie van de octale regelset "256020".

Oorspronkelijke locatie	Bezet						Niet-bezet											
Buren	8	7	6	5	4	3	2	1	0	8	7	6	5	4	3	2	1	0
Volgende generatie	0	1	0	1	0	1	1	1	0	0	0	0	0	1	0	0	0	0
Octaal		2			5			6			0			2			0	

Tekstueel staat hier:

Op dit moment overleven levende cellen als ze 1, 2, 3, 5 of 7 buren hebben; anders sterven ze. Er wordt een cel geboren op elke locatie die op dat moment onbezet is en precies 4 buren heeft.

Met dit coderingsformaat kan elk van de 262.244 mogelijke regelsets voor het Game of Life worden gespecificeerd.

In de originele versie uit 1970 werd het spel gespeeld met tastbare damstenen op een dambord. Kort daarna werd het proces geautomatiseerd; tot op de dag van vandaag vinden innovaties plaats [3].

Interessante algoritmen

Het volgende stukje van de puzzel was het boek "A New Kind of Science" van Stephen Wolfram, waarin hij verschillende cellulaire automaten-algoritmes besprak die interessante visuele patronen produceerden. Een van de belangrijkste onderdelen van Wolframs analyse was het coderen van de regels voor de generatie-naargeneratie overgangen in de vorm van binaire getallen. Door de regels op deze manier te coderen, is het mogelijk om de gevolgen van alle mogelijke regelsets op een goed gedefinieerde en herhaalbare manier te onderzoeken [4].

Dit zette me aan het denken over een manier om de regels van GoL

zo te coderen dat het mogelijk zou zijn om alle mogelijke regelsets te verkennen. Ik realiseerde me dat je de regels zou kunnen coderen in een 18-bits binair getal (zie het kader **Codeer de regels in zes octale cijfers** voor details). Elk van de negen 'lage' bits (o tot 8) geeft de toestand van de locatie in de volgende generatie aan (1 = bezet, 0 = onbezet) voor een onbezette locatie met nul tot acht buren – bit 0 geeft de toestand in de volgende generatie als de locatie momenteel nul buren heeft; bit 1 voor één buur enzovoort. Op dezelfde manier geven de 'hoge' negen bits (9 tot 17) de toestand in de volgende generatie voor een bezette locatie met nul tot acht buren – bit 9 voor nul buren; bit 10 voor één buur enzovoort. Met behulp van deze codering is het mogelijk om elke reeks regels te specificeren gebaseerd op het aantal buren dat een locatie heeft. Interessant genoeg is het niet nodig dat deze regels 'biologisch zinvol' zijn. Bijvoorbeeld regelset 256020, waar cellen blijven



bestaan als ze 1, 2, 3, 5 of 7 buren hebben. Deze regelset toont interessant gedrag, ook al is er geen biologische reden waarom 4, 6 en 8 buren tot overbevolking leiden zijn en 3, 5 en 7 buren niet. Aangezien alle mogelijke regelsets kunnen worden gecodeerd in 18 binaire bits, en omdat 2¹⁸ = 262.144, betekent dit dat er 262.144 manieren zijn om GoL te spelen. Dit maakt het mogelijk om een schakeling te ontwerpen waarmee een gebruiker al deze mogelijkheden kan onderzoeken en kan observeren hoe een populatie van elke verschillende 'soort' zich in de loop van de tijd ontwikkelt.

Het laatste deel van het softwareontwerp kwam voort uit vele gesprekken met de elektronische kunstenares Kelly Heaton [5], die ik leerde kennen via een artikel in Elektor Magazine [6]. Ze moedigde me aan om buiten de gebaande paden te denken en iets te maken dat niet alleen trouw is aan het algoritme maar ook visueel boeiend. Dit bracht me ertoe om het standaard kleurschema voor GoL-simulaties, waarbij bezette locaties gekleurd zijn en onbezette locaties zwart, aan te passen. Ik wilde de patronen een sterker gevoel van verandering geven, dus maakte ik pasgeboren cellen blauw; cellen die langer dan één generatie blijven bestaan zijn groen; en cellen die sterven laten een diep paarse 'geest' achter gedurende één generatie (deze 'geesten' worden niet als buren behandeld). Dit is gedetailleerd weergegeven in **figuur 2**.

De hardware

Het laatste onderdeel, de hardware, kwam als kerstcadeau van mijn zoons: een Adafruit Matrix Portal [7] en een 32×64 RGB LED-matrix [8].



Figuur 2. Toestandsdiagram voor meerkleurige celpatronen.

Ik monteerde alle onderdelen in een acryl behuizing zodat alle 'ingewanden' zichtbaar bleven. Het eindproduct is te zien in **figuur 3**. Links zie je drie sets van drie duimwielschakelaars die de regels en parameters voor elke run instellen. De bovenste en middelste set van drie schakelaars stellen de regels in als zes octale cijfers – elk octaal cijfer telt 3 binaire bits; vermenigvuldigd met 6 cijfers geeft dat de vereiste 18 binaire bits. De bovenste drie cijfers



Figuur 3. Het opgebouwde project.



Figuur 4. Het schema zonder Matrix Portal, 64×32-display en USB-voeding.

specificeren de regels voor bezette locaties; dus de regels voor welke cellen overleven in de volgende generatie. De middelste drie cijfers specificeren de regels voor onbezette locaties; dus de regels voor de geboorte van nieuwe cellen. De oorspronkelijke ('canonieke') regels van Conway zouden bijvoorbeeld worden uitgedrukt als 014010. Het kader Codeer de regels in zes octale cijfers beschrijft dit in meer detail. Onder het scherm, van links naar rechts, zie je een printje met de MCP23017 port expander, het Matrix Portal MCU-board, de resetknop en de schakelaar voor de displaymodus. Elke reeks begint met een willekeurige rangschikking van bezette en onbezette locaties en verloopt dan gedurende een vast aantal generaties voordat opnieuw wordt begonnen met een nieuwe willekeurig willekeurige verzameling cellen. De onderste drie cijfers specificeren de details van dit proces. Het linker cijfer bepaalt de dichtheid van organismen in de beginpopulatie; de kans dat een locatie bezet is bij de start is 10% maal de instelling van dit duimwiel. Een instelling van '3' betekent dus dat de wereld ruwweg voor 30% uit gevulde cellen bestaat. Omdat sommige regelsets interessantere resultaten geven met meer of minder bezette locaties, maakt dit het mogelijk om de effecten van variërende initiële celdichtheid te onderzoeken. Het middelste cijfer specificeert het aantal generaties dat uitgevoerd moet worden voordat er opnieuw gestart wordt. Hiervoor heb ik een exponentiële schaal gekozen om een breed scala aan runlengtes mogelijk te maken. Om precies te zijn loopt het proces gedurende 2×10^(N/2) generaties; een instelde waarde '4' resulteert dus in 200 generaties. Sommige patronen lossen snel op, terwijl andere nooit oplossen; deze instelling helpt om de weergave in een interessante staat te houden. Tenslotte specificeert het rechtse cijfer de vertraging in seconden tussen generaties; de waarde '0' geeft dus de maximale snelheid. Het vertragen van de generaties maakt het mogelijk om (als je dat wenst) de regels in detail te volgen. De laatste twee regelaars zijn een drukknop die de run onmiddellijk herstart en de populatie opnieuw rangschikt en een tuimelschakelaar om tussen het standaard-kleurschema te kiezen (blauw = bezet; blanco/uit = onbezet) of mijn meerkleurenschema voor de locaties in de matrix.

Zelfbouw

Het schema van **figuur 4** is heel eenvoudig. De Matrix Portal en het display worden aangesloten zoals beschreven in de instructies van Adafruit voor de MCU en de LED-matrix. De getoonde toevoegingen krijgen hun +3,3V-voeding en I²C-communicatie van

Onderdelenlijst

- Adafruit Matrix Portal CircuitPython Powered Internet Display (Adafruit 4745) 64x32 RGB LED Matrix – steek 5 mm (Adafruit 2277)
- 3 stuks 3-cijferige BCD-gecodeerde duimwielschakelaar
- (bijv. Littlefuse 3P-3-23-3-1-0-2)
- MCP23017 Port Expander
- 4 stuks pull-up weerstanden 2 k Ω
- Drukknop
- SPST-wipschakelaar
- USB-voedingsadapter
- Acryl-behuizing

de Matrix Portal via de Stemma QT-connector. Al het zware werk wordt gedaan door een MCP23017 I²C Port Expander. De diverse BCD-schakelaars (SW1...SW3, SW6...SW11) worden afzonderlijk aangestuurd via uitgangen van de MCP23017, waarna de BCD-uitgangen van de betreffende schakelaar worden ingelezen. Omdat de MCP23017 geen weak pull-down-optie heeft, worden de BCD-lijnen naar massa getrokken door een groepje losse weerstanden. De laatste twee ingangen van de MCP23017 worden gebruikt met weak pull-ups om de reset- en displaymodus-schakelaars in te lezen. De voeding wordt geleverd door een netadapter van 5 V bij 4 A die de Matrix Portal voedt via een USB-C kabel.

De code is gebaseerd op de GoL-code van Adafruit voor de Matrix Portal and Display [9] die enkele slimme trucjes gebruikt om de geneste lussen die GoL vereist heel snel uit te voeren. Ik heb de code aangepast om de regels te implementeren op basis van de getallen die zijn ingevoerd via de duimvielschakelaars in plaats van de 'standaard' regels. In het kort: de binaire waarden van de duimwielschakelaars worden ingelezen en gebruikt om een lijst van 18 elementen met enen en nullen te maken die de status van de locatie in de volgende generatie voorstelt, afhankelijk van de huidige bezetting van de locatie en het aantal niet-lege buren (die geen 'geest' zijn). Ik tel dan de buren en gebruik die telling, met 9 daarbij opgeteld als de locatie momenteel bezet is, en gebruik dat als index in de lijst met regels. Ik implementeer ook het meerkleurenpatroon dat hierboven is beschreven. De code is beschikbaar op de Elektor Labs-projectpagina [10].

En verder...

De resultaten zijn fascinerend. Ik heb een YouTube-afspeellijst [11] gemaakt met een verzameling korte video's die de patronen laten zien die ontstaan bij verschillende instellingen van de schakelaars. Het is opmerkelijk hoe sommige veranderingen in de regels een groot effect hebben, terwijl andere slechts subtiele veranderingen veroorzaken in hoe de patronen groeien, veranderen, verschuiven en uitsterven. Ik ben nog maar net begonnen met het inventariseren van de mogelijkheden (262.144 is niet bepaald weinig!), maar hier zijn een paar dingen die me zijn opgevallen:

- > het instellen van de 'laagste' bits (dus de bits 0 en 9) waarmee geboorten en/of overleving met nul buren wordt toegestaan – leidt tot dramatische positief/negatieve omkeringen in elke generatie;
- het instellen van 'hogere' bits het controleren van geboorten en overleving met hogere aantallen buren – heeft meestal

minder effect, misschien omdat locaties met veel buren zeldzamer zijn;

- > de oorspronkelijke regelset, 014010, ontwikkelt zich meestal tot een aantal relatief stabiele structuren of oscillerende figuren. Andere regels, zoals 114110, zijn favoriet bij mijn vrouw omdat ze nooit lijken op te lossen;
- > je kunt regels maken waarbij structuren worden gecreëerd die geleidelijk op interessante manieren worden gevuld als je overlevingskansen toestaat voor veel buren en geboorten bij slechts een paar, bijvoorbeeld 256020.

Op dit moment staat het naast onze keukentafel en ik zie vaak dat iemand in huis een nieuwe regelset heeft gevonden die een fascinerend nieuw patroon oplevert.

Omdat we hiermee hebben gespeeld en het met anderen hebben gedeeld, hebben we verschillende ideeën opgedaan voor wie overweegt er zelf een te bouwen. Ten eerste zou het interessant zijn om meer kleuren toe te voegen aan het meerkleurenschema. De kleuren zouden bijvoorbeeld geleidelijk kunnen veranderen naarmate de cellen 'rijpen'. Kelly Heaton stelde verder voor om de kleuren in elkaar te laten overvloeien in plaats van abrupt te veranderen voor een meer vloeiend effect. Tot slot stelden de leden van mijn huishouden die geen octaal spreken voor om een manier te bedenken om de overeenkomst tussen de bits in de regels en de aantallen buren duidelijker te laten zien. Je zou een rij van 18 LED's kunnen toevoegen om het binaire equivalent van de octale instellingen weer te geven of, nog eenvoudiger, 18 individuele schakelaars gebruiken, één voor elke bit in de regelinstellingen. Ik hoop dat lezers van dit artikel geïnspireerd worden om een versie van dit project te bouwen, met een speciaal scherm en MCU, of helemaal op de computer, en de regelsets die ze interessant vinden te delen. Wie had een idee dat het delen van 6-cijferige octale getallen zo interessant kon zijn? 📕

230258-03



Over de auteur

Brian White is hoogleraar biologie aan de Universiteit van Massachusetts in Boston. Hij heeft een achtergrond in biologie van MIT en Stanford. Als docent doet hij ook onderzoek naar biologie en biologieonderwijs en ontwikkelt hij gerelateerde software. Brian is daarnaast elektronicus, muzikant en radioamateur (KA1TBQ). Meer informatie over zijn elektronicaprojecten is te vinden op [12].

Vragen of opmerkingen?

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



Gerelateerde producten

- > MAX7219 Dot Matrix Module (Set of 8) www.elektor.nl/18422
- > Adafruit Feather RP2040 www.elektor.nl/19689
- > ESP32-C3-DevKitM-1 www.elektor.nl/20324



WEBLINKS

- [1] Wikipedia, Conway's Game of Life: https://nl.wikipedia.org/wiki/Game_of_Life
- [2] Martin Gardner, "MATHEMATICAL GAMES: The fantastic combinations of John Conway's new solitaire game 'life," Scientific American 223 (October 1970): 120-123: https://web.stanford.edu/class/sts145/Library/life.pdf
- [3] LifeWiki: https://conwaylife.com/wiki
- [4] Stephen Wolfram, A New Kind of Science, en dan vooral dit gedeelte: https://wolframscience.com/nks/p53--more-cellular-automata
- [5] Kelly Heaton Studio: https://kellyheatonstudio.com
- [6] C.J. Abate, "Kunst maken met elektriciteit: een vraaggesprek met Kelly Heaton", Elektor Summer Circuits 2022: https://www.elektormagazine.nl/magazine/elektor-263/60817
- [7] Matrix Portal CircuitPython-Powered Internet Display: https://adafruit.com/product/4745
- [8] 64×32 RGB LED Matrix steek 5 mm: https://adafruit.com/product/2277
- [9] Adafruit-voorbeeld: Conway's "Game of Life":
- https://learn.adafruit.com/rgb-led-matrices-matrix-panels-with-circuitpython/example-conways-game-of-life
- [10] Projectpagina op Elektor Labs: https://elektormagazine.com/labs/262144-ways-to-play-the-game-of-life
- [11] Game of Life video's: https://youtube.com/playlist?list=PL2wCLIpn1MkRpBHeZ2svMs-CdLvf2B_9N
- [12] Website van de auteur: https://brianwhite94.wixsite.com/electronics





gegrepen

de Chinese draak

Ilse Joostens (België)

Fijn Chinees kraakporselein met blauwwitte motieven was populair in Nederland en het werd sinds het begin van de 17e eeuw door de Vereenigde Oostindische Compagnie, die ook een kamer had te Delft, geïmporteerd. Dit porselein werd vooral als siervoorwerp gebruikt in rijke huishoudens en toen de import na de val van de Ming-dynastie in 1644 stilviel namen de Delftse plateelbakkerijen de productie over met goedkopere kopieën in aardewerk met in het begin typische Ming-achtige motieven op wit tinglazuur. Goed 380 jaar later lijken de rollen omgekeerd en is het nu China dat onze producten kopieert.

Kopietje nodig?

Uiteraard was Delfts blauw [1] meer dan kopieën van Chinees porselein en de Delftse keramiekschilders wisten niet alleen de kwaliteit van hun producten tot grote hoogte te brengen maar zorgden ook met nieuwe technieken steeds voor verrassingen. China daarentegen kopieert het Westen erg letterlijk met ondermeer de in 2019 geopende onderzoekscampus van het Chinese technologiebedrijf Huawei in de stad Dongguan, die eruit ziet als een fake Europa [2]. Het treintje dat stations als Heidelberg, Bologna en Granada aandoet lijkt verdacht op dat van de Zwitserse Jungfrau Railway. Eerder werden reeds delen van Parijs, zoals de Champs-Élysées, en het pittoreske Oostenrijkse dorpje Hallstatt [3] nagebouwd als toeristische attractie. Helaas gaat het kopiëren veel verder, van alle mogelijke producten tot kunstwerken en Chinese winkels die de look, feel en service van succesvolle Westerse retailconcepten repliceren zoals een welbekend Zweeds woonwarenhuis of winkels van een computermerk met een half opgegeten pitvrucht als logo.

Deze winkels zijn na enig juridisch getouwtrek opgedoekt, maar het kopiëren van producten en ideeën gaat gewoon verder. Nog niet zo lang geleden viel ik bijna van mijn stoel toen ik bij Ome Ali een twee meter hoge replica van de sculptuur Kindred Spirits van Alex Pentek [4] zag staan. Ook de elektronicasector blijft niet gespaard. Velen zijn bekend met namaak-IC's en gerecycleerde componenten die als nieuw worden verkocht. Naast dubieuze onderdelen kun je alle mogelijke elektronische modules, kits, apparaatjes, gereedschappen en machines van klein tot groot tegen zeer lage prijzen bestellen. Zelfs nicheproducten worden in het vizier genomen en ook Dalibor Farný, de Tsjechische maker van de R|Z568M-nixiebuizen, moet stilaan vrezen voor duchtige concurrentie.

In 2016 heb ik een artikel voor Elektor rond 7-segment displays geschreven [5], op basis van de toen nog vrij nieuwe LED-filamenten. Blijkbaar kan je nu bij Ome Ali kits bestellen voor klokjes met LED-filamenten en dat voor prijzen waarvoor ik zelfs de onderdelen nog niet bij elkaar geharkt krijg. Ook de verzendkosten zijn een schijntje omdat China bij de Universal Postal Union de status van ontwikkelingsland heeft waardoor zij veel goedkoper kunnen verzenden [6]. Het is die organisatie blijkbaar ontgaan dat China met succes een ruimtetuig op de achterkant van de maan heeft neergezet. Ik heb er geen probleem mee dat een idee gebruikt wordt, maar met je eigen idee uit de markt geconcurreerd worden is wel pijnlijk.

Tofu-dreg

Zoals in zoveel landen komt corruptie, omkoping, liegen en bedriegen ook in China in alle lagen van de maatschappij voor, ook al is de meerderheid van de Chinezen vriendelijk en correct. De problemen die dit met zich meebrengt zijn niet min zoals gebouwen opgetrokken uit minderwaardige materialen





Children and a second s

Figuur 1. Goedkope netadapter – componentenzijde...

met regelmatig instortingen tot gevolg. Tofu-dreg zijn de resten welke overblijven na het bereiden van tofu en is tevens slang voor slecht uitgevoerde projecten en broddelwerk [7]. Naast gevaarlijke infrastructuur en zelfs namaakvoedsel, kampt ook de elektronicasector met deze problemen en veel Chinese consumentengoederen, welke hier geïmporteerd en verkocht worden, zijn niet alleen van slechte kwaliteit maar soms zelfs levensgevaarlijk.

Een constante zijn goedkope netadapters en USB-laders opgebouwd met een minimaal aantal onderdelen. Enige isolatieafstand tussen het primaire en secundaire deel is quasi onbestaand en een stuk dun printspoor dient als zekering. Het is dan ook niet zo vreemd dat

Figuur 2. ...en koperzijde (figuren 1 en 2: foto's Leo Potjewijd).

zo nu en dan iemand in bad geëlektrocuteerd wordt door een smartphone. Het lijkt een *fait divers* in de media en buiten een algemene waarschuwing om je smartphone niet met de lader aangesloten te gebruiken wordt er geen aandacht besteed aan de onderliggende oorzaken. Recent zag ik op een forum een zoveelste foto van de ingewanden van zo'n netadapter (**figuur 1** en **figuur 2**). Buiten de genoemde gebreken bleek de optocoupler op de print geen enkele functie te vervullen als je goed naar de printsporen kijkt. Een onderdeel teveel dus, rare jongens en meisjes, die Chinezen...

Ook van de zoutlamp die ik met Kerst cadeau kreeg werd ik niet vrolijk. Deze was dimbaar maar flikkerde ergerlijk als de dimmer niet

Figuur 3. Het lampje uit de zoutlamp van Chinese makelij.

op het maximum stond en ik heb alles maar uit elkaar gehaald en gereviseerd. Het LED-lampje (figuur 3) bestond intern uit niet meer dan uit een bruggelijkrichter, een IC wat gloeiend heet werd en een led-matrix, zelfs geen afvlakcondensator. De behuizing van de dimmer (figuur 4) kon je zonder gereedschap uit elkaar halen en het netsnoer was niet meer dan verkoperd aluminium met een doorsnede van nog geen tiende vierkante millimeter per geleider (figuur 5). In combinatie met een neppe Chinese zekeringautomaat [8] zijn de betere pyrotechnische effecten bijna gegarandeerd (figuur 6). Het is niet altijd makkelijk, maar wat meer lokaal geproduceerde spullen kopen en onze eigen mensen iets gunnen lijkt me nog niet zo slecht.

230609-03



Figuur 4. De dimmer van de LED-zoutlamp.



Figuur 5. Het akelig dunne netsnoer van de zoutlamp...



Figuur 6. ... werd (bij 6 A) wel heel erg akelig heet (bij Sp2 op de foto bijna 118 °C).

- [1] Wikipedia: Delfts Blauw: https://nl.wikipedia.org/wiki/Delfts_blauw
- [2] The Atlantic: Photos of Huawei's European-Themed Campus in China: https://www.theatlantic.com/photo/2019/05/photos-of-huaweis-european-themed-campus-in-china/589342/
- [3] Wikipedia: Hallstatt (China): https://en.wikipedia.org/wiki/Hallstatt_(China)
- [4] Wikipedia: Kindred Spirits (sculpture): https://en.wikipedia.org/wiki/Kindred_Spirits_(sculpture)
- [5] Elektor Magazine: LEDitron Ledfilamenten als 7-segmentdisplay: https://www.elektormagazine.nl/magazine/elektor-201605/28993
- [6] RTL Nieuws: Waarom AliExpress zo goedkoop pakketjes kan versturen:
 https://www.tlpiouwo.pl/toop/artikal/4760016/bastalling_pakket_aliovpress
- https://www.rtlnieuws.nl/tech/artikel/4760916/bestelling-pakket-aliexpress-china-goedkoop-oneerlijke-concurrentie
- [7] YouTube: China Insights Fragile steel bars/Tofu-dreg project in China/Shaky building/Collapsing buildings/Poor quality: https://www.youtube.com/watch?v=s-2DtL-Wjkc
- [8] YouTube: bigclivedotcom Inside a fake un-trippable circuit breaker: https://www.youtube.com/watch?v=2TJEzdqtXIQ

Draaien met die (DC-borstel) motor!

voorbeeldprojecten uit de Elektor Motor Control Development Bundle

Dogan Ibrahim (Verenigd Koninkrijk)

Er is geen betere introductie voor programmeren en praktische elektronica dan het bestuderen van hoe je dingen kunt laten bewegen, knipperen of geluid laten maken. Voor beweging is vaak een of andere motor nodig en een kleine DC-borstelmotor is ideaal om kennis op te doen voordat je begint aan serieuze robotica en mechatronica. We zetten de eerste stappen in de richting van intelligente DCmotorbesturing met behulp van een ontwikkelkit van Elektor die hardware, software en een degelijke handleiding combineert.

Figuur 1. De hoofdrolspelers van het project. **Opmerking van de redactie:** dit artikel is een deel van de 192 pagina's tellende Elektor-uitgave Motor Control Development Kit. Deze maakt deel uit van de Motor Control Development Bundle van Elektor. Dit gedeelte is opgemaakt en ietwat aangepast aan de conventies en pagina-indeling van Elektor Magazine. Auteur en redacteur helpen je graag met vragen. Contactgegevens vind je in het kader Vragen of opmerkingen.





In dit artikel worden drie eenvoudige projecten met DC-motoren beschreven op basis van een kleine permanent-magneet DC-borstelmotor die wordt aanstuurd door het Cytron/Maker Pi RP2040 development board. Beide producten zitten in de *Motor Control Development Bundle* die verkrijgbaar is in de Elektor-shop [1]. De projecten zijn meer experimenteel en educatief bedoeld dan als volledig uitgewerkte voorbeelden.

DC-motor: aan/uit-regeling

Dit is een basisproject dat laat zien hoe een kleine DC-borstelmotor (die op een gelijkspanning van 1...6 V draait) kan worden aangesloten op een van de DC MOTOR-printkroonstenen van de Maker Pi RP2040. De motor wordt 5 seconden in de ene richting geschakeld en staat dan 5 seconden stil. Vervolgens wordt hij 5 seconden in de andere richting geschakeld en weer 5 seconden gestopt. Dit proces wordt eindeloos herhaald totdat het handmatig wordt onderbroken.

Het doel van dit project is om het principe te tonen van de aansluiting, werking en aansturing van een DC-borstelmotor vanaf het Maker Pi RP2040 de velopment board. **Figuur 1** toont het 'blokschema' van het project en **figuur 2** de hier gebruikte motor. Het is een kleine DC-borstelmotor die is gemaakt voor spanningen van +1 V tot +6 V en die een aanbevolen bedrijfsspanning van +3 V heeft. Laten we eens kijken of we hem in beweging krijgen.

Op het development board zit een tweekanaals H-brug motordriver, die maximaal twee DC-borstelmotoren, M1 en M2, of een stappenmotor kan aansturen. De I/O pinnen zijn als volgt toegewezen:

- M1A: GP8
 M1B: GP9
 M2A: GP10
- > M2B: GP11



Figuur 2. De door ons gebruikte kleine DC-motor.

.....

Listing 1: DCMotor1.py

Figuur 3. Zo wordt de motor aangesloten op het RP2040 Maker Pi development board.



De waarheidstabel van de motordriver is te zien in **tabel 1**. In dit project worden de motoringangen PWM1A (GP8) en PWM1B (GP9) gebruikt met de motoruitgangen op M1A en M1B. Deze uitgangen (*Output A*) zijn beschikbaar op de MOTOR 1 printkroonstenen middenboven op het development board. Zoals te zien is in tabel 1, wordt de motor aangestuurd via de poorten GP8 en GP9. Als je bijvoorbeeld GP8 hoog zet en GP9 laag, draait de motor de ene kant op. Evenzo draait de motor de andere kant op als je GP8 laag en GP9 hoog maakt.

Figuur 3 toont het project inclusief de op de MOTOR 1-uitgangen aangesloten motor.

Het programma waarmee je meteen aan de slag kunt staat in **listing 1**. Het draagt de naam *DCMotor1.py* en maakt deel uit van een groot software-archief dat gratis beschikbaar is op de Elektor-website [1]. Kijk op die pagina onder *Downloads* → *Software_Motor Control Development Kit*.

De motor wordt aangestuurd met een PWM-signaal zoals elders in de *Guide* beschreven. De ingebouwde DC-motorbibliotheek met de naam *adafruit_motor* wordt in dit project gebruikt. Aan het begin van het programma worden de bibliotheekmodules *board*, *time*, *pwmio* en *motor* in het programma geïmporteerd. De poorten GP8 en GP9 van Motor 1 worden vervolgens geconfigureerd om met een PWM-signaal van 50 Hz te werken. De rest van het programma draait in een oneindige lus.

De throttle class-eigenschap in de software kan een waarde aannemen tussen -1 en +1 en regelt de motor als volgt:

throttle = 0 motor uit
throttle = 1 motor draait op volle snelheid vooruit
throttle = 0.5 motor draait vooruit op 50% van zijn
maximale snelheid
throttle = -1 motor draait op volle snelheid achteruit
throttle = -0.5 motor draait achteruit op 50% van
zijn maximale snelheid

In dit programma draait de motor op 25% van zijn maximale snelheid door throttle in te stellen op +0,25 in de ene richting en -0,25 in de andere richting.

4	
BRUSHED DC MOTOR CONTRO	DL
In this program a brushed DC moor is The motor is turned ON in forward diru and then stopped for 5 seconds, then for 5 seconds and then stopped again process is repeated forever	controlled as follows: ection for 5 seconds in reverse direction for 5 seconds. This
Author: Dogan Ibrahim	
File : DCMotor1.py	
Date : February, 2023	
ŧ	
Import board	
Import time	
import pwinto	
+	
t Initialize DC Motor 1	
nia = pwmio.PWMOut(board.GP8. frequency	=50)
<pre>nlb = pwmio.PWMOut(board.GP9, frequency)</pre>	=50)
notor1 = motor.DCMotor(m1a, m1b)	
while True:	
<pre>motor1.throttle = 0.25</pre>	# 25% of full speed
<pre>time.sleep(5)</pre>	
<pre>motor1.throttle = 0</pre>	# Idle
<pre>time.sleep(5)</pre>	
<pre>motor1.throttle = -0.25</pre>	# 25% of full speed
time.sleep(5)	
<pre>motor1.throttle = 0</pre>	# Idle
<pre>time.sleep(5)</pre>	

Tabel 1. Waarheidstabel van de motordriver.

Input A	Input B	Output A	Output B	Motoractie
Laag	Laag	Laag	Laag	remt
Hoog	Laag	Hoog	Hoog	voorwaarts
Laag	Hoog	Laag	Hoog	achterwaarts
Hoog	Hoog	Hi-Z (open)	Hi-Z (open)	vrijloop

Listing 2: DCMotor2.py.





DC-motorregeling met twee snelheden

Dit is opnieuw een heel eenvoudig project waarbij een kleine DC-borstelmotor wordt aangesloten op het Maker Pi RP2040 development board. De drukknop op de print (aangesloten op poort GP20) wordt ook gebruikt in het project. Normaal draait de motor op 25% van zijn maximale snelheid. Als je op de knop drukt, draait hij op 50% van de maximale snelheid.

De 'ingrediënten' en de aansluitingen van de motor en het development board voor dit project zijn hetzelfde als in het vorige project.

Listing 2 toont het programma *DCMotor2.py*. Het hoofdprogramma draait in een lus, waarin de status van de knop op GP20 wordt gecontroleerd. Als de knop wordt ingedrukt, wordt de motorsnelheid verhoogd tot 50% van de maximale snelheid.

De snelheid van een DC-motor meten met een draai-encoder

Dit project laat zien hoe een draai-encoder kan worden gebruikt om de snelheid van een DC-motor te meten. De snelheid wordt vervolgens weergegeven op het display. In de 'diagramweergave' ziet dat eruit als in **figuur 4**. De motor met ingebouwde draai-encoder is niet inbegrepen in de Motor Control Development Kit en moet apart worden aangeschaft. Hij is onder andere bij eBay en AliExpress verkrijgbaar.

We gebruiken een kleine DC-borstelmotor met tandwieloverbrenging die over een ingebouwde draai-encoder beschikt. **Figuur 5** toont een foto van de motor waar de draai-encoder aan de achterzijde is bevestigd. In dit project wordt de volgende motor gebruikt: 6V-motor met vertraging, 2 W, type GBMQ-GM12BY20, met encoder, 70 RPM. Het opgegeven onbelaste toerental van de motor is 70 RPM. In dit project gebruiken we een externe +5V-voeding voor de motor en daarom kun je verwachten dat de onbelaste snelheid minder dan 70 RPM bedraagt. Let op dat het stroomverbruik van de motor kan oplopen tot 200 mA.

Een draai-encoder (of as-encoder) is een sensor die de hoekpositie van iets roterends (zoals een motor) omzet in een elektrisch signaal. Draai-encoders worden gebruikt in motorbesturingen om de snelheid van de motor of de positie van de motoras te meten. Er zijn in principe twee typen draai-encoders: optische encoders en encoders op basis van het Hall-effect. De eerste werken volgens optische principes, waarbij licht door sleuven (of gaten) in een (metalen of andere) schijf op een fotodiode valt. Door het aantal openingen te tellen dat in een bepaalde tijd voor de fotodiode passeert, kun je (lees: je microcontroller) de snelheid van de motor berekenen.

In dit project gebruiken we een Hall-effect draai-encoder. Twee statische magneetsensoren (fase A en fase B genoemd) worden op de encoderprint samen met een roterende magnetische schijf gebruikt. De sensoren leveren pulsen als de motor draait. Door de pulsen gedurende een bepaalde tijd te tellen, kun je de snelheid van de motor berekenen. Figuur 6 toont de uitgangsgolfvormen van de motor die in dit project wordt gebruikt. De bovenste golfvorm vertegenwoordigt fase A en de onderste fase B. Dit type encoder wordt ook wel kwadratuurencoder genoemd omdat de magnetische sensoren in een hoek van 90° ten opzichte van elkaar zijn geplaatst en er vier mogelijke uitgangstoestanden zijn. De draairichting kan eenvoudig worden bepaald uit de volgorde waarin de beide signalen van 0 naar 1 veranderen. Als het van fase A-signaal van laag naar hoog gaat terwijl fase B laag is, dan draait de motor in de ene richting. Als daarentegen het fase B-signaal van laag naar hoog gaat terwijl het fase A-signaal laag is, dan draait de motor in de andere richting - zie figuur 7.

De specificaties van de motor die in dit project wordt gebruikt zijn als volgt (afhankelijk van de aangelegde spanning):

- > voedingsspanning 6 V DC
- > draaisnelheid na vertraging: 70 RPM (bij 6 V)
- > vermogen: 2 W
- > stroom 170...200 mA
- > twee ingebouwde Hall-effect sensoren
- > gewicht: 14 gram

Aan de achterzijde van de motor zit een 6-polige connector. De pinning van deze connector is als volgt:

zwart	motorvoeding GND
rood	motorvoeding (+5 V in dit project)
geel	encodervoeding (+3,3 V in dit project)
groen	encodervoeding GND
blauw	encoderuitgang fase A
wit	encoderuitgang fase B

Figuur 8 toont het bedradingsschema voor dit project. Let op dat de motor wordt gevoed uit een externe +5V-voeding die tot 500 mA moet kunnen leveren. De encoder werkt op de +3,3V-aansluiting van het development board, compatibel met het ingangsspanningsniveau van de RP2040-processor. In dit project wordt ook alleen de fase-A-uitgang van de motor gebruikt.

Helaas vermelden fabrikanten en distributeurs van elektromotoren niet de overbrengingsverhouding en het aantal encoderpulsen dat de motor per seconde levert. Dit zijn echter belangrijke parameters. Daarom schrijven we hier een programma om deze belangrijke parameters van de gebruikte motor weer te geven. Het kan zijn dat je deze berekeningen moet uitvoeren om de belangrijkste specificaties van de gebruikte motor te achterhalen. Listing 3 toont het programma MotorPulses.py dat is geschreven om elke seconde het aantal motorimpulsen weer te geven dat wordt uitgevoerd door de fase A encoder.



Figuur 6. Uitgangsgolfvormen van een draai-encoder.





In dit programma wordt poort GP6 toegewezen aan het object Encoder en geconfigureerd als ingang, en het aantal pulsen dat wordt ontvangen van de encoder wordt berekend en weergegeven aan het einde van elke seconde. Let op dat dit programma de functie time.monotonic() gebruikt voor de timing, en dat is misschien niet erg nauwkeurig. In het ideale geval zouden we de pulsen als externe interrupts ontvangen en één-seconde timer-interrupts gebruiken om elke seconde het aantal ontvangen pulsen te tellen en weer te geven. Helaas ondersteunt CircuitPython geen externe of timer-interrupts.

Figuur 8. De bedrading van het project.

Listing 3: MotorPulses.py

#							
# DISPLAYING THE NUMBER OF ENCODER PULSES							
#							
# In this program a geared DC moto	or with Hall Effect sensors						
# is connected to the Maker Pi. The	# is connected to the Maker Pi. This program calculates						
<pre># and displays the number of pulse</pre>	s received from the encoder						
# every second. Only Phase A encoder output is used here							
#							
# Author: Dogan Ibrahim							
<pre># File : MotorPulses.py</pre>	<pre># File : MotorPulses.py</pre>						
# Date : March, 2023							
#							
import board							
import digitalio							
import time							
Encoder = digitalio.DigitalInOut(board.GP6)						
Encoder.direction = digitalio.Dire	ection.INPUT						
<pre>while Encoder.value == 0:</pre>	# Wait while 0						
pass							
<pre>strtime = time.monotonic()</pre>	<pre># Start time,rising edge</pre>						
	# detected						
count = 0	# Intialize count						
#							
# Main program loop							
#							
while True:							
<pre>while Encoder.value == 1:</pre>	# Wait while 1						
pass							
<pre>while Encoder.value == 0:</pre>	# Wait while 0						
pass							
<pre>endtime = time.monotonic()</pre>	# End time						
if endtime-strtime > 1.0:	# Just over a second						
print(count)	# Display count						
<pre>strtime = time.monotonic()</pre>)# Start time						
count = 0							
else:							
count = count + 1	# Increment count						
<pre>while(Encoder.value) == 1:</pre>	# Wait while 1						
pass							

Listing 4: MotorSpeed.py

#	
# DISPLAYING THE MOTOR SPEED	
#	
# This program displays the motor spee	d in RPM using the
<pre># number of encoder pulses received ev</pre>	ery second and the
# formula given in the text	
#	
# Author: Dogan Ibrahim	
# File : MotorSpeed.py	
# Date : March, 2023	
#	
import board	
import digitalio	
import time	
Encoder = digitalio.DigitalInOut(board	.GP6)
Encoder.direction = digitalio.Directio	n.INPUT
while Encoder.value == 0:	Wait while 0
pass	
<pre>strtime=time.monotonic()</pre>	# Start time
count=0	# Intialize count
#	
# Main program loop	
#	
while True:	
<pre>while Encoder.value == 1:</pre>	# Wait while 1
pass	
while Encoder.value == 0:	# Wait while 0
pass	
endtime=time.monotonic()	# End time
if endtime-strtime > 1.0:	# Just over a second
RPM = count * 60 / 880	# Motor speed
<pre>print("Speed=%6.2f RPM" %RPM)</pre>	<pre># Display speed</pre>
<pre>strtime=time.monotonic()</pre>	# Start time
count=0	
else:	
count=count+1	# Increment count
while(Encoder, value) == 1*	# Wait while 1
nass	a nore miller
Pubb	

CircuitPython REPL Speed= 63.95 RPM Speed= 64.02 RPM Speed= 63.95 RPM Speed= 63.89 RPM Speed= 63.95 RPM

Gerelateerde producten

Motor Control Development Bundel www.elektor.nl/20534

Figuur 9. Uitvoer van het programma.

◀

Figuur 10: De digitale fototachometer DT-2234C.

►





Als de motor draait, geeft het programma *MotorPulses. py* 938 pulsen/seconde weer. We hebben een miniatuurautoband op de motoras bevestigd en van een markering voorzien zodat de omwentelingen extern geteld konden worden. Het aantal omwentelingen van de motoras in onbelaste toestand bleek 64 RPM te zijn.

Per minuut worden $938 \times 60 = 56.280$ pulsen ontvangen, en dit komt overeen met een onbelast toerental van 64 RPM. Daaruit volgt 56.280/64 = 879,38 (880 is het dichtstbijzijnde gehele getal) en dan kan het motortoerental als volgt worden berekend met de volgende formule:

toerental (RPM) = (gemeten aantal pulsen per seconde × 60) / 880

Als de pulstelling bijvoorbeeld 450 per seconde bedraagt, dan is het motortoerental:

toerental = 450 × 60 / 880 = 30,68 RPM

We gebruiken deze formule om het motortoerental te berekenen.

Het programma *MotorSpeed.py* van **listing 4** wordt gebruikt om het motortoerental te berekenen en weer te geven aan de hand van bovenstaande formule. Dit programma is in essentie hetzelfde als in dat van **listing 3**, maar hier wordt de motorsnelheid berekend en weergegeven op het display. **Figuur 9** geeft een voorbeeld van de uitvoer van het programma.

Zoals eerder opgemerkt, hadden nauwkeuriger resultaten kunnen worden verkregen als externe interrupts waren gebruikt om de encoderpulsen te tellen, en timer-interrupts om de tijd te meten.

Zeker weten, RP2040?

Het motortoerental kan eenvoudig worden gemeten en gecontroleerd met een digitale foto-tachometer. Daarvan zijn er talloze op de markt, voor elk budget. De auteur gebruikte het model DT-2234C (**figuur 10**) dat ongeveer € 9 kostte (plus de prijs van een 9 V 'blokje'). Voordat je deze meter gebruikt, moet je een stuk reflecterend papier op de motoras of het wiel bevestigen, zoals in **figuur 11**. Het motortoerental wordt met de DT-2234C tachometer als volgt gemeten:

- > laat de motor draaien;
- druk op de TEST-knop van de toerenteller en richt de lamp van het apparaat op het reflecterende papier;
- het toerental wordt continu op het LC-display weergegeven;
- je kunt de metingen opslaan door op de MEM-knop te drukken.

Opmerking: in dit project wordt slechts één fase-uitgang van de enoder gebruikt (fase A). Het toerental kan



Reflective paper

Figuur 11. Plak een stukje reflecterend papier op het wiel.

nauwkeuriger worden gemeten als beide fase-uitgangen (dus fase A en fase B) worden gebruikt. Het is ook mogelijk om de motor aan te sturen via de MOTOR 1 of MOTOR 2 schroefaansluitingen van het Maker Pi RP2040 development board. Dit beperkt de maximale motorspanning tot +3,3 V en vereist ook dat de motorsnelheid door een PWM-signaal via de poorten GP8/GP9 (MOTOR 1) of GP10/GP11 (MOTOR 2) wordt geregeld. *vertaling: Hans Adams* – 230506-03

Vragen of opmerkingen?

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

Over de auteur

Dogan Ibrahim heeft een BSc (Hons) in Elektronic Engineering, een MSc in Automatic Control Engineering en een PhD in Digital Signal Processing and Microprocessors. Dogan heeft in veel organisaties gewerkt en is Fellow van de Institution of Engineering and Technology (IET) in het Verenigd Koninkrijk en is erkend elektrotechnisch technicus. Dogan is auteur van meer dan 100 technische boeken en meer dan 200 technische artikelen over elektronica, microprocessoren, microcontrollers en aanverwante gebieden. Dogan is gecertificeerd Arduino-professional en heeft vele jaren ervaring met bijna alle soorten microprocessoren en microcontrollers.

WEBLINK

[1] Motor Control Development Bundle: http://www.elektor.nl/motor-control-development-bundle





ESP32/RS-232-adapter

draadloze verbinding voor klassieke meetinstrumenten

Georg Hofstetter (Duitsland)

RS-232 kan ook draadloos! Een moderne en goedkope ESP32-module geeft een seriële interface vleugels. Naast de communicatiemogelijkheden via een seriële verbinding stelt deze kleine adapter SCPI-testapparatuur in staat om commando's te ontvangen en gegevens naar het thuisnetwerk te sturen via MQTT.

Als je in de loop der jaren aan verschillende elektronicaprojecten hebt gewerkt, is de stapel testapparatuur op je werkbank gegroeid – dat kan niet anders. In mijn geval gebruik ik een Keithley SourceMeter 2400 om fouten in elektronische apparaten op te sporen, ze in bedrijf te stellen of zelfs de eigenschappen van verschillende componenten te meten. Het is ook een geweldig instrument voor het testen van de lekstroom van elektrolytische condensatoren of het meten van hun nominale capaciteit. Een ander instrument in mijn verzameling is een AOR AR5000-ontvanger, die ik gebruik om in contact te blijven met radioamateurs waar ook ter wereld.

Beide apparaten kunnen, net als vele andere in het lab, bestuurd worden door een PC dankzij hun ingebouwde seriële RS-232-interfaces. De beschikbaarheid van USB/RS-232adapterkabels betekent dat aansluiting op een normale PC vrij eenvoudig zou moeten zijn. In mijn geval werkte de adapterkabel prima met de AR5000, maar de communicatie met de Keithley SourceMeter was merkbaar minder stabiel. Er gingen af en toe karakters verloren tijdens het verzenden van SCPI-commando's en de verbinding viel regelmatig uit.

De RS-232-verbinding

De RS-232-standaard, vastgelegd in de jaren 1960, wordt gebruikt om seriële databits van een karakter te verzenden met behulp van veranderingen in spanningsniveaus. De elektrische eigenschappen werden gedefinieerd in V.28, terwijl de overkoepelende functionaliteit geregeld was in V.24. De V.28 standaard bepaalt dat, afhankelijk van de logische bitwaarde, een logische '0' wordt verzonden als een tot 5...12 V stijgende flank en voor een logische '1' als een tot –5...–12 V dalende flank. De ontvanger heeft een iets breder spanningsvenster en zal een signaal in het bereik van 3...12 V interpreteren als een logische '0' en in een bereik van –3...–12 V als een logische '1'. Hierbij wordt rekening gehouden met signaaldegradatie in de kabel door ruis, spanningsval en flankafronding ten gevolge van de impedantie-eigenschappen van de kabel.

In oudere, goedkope USB/serieel-adapters en sommige laptops is de zwaai van het signaal aan de zenderzijde nauwelijks groter dan ± 5 V of zelfs ± 3 V als gevolg vereenvoudigingen in het ontwerp van de RS-232-driver om de ontwikkelings- en componentkosten te minimaliseren. Dit heeft geleid tot de populaire overtuiging dat laptops en RS-232-apparatuur vaak niet goed samenwerken. De afbeelding van **figuur 1** (uit een TI-document) illustreert de standaard-conforme spanningsniveaus.

Of het eerder genoemde communicatieprobleem met het Keithley-instrument te wijten was aan het gebruik van een USB-verlengkabel, een aardlus of de elektrische eigenschappen van de ingebouwde transceiver is hier niet relevant. Een USB-kabel van de ene kant van het lab naar de andere is immers niet alleen slordig maar ook een struikelgevaar. Wat we nodig hebben is een nette, compacte en draadloze oplossing!

Ik zocht overal naar een kant-en-klare oplossing voor mijn probleem, maar ik vond slechts vrij grote modules voor DIN-rail-montage of DHZ-ontwerpen die waren ontworpen om



één specifiek probleem op te lossen en voor mij te weinig mogelijkheden hadden. Zinloos – als ik een nette, universele RS-232 radiocommunicatie-eenheid wilde, dan moest ik mezelf achter de oren krabben en zelf zoiets ontwerpen.

Ontwerpcriteria

Dit ontwerp is bewust gericht op de maker-lab omgeving en wordt zeker geen fraai ingekast model zoals je in een elektronicawinkel kunt kopen. Toch is het een goede gewoonte om het ontwerp van het apparaat zorgvuldig te overwegen, zodat het alle taken kan uitvoeren waarvoor het bedoeld is.

Een ESP32 doet al het werk

De RS2-32-gateway is primair bedoeld om gegevens van en naar het netwerk te zenden. Dit wordt gewoonlijk gedaan via TCP-poort 23 (Telnet). Applicatie-specifieke software-aanpassingen zouden echter ook mogelijk moeten zijn, die de gebruiker kan kiezen aan de hand van de eisen van de betreffende applicatie. Voor mij was een SCPI-naar-MQTT-service belangrijk, zodat de SourceMeter-testapparatuur zijn metingen 's nachts naar Grafana (zie verderop) kan sturen voor evaluatie.

De ESP32 biedt voldoende rekenkracht (naast het verzenden van eenvoudige TCP-packets) zodat het aangesloten apparaat eenvoudig kan worden geïntegreerd in Home Assistant of een vergelijkbare service via MQTT of om het te delen via een speciaal frontend via de ESP32-webserver.

Een alternatief is de ESP8266, die iets minder ruimte inneemt en iets goedkoper is. De bespaarde ruimte gaat echter ten koste van een aanzienlijk zwaarder belaste CPU en minder RAM-geheugen. Bovendien mist de ESP8266 een (in deze toepassing voordelige) Bluetooth-module naast de gewenste WiFi-functie.

Het nadeel van de ESP32 is natuurlijk zijn aanzienlijk hogere stroomverbruik (maximaal wordt ongeveer 500 mA van de 3,3V-rail genomen). Natuurlijk zal de CPU niet continu bezig zijn met rekenen, maar afhankelijk van hoe goed de software uiteindelijk is geoptimaliseerd, zal de piekstroom waarschijnlijk deze waarde bereiken, zoals aangegeven in tabel 4.2 van de datasheet [1]. Een vergelijking met de datasheet van de ESP8266 [2] laat zien dat het stroomverbruik van de ESP32 ongeveer 35% hoger is tijdens zenden en ongeveer 80% hoger tijdens ontvangen.

Een net, compact ontwerp

De meeste mensen zijn het er waarschijnlijk wel mee eens dat er al genoeg kabels rondslingeren op en onder werkbanken. Deze print kan rechtstreeks op een D-sub DE-9 connector worden gesoldeerd en is qua afmetingen niet veel groter dan een standaard IEC C13 netstekker (inclusief de buigstraal). Dit betekent dat de print in aangesloten toestand de footprint van het RS-232-apparaat niet veel groter maakt. Als er een behuizing nodig is, kan deze eenvoudig 3D-geprint worden; in het ideale geval past de schakelingt zelfs in een standaardbehuizing.

Voeding via micro-USB

De print wordt gevoed via een micro-USB-connector (inmiddels de standaard voor veel kleine apparaten). De +3,3 V die de ESP32 nodig heeft, wordt door de Advanced Monolithic Systems [3] AMS1117 lineaire spanningsregelaar uit de +5V-voeding van de USB-poort afgeleid. Hoewel de AMS1117 geschikt is voor maximaal 1 A, wordt hij behoorlijk heet op deze relatief kleine print. Een efficiënter alternatief is de Texas Instruments [4] TPS62291 schakelende step-down-converter, die wordt geleverd in een WSON-6 behuizing van 2x2 mm. Deze levert ook maximaal 1 A rn zou wat ruimte besparen. Tenzij je een microscoop hebt en alle benodigde gereedschappen en ervaring om op dit niveau te werken, is deze chip lastiger te monteren vanwege de kleine afmetingen van de behuizing. Wie aan zijn soldeervaardigheden twijfelt, kan nuttige aanwijzingen en tips vinden in de video [5]. Er moet ook worden opgemerkt dat de verkrijgbaarheid van dit specifieke onderdeel de afgelopen jaren aan schommelingen onderhevig was. Hierdoor is de prijs bij sommige distributeurs toegenomen tot meer dan € 20 in plaats van de meer gebruikelijke € 1,70. Als je de prijzen op internet vergelijkt, kun je exemplaren vinden voor prijzen die doen denken aan de goede oude tijd. Om het gedoe te verminderen, heb ik in deze versie alleen de AMS1117 gebruikt, wat het solderen met de hand veel eenvoudiger maakt.

Voeding via de 9-polige sub-D connector

Als je er niet vies van bent om af en toe de gebaande paden te verlaten, kan de print ook gevoed worden via de 9-polige sub-D connector. Met een kleine aanpassing aan de eindapparaten kan de vereiste 5 V worden geleverd via de RS-232-connector (waardoor een extra voeding wordt vermeden). De standaard ondersteunt een dergelijke voedingsoptie via de connector echter niet, dus we moeten er rekening mee houden dat deze aanpassing ook niet conform de standaard is.

Volgens V.28 kunnen alle signaallijnen spanningen voeren van +15 V tot -15 V; voor voedingsdoeleinden is het zinvol om een van de pinfuncties te 'misbruiken' die zelden wordt benut. Pin 9 is de zogenaamde Ring Indicator (RI) en wordt alleen gebruikt door een modem om een inkomend gesprek te signaleren. RS-232-modems zijn tegenwoordig zo goed als redundant, waardoor de RI-pin een geschikte kandidaat is om voor de voeding te gebruiken.

Eerst moeten we echter controleren wat er gebeurt als we onverwacht de maximale spanning aanleggen, bijvoorbeeld als wer een eindapparaat aansluiten dat onze voedingslijn aanstuurt conform V.28.





Figuur 2. Het eenvoudige schema van de ESP32/RS-232-dongle.

De +15 V is geen probleem voor de AMS1117 zelf; deze spanning valt binnen de gespecificeerde limieten. Als er echter -15 V zou aanliggen, zouden de interne beveiligingsdiodes van de AMS regelaar de gewoonlijk 10...20 mA van het RS-232 signaal naar massa afleiden. Empirisch is gebleken dat deze diodes hier geen problemen mee hebben, en volgens hun specificatie zijn de RS-232-drivers kortsluitvast, dus ook hier treedt geen schade op.

De VDD-voedingsspanning van onze RS-232-transceiver is slechts gespecificeerd tot 5,5 V en is ook verbonden met de +5 V. Gelukkig daalt de spanning in dit geval tot ongeveer 3,5 V vanwege de stroombegrenzing door de RS-232-drivers. Toegegeven, we gaan hier een beetje kort door de bocht, maar we hebben al deze factoren alleen overwogen voor het zeldzame geval dat RI op pin 9 daadwerkelijk wordt aangestuurd door het eindapparaat. Meestal is deze pin niet aangesloten.

Echte RS-232

Zoals eerder vermeld is ons doel hier om zo dicht mogelijk bij de V.28 standaard te blijven, daarom is een data-transceiver absoluut noodzakelijk. De ST232EBTR van ST [6] is hiervoor geschikt; deze chip is verkrijgbaar in een TSSOP-16-behuizing. Met een prijs van ongeveer € 1 is dit IC heel betaalbaar; het beschikt over een interne spanningsverdubbelaar en inverterschakelingen, genereert voedingsspanningen van plus en min 10 V, die worden gebruikt voor de RS-232-signalen (die typisch ongeveer 9 V bedragen onder normale bedrijfsomstandigheden). Het schema van de ESP32/RS-232-dongle met de ESP32-WROOM-module (of de WROVER met PSRAM) en de twee IC's (ST232EBTR en AMS1117) plus alle periferie is te zien in **figuur 2**.

Plaatsing van onderdelen

Voor zo'n kleine print met zo weinig componenten heeft het weinig zin om die professioneel te laten bestukken of de extra kosten te maken van een soldeerpasta-stencil en aansluitende assemblage met behulp van een reflow-plaat. De print van figuur 3 kan op de ouderwetse manier met de hand worden gesoldeerd omdat er weinig componenten zijn en de soldeereilandjes goed toegankelijk zijn. De eerste component die moet worden gemonteerd is de ST232 (U3). Deze is relatief plat en er is wellicht enige nabewerking nodig om zuivere aansluitingen te krijgen. Vervolgens kunnen de condensatoren, weerstanden en LED op hun plaats worden gesoldeerd, gevolad door de AMS1117 (U1) en de USB-connector. Dan kan de ESP32 en als laatste de 9-polige sub-D-connector worden gesoldeerd.

De firmware

De allereerste functie van de firmware is het doorsluizen van karakters die via het netwerk worden ontvangen naar de RS232-poort en vice versa, maar dankzij de veelzijdigheid van de ESP32 is hier veel meer mogelijk. De firmware, zoals te vinden in de repository [7], biedt momenteel twee bedrijfsmodi – Telnet en MQTT/SCPI – die aansluitend worden besproken.

Telnet/Raw TCP-Socket

Zoals we al hebben aangestipt bij de ontwerpcriteria, is de primaire functie het overbrengen van gegevens van het netwerk naar de RS-232poort. Voor zulke adapters is het gebruikelijk om de karakters zonder bewerking te verzenden via TCP-poort 23. Deze poort werd vroeger gebruikt voor Telnet-verbindingen naar een server, maar wordt tegenwoordig zelden gebruikt (behalve bij embedded systemen). Voor ons doel, het verzenden van commando's of meetwaarden van en naar seriële apparaten, is dit in een thuisnetwerk meer dan afdoende. Telnet is een netwerk-communicatieprotocol dat is ontwikkeld om de afstandsbediening van een computer via een netwerk mogelijk te maken. Het biedt een op tekst gebaseerd communicatiekanaal waarmee een gebruiker toegang kan krijgen tot een computer op afstand en daar opdrachten kan uitvoeren. De verbinding is bidirectioneel, zodat zowel de invoer als het antwoord via het netwerk



Figuur 3. Deze kleine print met goed toegankelijke pads is ideaal om SMDcomponenten met de hand te solderen.


Figuur 4. Een voorbeeld van een Grafana-dashboard, dat golfvormen toont die zijn opgebouwd uit gemeten waarden in de loop van de tijd (niet gerelateerd aan dit project).

worden doorgegeven. Telnet werd oorspronkelijk ontwikkeld in het nog primitieve tijdperk van de vroege internetstandaarden en wordt vaak gebruikt in toepassingen waar eenvoudige tekstgebaseerde communicatie volstaat. Het wordt echter als onveilig beschouwd omdat het geen encryptie biedt, waardoor mogelijk alle overgedragen gegevens kunnen worden onderschept, inclusief wachtwoorden en andere gevoelige informatie.

Voor de eerste communicatietests kun je het hulpprogramma *telnet.exe* gebruiken, dat standaard in eerdere versies van Windows zat. Als het niet beschikbaar is, kun je het gratis en meer universele alternatief *PuTTY* [8] gebruiken.

MQTT/SCPI

Voor toepassingen die regelmatig meetwaarden moeten vastleggen over perioden van meerdere uren, kun je geschikte Pythonscripts ontwikkelen of zelfs kant-en-klare tools op je PC starten en ze uitlezen via de Telnet-poort. Een praktisch alternatief voor degenen die al een geschikte infrastructuur in hun thuisnetwerk hebben opgezet, is om de energieverslindende PC helemaal te omzeilen en een eenvoudiger meetopstelling te gebruiken.

In veel huishoudens worden al services zoals MQTT, InfluxDB en Grafana gebruikt voor het vastleggen van meetwaarden van sensoren op een Raspberry Pi of via Docker Containers met behulp van hun NAS. Zonder al te veel in detail te treden en in eenvoudige bewoordingen:

> MQTT is een lichtgewicht berichten-

protocol dat is ontworpen voor efficiënte communicatie tussen IoT-apparaten en netwerken met geringe bandbreedte en grote latentie. Het gebruikt een informatie-broker met een publish/ subscribe-model om naadloze gegevensuitwisseling met minimale overhead mogelijk te maken.

- InfluxDB is een tijdreeks-database, ontworpen om efficiënt om te gaan met hoge schrijf-/vraag-belastingen voor gegevens met tijdstempels. Tags zijn key/value-paren voor indexering terwijl velden de eigenlijke data bevatten.
- > Grafana haalt gegevenswaarden uit een database en geeft de informatie op een gebruiksvriendelijke manier weer in de vorm van door de gebruiker gedefinieerde grafieken.

Voor degenen die thuis zoiets hebben ingericht, biedt deze firmware de optie om meetwaarden gegenereerd door het SCPI-compatibele apparaat direct te publiceren naar een MQTT-broker voor verdere verspreiding.

De set commando's die door diverse SCPI-apparaten worden gebruikt, varieert aanzienlijk. Het enige apparaat dat momenteel door de code wordt ondersteund is de Keithley SourceMeter 2400. Dit is echter als een goed uitgangspunt voor iedereen om eigen firmware-varianten toe te voegen. Bekwame embedded ontwikkelaars worden uitgenodigd om de software-functionaliteit uit te breiden en te delen met de gemeenschap.

Om te zien hoe meetweergaven kunnen worden weergegeven in verschillende toepassingen, zie **figuur 4** (Grafana), **figuur 5** (MQTT



Figuur 5. MQTT Explorer toont metingen die gepubliceerd door de ESP32/RS-232-adapter via MQTT zijn verzonden.

Figuur 6. Accuspanning (groen) en stroom (geel) tijdens het laden van een 12V-lood/zuur-accu, gemeten via een ESP32/RS232-adapter.

Explorer, een uitgebreide MQTT-client voor verschillende platforms [10]) en **figuur 6** (ook Grafana).

Om dit mogelijk te maken moet de ESP32 een paar dingen regelen en daarvoor heeft hij informatie nodig die we via een webinterface moeten verstrekken.

WiFi-setup

Als geen geconfigureerd access point (AP) wordt gevonden of als er geen is geconfigureerd, dan wordt de ESP32 actief en biedt zichzelf aan als een AP met de naam *esp232-config*. Het is het beste om verbinding te maken met dit AP met behulp van een smartphone. Na het inloggen kun je de belangrijkste parameters configureren door in een browser naar *http://192.168.4.1/* te gaan. De webpagina die op de ESP32 wordt gehost is te zien in **figuur 7**. In **tabel 1** kun je nalezen wat in elke rij moet worden ingevoerd.

OTA: Over-the-Air updates

In de Arduino IDE of in PlatformIO, waar ik de voorkeur aan geef, kan de ESP32 'op afstand' worden bijgewerkt. Het onderliggende protocol wordt geïmplementeerd door de ArduinoOTA-component. Hoewel deze functie frequente updates van de microcontroller vergemakkelijkt, kwamen er in de praktijk belangrijke nadelen aan het licht. Het lijkt erop dat het voortdurend toewijzen en vrijgeven van buffers voor elk ontvangen UDP-pakket het geheugen van de ESP32 fragmenteert, wat kan leiden tot onvoorspelbare resets.

	ESP232 - ESP232
	v1.13 - b376f34
	[Enable OTA]
Hostname:	ESP232
WiFi SSID:	g3gg0.de
WiFi Password:	
WiFi networks:	Scan WiFi
MQTT Server:	
MQTT Port:	
MQTT Username:	
MQTT Password:	
MQTT Client Identification:	ESP232
Baudrate:	57600
Data bits (5-8):	8
Parity (0=none, 1=even, 2=odd):	0
Stop bits (0=1, 1=1.5, 2=2):	0
Connect Message:	
Disconnect Message:	
Verbosity:	Serial UDP _
Publish rate [ms]:	500
Publish data via MQTT:	Publish string Publish parsed Exit REM Publish MEAS
Update URL (<u>Release</u>):	
	Save Save & Reboot



Tabel 1. Gegevens voor ESP32 WiFi-setup.

Hostnaam	De naam die het apparaat gebruikt om zichzelf te identificeren via MDNS in het netwerk.
WiFi SSID/Password	Het WiFi-netwerk waarmee het apparaat verbinding moet maken bij het opstarten.
WiFi Networks	Toont een lijst met gevonden WiFi-netwerken; als je op een netwerk klikt, wordt de naam weergegeven.
MQTT []	Configuratie-instellingen voor de MQTT-client, in combinatie met SCPI hieronder.
Baudrate / Data bits / Stop bits	Corresponderende parameters voor RS232-communicatie.
Dis-/Connect Message	[In speciale gevallen] Bericht dat moet worden verzonden via RS232 voor TCP-verbinding.
Verbosity	[Voor experts] Debuggen van communicatie via UDP.
Publish data via MQTT	[Voor experts] Voor de Keithley Source Meter 2400: displaydata parsen of een meting starten en de resultaten publiceren via MQTT.
Update URL	Voer voor firmware-updates gewoon de HTTP URL's in en het verwachte .bin-bestand wordt gedownload en geflashed.

Tabel 2: Pinning van de programmeer-interface.

Pin	Betekenis	Niveau
100	Stelt de boot-modus in, heeft pas effect na een RESET.	Flash: GND (normaal: open, 3,3 V)
RESET	De resetpin van de ESP32 (CHIP_PU)	actief: GND, niet actief: open (3,3 V)
TXD0	Zendsignaal van de ESP32	0 V / 3,3 V
RXD0 (U3_34)	Ontvangstsignaal van de ESP32	0 V / 3,3 V
+5V	Voedingsspanning	5 V
GND	Massa	GND

Tot overmaat van ramp gebeurt dit ongeacht of er OTA-packets onderweg zijn. Je kunt je voorstellen dat een crashbericht dat wordt uitgevoerd via de UART van de ESP32 zelden goed wordt ontvangen door het aangesloten apparaat. Gelukkig zijn er verbeteringen [11] die het crashpercentage aanzienlijk verlagen. Vanwege deze mate van onzekerheid wordt de OTA-functie alleen kortstondig en op verzoek van de gebruiker geactiveerd. Daarom is er een veld (Enable OTA) voor in de webinterface.

Flashen

Als je nog geen Pogo-adapter voor de ESP32-modules hebt, moet de microcontroller

na assemblage worden voorzien van de juiste firmware met behulp van de TX- en RX-pinnen van de ESP32. Download hiervoor de broncode van [7], importeer deze in PlatformIO, compileer en flash deze naar het board.

Rechtsonder op de print is voorzien in een programmeerpoort. Voor sporadisch gebruik kan je gewoon het IOO-signaal aan massa leggen en de TXD0 en RXD0 (pin U4_34 van de microcontroller, zie figuur 8) aan een USB/ UART-adapter solderen.

Zoals tabel 2 toont, heeft de ESP32 een paar 'strapping pins' die moeten worden ingesteld als je de ROM-code met deze methode laadt.



Figuur 8. Aansluitingen voor de programmer op de print.

In deze opstelling hoeven we ons maar om één ervan zorgen te maken (IO0). Zodra de firmware is geladen, zou na ongeveer 30 seconden een LED moeten gaan knipperen.

Niet alles is even mooi

De ESP32 heeft zijn capaciteiten bewezen in mijn projecten. Er waren echter aanvankelijk verschillende problemen in de Arduino-bibliotheek met betrekking tot het gebruik van de UART-buffer, gerelateerd aan de code die de bibliotheken in het IDF-framework rechstreeks benadert. Bugs in de ArduinoOTA-bibliotheek zorgden er ook voor dat ik een aantal nachten bezig was om de problemen te lokaliseren. Als gevolg hiervan zijn er bepaalde workarounds in de broncode. Maar omdat deze firmware open source is, zal hij blijven evolueren en profiteren van verbeteringen die door de community worden bijgedragen.

De volgende generatie

Wat me tot nu toe stoorde aan v1.0 van het project is het ietwat omslachtige flash-proces tijdens de eerste installatie (of flashen om fouten in de code te corrigeren). Dankzij een flash-adapter met Pogo-pinnen die ik voor al mijn projecten gebruik, gaat dit veel makkelijker, maar niet iedereen heeft een geschikte adapter bij de hand of werkt graag met jumperkabels.

Daarom gaat de ESP32/RS-232-adapter naar zijn volgende incarnatie: die wordt momenteel ontwikkeld als versie 2.0 met behulp van een ESP32-S3-PICO. Er zijn al enkele eerste prototypes gebouwd en die zien er veelbelovend uit. De PICO is een complete module





met SPI-flash en RAM, alles geïntegreerd in een LGA-56 behuizing. Hoewel de PICO er op het eerste gezicht uitziet als een QFN, die meestal gemakkelijk met de hand te solderen is, moet GND verbonden worden met een pad aan de onderzijde, die bij handmatige assemblage via een gat gesoldeerd moet worden. Deze behuizing heeft ook geen aansluitpinnen, zodat de aansluitingen slechts moeizaam vanaf de rand gesoldeerd worden. Het lijkt erop dat een (mini)reflow-plaat hier de beste methode is [12].

Het grote voordeel van de S3 is dat deze de complete USB-periferie aan boord heeft om te flashen of te debuggen. Indien nodig kan ESP232 v2.0 ook worden gebruikt als een USB/-RS-232-adapter. Of we daarmee uitkomen waar we begonnen zijn is een retorische vraag. Als je bedenkt dat de ESP8266 al van meet af aan ideaal was om een WiFi/serieel-adapter te bouwen, vraag je je af waarom iets dergelijks vandaag de dag nog steeds niet op de markt is.

220352-03

Vragen of opmerkingen?

Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur via elektor@g3gg0.de of naar de redactie van Elektor via redactie@elektor.com.



Over de auteur

Georg Hofstetters interesse in elektronica werd gewekt

in de dagen van de Commodore 64 en gaatjesprint. Na zijn opleiding tot elektronicus behaalde hij een graad in computerwetenschappen. Sindsdien ontwikkelt of reverse-engineert hij software en embedded systemen en publiceert hij geselecteerde projecten op zijn website www.g3gg0.de. Enkele van de bekendere projecten zijn firmware-aanpassingen voor Canon DSLR's met de naam Magic Lantern (*www.magiclantern.fm*) en voor de Toniebox (*https://gt-blog.de/ toniebox-hacking-how-to-get-started*).



WEBLINKS

- [1] ESP32-datasheet: https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf
- [2] ESP8266-datasheet : https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf
- [3] AMS1117: http://www.advanced-monolithic.com/pdf/ds1117.pdf
- [4] TPS62291: https://www.ti.com/product/de-de/TPS62291
- [5] Handmatig SMD's solderen (Elektor-video): https://www.elektormagazine.de/news/uberzeugendes-video-loten-von-smd-bauteilen
- [6] ST232: https://www.st.com/resource/en/datasheet/st202eb.pdf
- [7] Project-repository: https://github.com/g3gg0/ESP232
- [8] PuTTY: https://www.putty.org/
- [9] MQTT Explorer: https://mqtt-explorer.com/
- [10] WiFiUDP-crashprobleem: https://github.com/espressif/arduino-esp32/issues/4104
- [11] Texas Instruments Application Report: RS-232 Frequently Asked Questions: https://www.ti.com/lit/an/slla544/slla544.pdf
- [12] M. Divito, "Mini-reflowplaat", Elektor november/december 2023: http://www.elektormagazine.nl/230456-03

Alle begin.

...gaat verder met de opamp

Eric Bogers (Elektor)

In de vorige aflevering hebben we een blik geworpen op discrete FET's, en hebben we (eindelijk!) een begin gemaakt met onze bespreking van wat wellicht het belangrijkste (analoge) elektronica-onderdeel is: de opamp. Daarover valt een heleboel te vertellen, zoals u zult merken, en we kunnen (en gaan!) er ook een heleboel aan rekenen

De operationele versterker als 'black box'

Wanneer we in de praktijk een opamp willen gebruiken, zijn we (als beginnende elektronica-hobbyisten) nauwelijks geïnteresseerd hoe dat ding er 'van binnen' uit ziet; we hoeven slechts enkele (belangrijke) parameters te kennen om aan de slag te gaan. We beschouwen de opamp dan als 'zwarte doos' met in- en uitgangen (**figuur 1**). In de eerste plaats zijn we geïnteresseerd in de open-lusversterking,



Figuur 1. De opamp als black box.



die doorgaans tussen 10.000 en 100.000 ligt. Met behulp van deze parameter kan de uitgangsspanning worden berekend (waarbij U_{ni} de spanning op de niet-inverterende ingang is en U_i de spanning op de inverterende ingang):

$U_{\text{out}} = V_0 \cdot (U_{\text{ni}} - U_{\text{i}})$

Hieruit zouden we de indruk kunnen krijgen dat de exacte waarde van de open-lus-versterking (dat is de versterking zonder terugkoppeling) van doorslaggevend belang is. Dat is echter niet het geval, zoals we verderop nog zullen merken.

Bij opamps die met bipolaire transistoren zijn opgebouwd, ligt de ingangsimpedantie in de orde van grootte van 1 M Ω . Als voor de ingangsschakeling van de opamp FET's (veldeffect-transistoren) zijn gebruikt, ligt die ingangsimpedantie nog enkele orden van grootte hoger.

De maximale voedingsspanning ligt meestal tussen ±16 V en ±22 V; bij hoogspannings-opamps kan die waarde echter oplopen tot ±150 V. De maximale uitgangsstroom ligt doorgaans in de orde van grootte van 20 mA; vermogens-opamps halen (indien afdoende gekoeld) een paar ampère. Over de grensfrequentie komen we nog uitgebreid te spreken.

Versterkerschakelingen

De twee gebruikelijke versterkerschakelingen met opamps zijn de inverterrende en de niet-inverterende versterker. De eerste keert zoals de naam al suggereert - de polariteit van het ingangssignaal om, terwijl de tweede dat juist niet doet.



De inverterende versterker

Figuur 2 toont het principeschema van de inverterende versterker. Deze heeft, zoals u ziet, eigenlijk slechts twee externe onderdelen nodig: twee weerstanden. Voor audiotoepassingen komen daar doorgaans nog een paar condensatoren bij.

Hoeveel versterking levert deze schakeling? Omdat de niet-inverterende ingang aan massa is gelegd, geldt om te beginnen voor de open-lusversterking:

$$V_0 = -\frac{U_{out}}{U_V} \implies U_V = -\frac{U_{out}}{V_0}$$

Voor de ingangsspanning geldt:

$$U_{\rm in} = U_{\rm R1} + U_{\rm V} = I_{\rm R1} \cdot R_1 - \frac{U_{\rm out}}{V_0}$$

De versterking is gedefinieerd als 'uitgangsspanning gedeeld door ingangsspanning'. Als we daarin het hierboven verkregen resultaat invullen, vinden we:

$$V = \frac{U_{\text{out}}}{U_{\text{in}}} = \frac{U_{\text{out}}}{I_{\text{R1}} \cdot R_1 - \frac{U_{\text{out}}}{V_0}}$$

Voor de stromen in en uit het knooppunt voor de inverterende ingang geldt:

$$I_{\text{R1}} = I_{\text{R2}} + I_{\text{Ri}} = I_{\text{R2}} + \frac{U_{\text{V}}}{R_{\text{i}}} = I_{\text{R2}} - \frac{U_{\text{out}}}{R_{\text{i}} \cdot V_{0}}$$

Invullen in de vergelijking voor de versterking levert:

$$V = \frac{U_{\text{out}}}{U_{\text{in}}} = \frac{U_{\text{out}}}{I_{\text{R2}} \cdot R_1 - \frac{U_{\text{out}} \cdot R_1}{R_1 \cdot V_0} - \frac{U_{\text{out}}}{V_0}}$$

Voor de maas met weerstand R2 geldt:

$$U_{\text{out}} = U_{\text{V}} - U_{\text{R2}} = -\frac{U_{\text{out}}}{V_0} - I_{\text{R2}} \cdot R_2$$

Hieruit kunnen we I_{R2} oplossen:

$$I_{\rm R2} = \frac{U_{\rm out}}{V_0 \cdot R_2} - \frac{U_{\rm out}}{R_2}$$

Wanneer we dit ook in de vergelijking voor de versterking invullen en daarna overal U_{out} wegstrepen, komen we (eindelijk) uit bij de exacte vergelijking voor de versterking van de inverterende versterker:

$$V = \frac{1}{-\frac{R_1}{V_0 \cdot R_2} - \frac{R_1}{R_2} - \frac{R_1}{R_1 \cdot V_0} - \frac{1}{V_0}}$$



Figuur 2. De inverterende versterker.

Nu wordt het leuk. We redeneren als volgt: doorgaans is de open-lusversterking veel groter dan de verhouding R2/R1, terwijl ook de inwendige weerstand van de opamp veel groter is dan de externe weerstanden. Met dat in gedachten kunnen we de vergelijking voor de versterking in goede benadering vereenvoudigen tot:

$$V \approx -\frac{R_2}{R_1}$$

Dat is een lekker handzame formule om de versterking van een inverterende opamp-versterker te berekenen! Laten we aan de hand van een getallenvoorbeeld eens kijken hoe groot de afwijking in werkelijkheid is. Voor een versterkerschakeling met $R_2 = 100 \text{ k}\Omega$ en $R_1 = 10 \text{ k}\Omega$ komen we met de vereenvoudigde formule uit op een versterking van –10. Volgens de eerder afgeleide exacte vergelijking (bij een aangenomen open-lusversterking van 100.000 maal en een ingangsweerstand van 1 M Ω) komen we uit op een versterking van –9,99889... maal (we laten het aan de geïnteresseerde lezer over dit zelf te berekenen). De afwijking is dus enkele orden van grootte kleiner dan de tolerantie van precisieweerstanden, zodat we die met een gerust hart onder het tapijt mogen vegen. (Helemaal afgezien van het feit dat we enkele andere factoren buiten beschouwing hebben gelaten, zoals uitgangsweerstand, temperatuurdrift, ruis...)

De niet-inverterende versterker

In **figuur 3** ziet u het principeschema van de niet-inverterende versterker. In de minimale uitvoering hebben we ook hier slechts twee externe weerstanden nodig.

Voor de uitgangsspanning van een opamp geldt:

$$U_{\text{out}} = (U_{\text{in}} - U_{\text{V}}) \cdot V_0 \implies (U_{\text{in}} - U_{\text{V}}) = \frac{U_{\text{out}}}{V_0}$$

Voor de versterking van de schakeling als geheel kunnen we schrijven:

$$V = \frac{U_{\text{out}}}{U_{\text{in}}} = \frac{U_{\text{out}}}{(U_{\text{in}} - U_{\text{V}}) + U_{\text{V}}}$$



Figuur 3. De niet-inverterende versterker.



In de loop van de afleiding zal duidelijk worden waarom we hierboven de ogenschijnlijk zinloze term U_V hebben geïntroduceerd. Omkeren van deze formule en invullen van de vergelijking voor de uitgangsspanning van de opamp levert (we laten een paar tussenstappen weg om dit verhaal niet 'droger' te maken dan het al is...):

$$\frac{1}{V} = \frac{U_{\text{out}}}{V_0 \cdot U_{\text{out}}} + \frac{U_V}{U_{\text{out}}} = \frac{1}{V_0} + \frac{U_V}{U_{\text{out}}}$$

 R_1 en R_2 vormen een spanningsdeler die door $R_{\rm i}$ wordt belast. Dat betekent voor $U_{\rm V}\!\!:$

$$U_{V} = \frac{U_{out} \cdot (R_2 \| R_1)}{R_1 + (R_2 \| R_1)} = \frac{U_{out}}{R_1 \cdot \left(\frac{1}{R_2} + \frac{1}{R_1}\right) + 1}$$

Wanneer we dit in de (omgekeerde) vergelijking voor de versterking invullen, krijgen we:

$$\frac{1}{V} = \frac{1}{V_0} + \frac{1}{R_1 \cdot \left(\frac{1}{R_2} + \frac{1}{R_i}\right) + 1}$$

Na omkering vinden we als exacte vergelijking voor de versterking:

$$V = \frac{1}{\frac{1}{V_0} + \frac{1}{R_1 \cdot \left(\frac{1}{R_2} + \frac{1}{R_1}\right) + 1}}$$

Ook in dit geval bedenken we dat de open-lusversterking veel groter is dan de verhouding R_2/R_1 en dat bovendien R_i zeer veel groter is dan R_2 . Wanneer we die overwegingen op de bovenstaande vergelijking loslaten, vinden we uiteindelijk voor de versterking van de niet-inverterende opamp-versterker:

$$V \approx \frac{R_1}{R_2} + 1$$

Ook dit is weer een lekker compacte formule, waarvoor we in veel gevallen niet eens een rekenmachine nodig zullen hebben. In de volgende aflevering duiden we nog wat dieper in de opamptheorie; daarna komen eindelijk 'echte' schakelingen aan bod! 230756-03

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.

Gerelateerd product

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





developer's zone

tips & trucs, vakkunstigheden en andere nuttige informatie



Aanbevolen ESPbibliotheken

Saad Imtiaz en Jean-François Simon (Elektor)

Op zoek naar ESP-gerelateerde bibliotheken? Bekijk deze aanbevelingen van het engineeringteam van Elektor.

WLED (@Aircoookie - GitHub)

WLED van Christian Schwinne is een snelle en veelzijdige implementatie van een ESP8266/ESP32-webserver om NeoPixel-LED's (WS2812B, WS2811, SK6812) aan te sturen. Het ondersteunt ook SPI-gebaseerde chipsets zoals de WS2801, APA102 en nog veel meer. De bibliotheek bevat meer dan 100 speciale effecten voor NeoPixels, 50 paletten, FastLED-ruiseffecten en meer! Het heeft een moderne UI met geavanceerde bedieningselementen. Configuratie gebeurt via het netwerk. Een van de opvallende voordelen is dat het tot tien LED-uitgangen per instantie ondersteunt en kan werken met RGBW-strips. Je kunt tot 250 gebruikerspresets gebruiken om eenvoudig kleuren/effecten op te slaan en te laden, en deze doorlopen. Presets kunnen ook worden gebruikt om automatisch API-calls uit te voeren. Er is ook een Nightlight-functie, die de LED's geleidelijk dimt. Wat updates betreft, wordt Over the Air (HTTP + ArduinoOTA) ondersteund; dit kan met een wachtwoord worden beveiligd. Als je je RGB-lampen wilt bedienen of je kamer een nieuw thema wilt geven, ga hier dan voor! Zie ook Adafruit_NeoPixel verderop. https://github.com/Aircoookie/WLED



ExpressLRS (@ExpressLRS - GitHub)

ExpressLRS is een open-source radio-link voor RC-toepassingen (radiobesturing). Het levert uitstekende prestaties met behulp van de SX127x/SX1280 LoRa-chip in combinatie met een Espressif- of STM32-microcontroller. Met ExpressLRS kunnen gebruikers profiteren van een goed bereik en een zeer lage latency, dankzij de LoRa-modulatie en de gereduceerde pakketgrootte. ExpressLRS ondersteunt hardware van vele fabrikanten: AxisFlying, BETAFPV, Flywoo, FrSky, HappyModel, HiYounger, HGLRC, ImmersionRC, iFlight, JHEMCU, Jumper, Matek, NamimnoRC, QuadKopters en SIYI. Het beschikt over 1kHz-pakketsnelheid, telemetrie, WiFi-updates, twee frequenties voor de RC-link (2,4 GHz of 900 MHz) en meer. Het is ongetwijfeld zeer interessant voor veel RC-projecten, met verschillende hardware geschikt voor verschillende eisen.

https://github.com/ExpressLRS/ExpressLRS

ESPHome (@esphome - GitHub)

ESPHome is een open-source framework dat het configureren en beheren van apparaten gebaseerd op ESP8266 en ESP32 vereenvoudigt. Het stelt gebruikers in staat om aangepaste firmware te maken voor deze apparaten zonder uitgebreid te moeten programmeren. Met ESPHome kun je de functionaliteit en instellingen van apparaten definiëren met behulp van een gebruiksvriendelijk YAML-configuratiebestand, waardoor het toegankelijk is voor zowel beginners als voor ervaren ontwikkelaars. De belangrijkste functies van ESPHome zijn ondersteuning voor een breed scala aan sensoren en componenten, automatisch vinden en integratie met populaire domotica-platforms zoals Home Assistant, en over-the-air updates voor naadloze firmware-upgrades. Het bevordert de ontwikkeling van DHZ smart home-oplossingen, waardoor gebruikers hun apparaten kunnen aanpassen aan specifieke behoeften, zoals temperatuurbewaking, verlichtingsregeling of huisbeveiliging. ESPHome heeft aan populariteit gewonnen in de smart home-community. Werp er beslist een blik op als je het nog niet kent!

https://github.com/esphome/esphome



ESPAsyncWebServer (@ me-no-dev - GitHub)

ESPAsyncWebServer is een asynchrone HTTP- en WebSocket-server voor ESP8266 die wordt gebruikt in de Arduino-omgeving.

Het kan gebruikt worden met PlatformIO en Arduino IDE. Het gebruik van een asynchroon netwerk betekent dat je meer dan één verbinding tegelijkertijd kunt afhandelen. Je wordt gebeld zodra het request klaar is en ontleed. Als je het antwoord verstuurt, ben je meteen klaar om andere verbindingen af te handelen, terwijl de server op de achtergrond het antwoord verstuurt. De verwerkingssnelheid is erg hoog! ESPAsyncWebserver biedt een gebruiksvriendelijke API en is compatibel met HTTP Basic en Digest MD5 Authenticatie (standaard), evenals met Chunked Response. Er zijn diverse plugins die hun eigen voordelen bieden, zoals verschillende locaties, events naar de browser sturen, geavanceerde URL-herschrijving en meer. *https://github.com/me-no-dev/ESPAsyncWebServer*

WifiManager (@tzapu - GitHub)

WiFiManager is een verbindingsmanager voor ESP-gebaseerde projecten. Het geeft de gebruiker de mogelijkheid om in runtime WiFi-referenties en aangepaste parameters te configureren met behulp van een captive portal. Hoe dat werkt? Wanneer je ESP opstart, wordt hij geconfigureerd in station-modus en probeert hij verbinding te maken met een eerder opgeslagen Access Point. Als dit niet lukt, zet de firmware de ESP in Access Point-modus en start een DNS- en webserver op. Vervolgens kun je met elk WiFi-apparaat met een browser (computer, telefoon, tablet) verbinding maken met het nieuwe Access Point

en de referenties instellen. De ESP maakt dan verbinding met het netwerk van je keuze! Er zijn ook opties om dit gedrag te veranderen of om handmatig het configuratieportaal en het webportaal onafhankelijk van elkaar te starten. Het is ook mogelijk om ze in non-blocking modus te laten werken. https://github.com/tzapu/

WiFiManager



U8glib (@olikraus - GitHub)

U8glib is een grafische bibliotheek met ondersteuning voor veel monochrome displays. De nieuwste versie van U8glib voor Arduino is beschikbaar in de Library Manager en kan ook worden gedownload van GitHub. Hij is compatibel met Arduino (ATmega en ARM), AVR, ARM (met voorbeeld voor LPC1114) en vele andere, zoals SSD1325, ST7565, ST7920, UC1608, UC1610, UC1701, PCD8544, PCF8812, KS0108 en meer. Deze bibliotheek ondersteunt vele lettertypen (zowel vaste breedte als proportioneel), muiscursormodus, landscape- en portrait-modus... Al met al een behoorlijk uitgebreide bibliotheek die voor veel projecten van pas zal komen. *https://github.com/olikraus/u8glib*



12:39 ② ● まゆ ・ る 秋湖宗 ad and 50% Sign in to Test-IoT 192.168.4.1	12:39 ② ● ま 歩 ・ 3 多 識 (学 uil unit 50%) Sign in to Test-IoT 192.168.4.1	12:39 ② ● c: ゆ ・ 8 約認念 atl anti 50% Sign in to Test-IoT 192.168.4.1	12:39 ⊗ ♥ ⊄ ♥ • ☎ ♥ 12:39 ⊗ ♥ ⊄ ♥ • 10 will will so% Sign in to Test-loT 192.168.4.1
WiFiManager	WIFI-RTU-46 .il	WiFiManager	No AP set
Test-IoT	SSID	Test-loT	esp32
Configure WiFi		Upload new firmware	Uptime
1-6-	Password	Choose file No file chosen	0 mins 40 secs
Into	Show Password	 May not runcion inside captive portal, open in proviser http://192.168.4.1 	3cb267ac
Exit			Chip rev 1
	Save		Flash size 4194304 bytes
Update	Refresh		PSRAM Size 0 bytes
	1		CPU frequency 240MHz
No AP set	No AP set		Memory - Free heap 180692 bytes available
			Memory - Sketch size Used / Total bytes 1199344 / 2510064
			Temperature 49.44 C* / 146.60 F*
			WiFi
			Connected No





Tasmota (@arendst - GitHub)

Tasmota is een open-source firmware ontworpen voor ESP apparaten, specifiek op maat gesneden voor domotica en smart homes. Het biedt ondersteuning voor ESP8266, ESP32, ESP32-S en ESP32-C3 microcontrollers. Theo Arends startte dit project in 2016 onder de naam Sonoff-MQTT-OTA. Zijn primaire

doel was om ESP8266-gebaseerde apparaten geproduceerd door ITEAD (Sonoff) uit te rusten met MQTT en over-the-air (OTA) firmwaremogelijkheden. Wat begon als een eenvoudige oplossing om een cloud-gebonden Sonoff Basic te wijzigen in een lokaal te beheren apparaat, is uitgegroeid tot een volledig ecosysteem dat geschikt is voor bijna elk ESP-gebaseerd apparaat. Tasmota is ontwikkeld voor PlatformIO en maakt eenvoudige configuratie mogelijk via een webgebaseerde gebruikersinterface (webUI). Updates kunnen draadloos worden uitgevoerd. Gebruikers kunnen apparaten automatiseren met timers of aangepaste regels. Volledige lokale bediening en uitbreidbaarheid zijn mogelijk via MQTT, HTTP, seriële of KNX-protocollen.

https://github.com/arendst/Tasmota

0

Esp32FOTA (@chrisjoyce911 - GitHub)

esp32FOTA is een eenvoudige bibliotheek om ondersteuning voor over-the-air updates aan je ESP32-project toe te voegen. Deze bibliotheek probeert toegang te krijgen tot een JSON-bestand dat op een webserver wordt gehost, analyseert de inhoud om te bepalen of er een nieuwere firmwareversie beschikbaar is en, zo ja, downloadt en installeert deze. Om dit update-proces te laten werken, heb je een webserver nodig met een geldig JSON-bestand erop (dat optioneel kan worden gecomprimeerd met zlib of gzip). De complete lijst met gedetailleerde vereisten (inclusief details over HTTPS) is beschikbaar op GitHub. Deze bibliotheek is behoorlijk uitgebreid en bevat ondersteuning voor gecomprimeerde firmware, SPIFFS/LittleFS partitie-updates en compatibiliteit met verschillende bestandssystemen voor het opslaan van certificaten en handtekeningen. Daarnaast biedt esp32FOTA ook webgebaseerde updates (met een webserver), batch firmware-syn-

chronisatie en de mogelijkheid om firmware-updates te forceren. Tot slot heeft de bibliotheek ook enkele geavanceerde functies zoals handtekeningcontroles voor gedownloade firmware-images, handtekeningverificatie en ondersteuning voor semantisch versiebeheer. In een notendop is het een bibliotheek die het ontdekken waard is als je OTA-updates wilt gebruiken voor je volgende project.

https://github.com/chrisjoyce911/esp32FOTA

chrisjoyce911/ esp32FOTA

Experiments in firmware OTA updates for ESP32 dev boards

RX 21 ③ 7 ☆ 301 ¥ 77 Contributors Issues Stars Forks



Willow (@toverainc - GitHub)

Willow is een IDF (IoT Development Framework) van Espressif dat wordt gebruikt om een ESP32-S3-BOX te veranderen in een veelzijdige spraakassistent met verschillende functies. Hij kan worden geactiveerd door aangepaste wekwoorden zoals "Hi ESP" of "Alexa" en luistert naar gesproken commando's, waarbij hij automatisch detecteert wanneer hij moet beginnen en stoppen met luisteren. Willow integreert goed met populaire domotica-platformen zoals Home Assistant, openHAB en generieke REST API's. Deze stemassistent presteert bewonde-renswaardig in veeleisende omgevingen en herkent wekwoorden en spraak vanaf een afstand van ongeveer 7,60 m. Hij zorgt voor audio van hoge kwaliteit met functies zoals automatische versterkingsregeling, echo-onderdrukking, ruisonderdrukking en bronscheiding. In omgevingen met veel WiFi-verkeer kan Willow audiocompressie toepassen om het gebruik van zendtijd te optimaliseren. Daarnaast biedt Willow on-device spraakherkenning, waarmee je lokaal tot 400 commando's kunt instellen. Je kunt ook de open-source Willow Inference Server gebruiken voor uitgebreidere spraaktranscriptiemogelijkheden. Kortom, een indrukwekkend project! *https://github.com/toverainc/willow*



Dit is een Arduino-bibliotheek voor het aansturen van NeoPixels. Dit zijn, in het jargon van Adafruit, individueel adresseerbare RGB-kleurenpixels en -strips gebaseerd op de WS2812-, WS2811en SK6812-LED's. Deze hebben elk een geïntegreerde driver en gebruiken een enkeldraads besturingsprotocol. Deze zijn vaak wat lastig te gebruiken vanwege de strakke timingvereisten voor de besturingssignalen en het specifieke protocol. Met deze bibliotheek wordt het veel gemakkelijker! Adafruit ontwikkelt en onderhoudt deze gratis. In ruil daarvoor kunnen gebruikers beslissen om enkele van hun producten aan te kopen. De belangrijkste doelstellingen van Adafruit NeoPixel zijn gebruiksvriendelijkheid en flexibiliteit in termen van ondersteunde chipsets. De bibliotheek ondersteunt AVR's (ATmega en ATtiny), Teensy, Arduino Due, Arduino 101, ATSAMD21/51, Adafruit STM32 Feather, ESP8266, ESP32, Nordic nRF51/52 en enkele IC's uit de XMC-serie van Infineon. Onder andere de vele goed geteste functies zoals setBrightness(), setPixelColor() en nog twaalf meer maken deze bibliotheek erg handig voor snelle prototyping.

https://github.com/adafruit/Adafruit_NeoPixel





ESP-DASH (@ayushsharma82 - GitHub)

ESP-DASH is een high-speed bibliotheek ontworpen om een functioneel en real-time dashboard op maat te maken voor ESP8266- en ESP32-microcontrollers. Het is ontwikkeld door GitHub-gebruiker ayushsharma82 en andere bijdragers, en het is momenteel bij versie 4 aangeland. Deze bibliotheek wordt geleverd met vele functies, waaronder grafieken, weergavekaarten, interactieve knoppen en tal van andere componenten, die het mogelijk maken om prachtige dashboards te maken. Deze zijn lokaal toegankelijk en vereisen geen internetverbinding, omdat alle gegevens op de chip worden opgeslagen. ESP-DASH genereert automatisch webpagina's en realtime-updates voor alle aangesloten clients. Gebruikers hoeven zich niet te verdiepen in HTML, CSS of JavaScript, omdat hij een gebruiksvriendelijke C++ interface heeft. Hij biedt voorgeconfigureerde componenten voor het beheer van je gegevens en biedt flexibiliteit - je kunt moeiteloos componenten toevoegen of verwijderen, direct vanaf de webpagina. Bovendien biedt hij ingebouwde ondersteuning voor grafieken, waardoor de functionaliteit wordt verbeterd. https://github.com/ayushsharma82/ESP-DASH

LVGL (@lvgl - GitHub)

Om bij het onderwerp dashboards en afbeeldingen te blijven: LVGL is een embedded grafische bibliotheek waarmee grafische gebruikersinterfaces kunnen worden gemaakt voor een groot aantal microcontrollers, waaronder natuurlijk ESP32 en Arduino (volledige lijst op GitHub). Deze bibliotheek is geschreven in C en is portable, dat wil zeggen dat hij niet werkt met externe afhankelijkheden. Hij kan worden gebruikt met of zonder een realtime-besturingssysteem (RTOS) en biedt uitgebreide ondersteuning voor displays, monochroom, ePaper, OLED enzovoort. Hij heeft 32 kB RAM en 128 kB flash-geheugen nodig. LVGL heeft een breed scala aan

> ingebouwde widgets, stijlen, layouts en meer, waardoor hij zeer aanpasbaar is. Animaties, anti-aliasing, transparantiecontrole, vloeiend scrollen, schaduwen, beeldtransformatie... de lijst gaat door en door. Verschillende invoermethoden zoals muis, touchpad en toetsenbord worden ondersteund. Make en CMake worden beide ondersteund, waardoor je op een PC kunt ontwikkelen en dezelfde UI-code op embedded hardware kunt gebruiken, wat voor sommige gebruikers een interessante functie kan zijn. *https://github.com/lvgl/lvgl*



Piëzo-elektrische componenten

vreemde onderdelen

David Ashton (Australië)

Het piëzo-elektrisch effect is al sinds de jaren 1880 bekend, maar hoe het van een wetenschappelijke curiositeit is uitgegroeid tot een belangrijk onderdeel van de moderne elektronica is een verhaal van innovatie en ontdekking. Deze unieke eigenschap van bepaalde kristallen heeft geleid tot het wijdverspreide gebruik ervan in de meest verschillende applicaties, van audiotransducers tot precisieinstrumenten in de astronomie en digitale beeldvorming.



Figuur 1. Piëzo-effect (bron: Sonitron Support – CC BY-SA 3.0, commons.wikimedia.org/w/index. php?curid=115322872).



Figuur 2. Een selectie van piëzo-signaalgevers. Dit zijn gewoon piëzo-elementen die met een wisselspanning moeten worden aangestuurd. A: Een typische piëzobuzzer. B: Dito, maar dan geopend om de constructie te laten zien. C: Een kristal-oortelefoontje. D: Een 100 W piëzohoorntweeter voor hifi-toepassingen.

Als op bepaalde kristallijne keramische materialen aan twee tegenovergestelde zijden van het kristal elektroden worden aangebracht en daar een spanning over wordt aangelegd, zal het kristal (enigszins) vervormen. Dit effect werd in 1880 voor het eerst aangetoond door de broers Pierre en Jacques Curie, hoewel het in 1861 al wiskundig was voorspeld door Gabrielle Lippmann. Gedurende de daaropvolgende decennia werd het echter beschouwd als niet meer dan een laboratoriumcuriositeit.

Dit effect kent vele toepassingen in de elektronica. Het meest bekend bij elektronicaliefhebbers zijn waarschijnlijk audiotransducers. Ik herinner me dat ik meer dan 50 jaar geleden mijn eerste kristalontvanger heb gebouwd met een kristaloortelefoontje – wat gewoon een piëzo-omvormer is die op een metalen schijfje is gemonteerd. **Figuur 1** toont het werkingsprincipe. Ze hebben het voordeel dat ze een vrij hoge impedantie hebben, wat belangrijk is voor een kristalontvanger, omdat deze dan niet zwaar belast wordt. Ze zijn ook erg dun, waardoor ze ook worden



Figuur 3. Diverse piëzo-buzzers (piepers). Deze hebben elektronica aan boord om het element aan te sturen. Achter: een 12V-piëzosirene. Midden: drie verschillende piëzo-buzzers. Voor: een AC/DC-piëzozoemer voor gebruik in industriële schakelborden (geschikt voor meerdere spanningen).



Figuur 4. Een piëzo-ontsteker uit een gasaansteker. Druk op de knop aan het uiteinde en hij 'klikt' en genereert dan een hoogspanningspuls die een vonk produceert tussen twee elektroden.

toegepast in wenskaarten die een melodietje spelen als je ze opent. Er zijn grotere transducers verkrijgbaar met een tenminste enigszins fatsoenlijke frequentierespons; sommige hifi-tweeters maken gebruik van piëzotechnologie (figuur 2). De meeste van deze transducers hebben een resonantiefrequentie waarbij het rendement het grootst is; hiervan wordt gebruik gemaakt in buzzers en sounders met een ingebouwde oscillator die de schijf (dicht) bij de resonantiefrequentie aanstuurt. Ze worden gebruikt als 'piepers' en alarmgevers (figuur 3) om waarschuwingssignalen te produceren. Ze hebben het voordeel dat ze heel efficiënt zijn en voor veel lawaai maar weinig stroom nodig hebben.

Andere toepassingen van piëzo-transducers zijn inkjetprinters (een kleine beweging wordt gebruikt om een druppeltje inkt te 'lanceren') en astronomische telescopen – piëzo-actuatoren worden gebruikt om zeer kleine correcties aan te brengen in de vorm van de zware spiegel die wordt gebruikt om het licht te verzamelen, om vervorming en onscherpte te minimaliseren. Ze kunnen relatief grote krachten uitoefenen over zeer kleine afstanden. Een andere toepassing is beeldstabilisatie in digitale camera's, waar een kleine maar nauwkeurige beweging met een snelle reactietijd nodig is.

Het effect is omkeerbaar – dat wil zeggen, als een piëzo-element wordt gebogen of vervormd, genereert het een spanning over de elektroden. Dit wordt gebruikt in gasontstekers, waarbij een element langzaam wordt vervormd en dan snel weer zijn oorspronkelijke vorm aanneemt door middel van een soort ratelmechanisme (**figuur 4**). Daarbij een puls met een zeer hoge spanning gegenereerd – genoeg om een vonk te produceren over twee elektroden waarmee het gas wordt ontstoken. Ze worden ook gebruikt in trillingssensoren, waar hun kleine formaat en lage kosten een groot voordeel zijn.

230684-03

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.



Over de auteur

David Ashton is geboren in Londen, groeide op in Rhodesië (nu Zimbabwe), woonde en werkte in Zimbabwe en woont nu in Australië. Hij is al geïnteresseerd in elektronica sinds hij 'drie turven hoog' was. Rhodesië was niet het middelpunt van het elektronica-universum, dus aanpassen, vervangen en onderdelen bij elkaar scharrelen waren vaardigheden die hij al vroeg aanleerde (en waar hij nog steeds trots op is). Hij heeft een elektronicalab gerund, maar heeft voornamelijk in de telecommunicatie gewerkt.



Slimme



objecteller eenvoudige beeldherkenning met Edge Impulse

Somnath Bera (India)

Ontdek hoe je een Raspberry Pi en een camera kunt transformeren in een slimme tool voor het tellen van objecten met behulp van het Edge Impulse-platform! Dit leuke en toegankelijke project laat zien hoe eenvoudig het is om te beginnen met Edge Impulse op een Raspberry Pi – perfect voor zowel beginners als gevorderden.

Edge Impulse is gespecialiseerd in het leveren van tools en platforms voor het ontwikkelen van machine learning-modellen voor edge computing, met name op embedded apparaten. Edge computing houdt in dat gegevens dichtbij de bron worden verwerkt, in plaats van op een externe server te vertrouwen. Dat kan perfect worden geïmplementeerd op een Raspberry Pi! In dit voorbeeld tellen we kleine objecten – simpele knopen op textiel.

Machine learning-platforms zoals Edge Impulse maken gebruik van zogenaamde modellen. Dat zijn specifieke soorten algoritmen die worden gebruikt voor gegevensanalyse en patroonherkenning. Deze modellen worden getraind om patronen te herkennen, voorspellingen te doen of taken uit te voeren op basis van ingevoerde gegevens. Objecten classificeren is eenvoudig mogelijk met Edge Impulsemodellen. Je kunt onderscheid maken tussen een mens en dieren of tussen fietsen en auto's om maar een paar voorbeelden te noemen. Bovendien kun je gemakkelijk een type object tellen temidden van andere soorten objecten. Het enige wat je nodig hebt is een camera van goede kwaliteit, voldoende licht, de juiste scherpstelling en, tot slot, een redelijke computer (een Raspberry Pi 3 of Raspberry Pi 4 is goed genoeg), en je bent klaar om te gaan tellen.

Vanaf het begin was dit project bedoeld voor implementatie op microcontroller-niveau – een Espressif ESP32, een Arduino Nicla

vision of iets in die geest. En daarom werd het gebouwd voor een zeer klein telgebied (120 × 120 pixels) met een relatief kleine knoop als object waar we in geïnteresseerd zijn. Uiteindelijk bleek dat zelfs voor zo'n klein gebied een MCU geen partij was. De modellen voor machine learning worden vooraf getraind op de Edge Impulse-servers en daarbij wordt een zogenaamd modelbestand gegenereerd dat wordt opgeslagen op het embedded apparaat. Dit modelbestand zelf is al ongeveer 8 MB groot! Daarom werd het project uiteindelijk geïnstalleerd op een Raspberry Pi computer, waar het probleemloos werkt.

Kennis versus wijsheid

Als je Edge Impulse [1] kent, dan is de helft van het werk al gedaan - geloof me. Voor de rest hoef je alleen maar je model aan te passen zodat een acceptabel prestatieniveau wordt bereikt. Een computer-AI model is als een kind. Als kind leerde je dingen als "A is een appel" en "B is een bal". Je kreeg een appel te zien vanuit verschillende richtingen en dan leerde je om dat ding "appel" te noemen. Hetzelfde gold voor "bal". En dan zal een kind vanuit alle mogelijke hoeken vrij gemakkelijk een appel en een bal herkennen! En zo is het ook met AI, die ze gemakkelijk kan identificeren. Stel je nu eens voor dat er een mand is waar ogenschijnlijk ballen ter grootte van een appel en appels ter grootte van een bal door elkaar liggen, en die er tot overmaat van ramp allemaal hetzelfde uitzien. Wat zou jij als kind doen? Met uitsluitend kennis van "appel" en "bal" zou je ze niet uit elkaar kunnen houden! Hetzelfde geldt voor AI. Maar stel dat die mand is uitgestald door een groente- en fruitverkoper. Naar alle waarschijnlijkheid zitten er dan geen ballen bij! En het zouden best allemaal appels kunnen zijn. Deze 'truc' om een appel in verband te brengen met een groente- en fruitverkoper zou je "wijsheid" kunnen noemen, wat je noch van een kind noch van AI kunt verwachten, tenzij het specifiek anders is aangeleerd. De mens heeft in de loop der jaren echter veel meer associaties geleerd die ons uiteindelijk genoeg wijsheid hebben gegeven om een appel met een groente- en fruitverkoper in verband te brengen. AI wordt echter zo snel zoveel beter dat het ooit met deze 'wijsheid' zal kunnen werken. Voorlopig moet je het ML-model voor appels en ballen vanuit alle mogelijke hoeken aanleren om ze zonder enige verwarring te kunnen herkennen (bijvoorbeeld het textuurprofiel

	of the fatest updates in edge Africial endeaded wit moustly leade	ers, visionaries, and researchers, sept a	Bera / count_t _	
Dathbard	#1 ▼ Click to set a description for this versi	ion		
Devices	Neural Network settings		:	
Data acquisition	Training settings			
Impulse <mark>d</mark> esign	Number of training cycles ③	60		
Create impulse	Learning rate ③	0.001	2	
Image	Data augmentation ③			
EON Tuner	Advanced training settings			
Retrain model	Validation set size 🕥	20	%	
Live classification	Split train/validation set on metadata key ③			
Model testing	Profile int8 model ③			
Versioning	Neural network architecture			
Deployment	Input layer (43	3,200 features)		
ry Enterprise Free				
t access to high job limits and training on GPUs.	(1	7		
Start free trial	FOMO (Faster Objects, Mor	e Objects) MobileNetV2 0.1		Figuur 1. Instellingen voor h

van een appel, de steel, de plooien van het vruchtlichaam, de blik van bovenaf en van onderaf en nog veel meer). In elk geval zijn er veel verschillende modellen met verschillende mogelijkheden beschikbaar in Edge Impulse om te testen en om mee te experimenteren.

Aan de slag met Edge Impulse

Open eerst een account in Edge Impulse [1], waarvoor je een e-mail ID nodig hebt. Zorg dat je een handjevol gelijke knopen bij de hand hebt. Als je de site opent vanaf een Raspberry Pi-computer, kun je met de camera van de Raspberry Pi (aangesloten op USB of cam-poort) beelden van knopen vanuit verschillende hoeken verzamelen (wat nodig is als het model voor daadwerkelijk gebruik wordt uitgerold). Edge Impulse heeft ook voorzieningen om je mobiele telefoon of laptop aan te sluiten als invoerapparaat voor het verzamelen van gegevens, wat ook handiger is voor data-acquisitie in het Edge Impulse-project.

Het project

Het Edge Impulse-project is grofweg onderverdeeld in de volgende stappen, die allemaal moeten worden gevolgd op de Edge Impulse-website.

- 1. Data-acquisitie: dit kunnen beelden, geluid, temperaturen, afstanden enzovoort. zijn. Een deel van de gegevens wordt apart gehouden als testgegevens, terwijl alle andere gegevens worden gebruikt als trainingsgegevens.
- 2. Impulse-ontwerp: het belangrijkste deel wordt Create Impulse genoemd. In deze context is een 'Impulse' naar een pijplijn of workflow voor het maken van een machine learning-model. Deze

Impulse omvat verschillende stadia, waaronder het aanpassen van invoerparameters die zijn gekoppeld aan de zojuist verzamelde gegevens, signaalverwerking, extractie van kenmerken en het machine learning-model zelf. 'Kenmerken' zijn individuele meetbare eigenschappen of karakteristieken van een geobserveerd fenomeen. In wezen zijn kenmerken de data-attributen die door modellen worden gebruikt om patronen te detecteren en beslissingen te nemen. De Impulse-pijplijn is onderverdeeld in:

- invoerparameters: afbeelding (breedte, hoogte), geluid (geluidsparameters);
- verwerkingsblok: hoe de invoergegevens te verwerken;
- leerblok: objectgegevens van dit model.

Je moet deze drie stappen selecteren en configureren.

- 3. Beeldverwerking: kenmerken genereren van de verzamelde beelden.
- 4. Objectdetectie: selecteer je neurale netwerkmodel en train het.

Voor het laatste gedeelte - de objectdetectie - is je expertise nodig, of veeleer 'vallen-en-opstaan', zodat de nauwkeurigheid van het model 85% of meer gaat bedragen. Soms moet je een aantal slechte afbeeldingen (ook wel uitschieters genoemd) uit het model verwijderen om de efficiëntie te verbeteren.

Er is een handvol modellen waarbij je kunt proberen hoe nauwkeurig het model is. Alles boven de 90% is geweldig, maar het hoeft zeker niet voor 100% nauwkeurig te zijn! Als dat wel zo is, dan is er iets mis met je gegevens. Het kan zijn dat er erg weinig data zijn of





Figuur 2. Scan de QR-code om dit model uit te voeren.

dat er onvoldoende kenmerken zijn. Controleer en probeer het in dat geval opnieuw! Voor dit project was de nauwkeurigheid 98,6%. Natuurlijk was ons aantal gegevens (ongeveer 40) klein. Maar voor een instapproject is dit behoorlijk goed (zie **figuur 1**). De bestanden voor dit project zijn beschikbaar op de Elektor Labs-pagina bij dit artikel [4].

Test van het model

Je kunt je model eerst testen op de testgegevens. Begin daar en richt je apparaat dan op de echte objecten en kijk of het werkt! In het dashboard van de openingspagina van Edge Impulse is de testfunctie beschikbaar. Je kunt het model direct in de browser uitvoeren, maar je kunt ook je smartphone gebruiken om het te testen. Hiervoor biedt Edge Impulse een QR-code die je kunt scannen met je smartphone (**figuur 2**). Richt de camera da op de knopen (**figuur 3**, **figuur 4** en **figuur 5**) en kijk of ze correct geteld worden.

Raspberry Pi-implementatie

Om het model op een Raspberry Pi computer uit te voeren, moet je het *.eim bestand downloaden. Maar in tegenstelling tot andere hardware (Arduino, Nicla Vision of ESP32, waar je direct kunt downloaden), moet je in het geval van de Raspberry Pi eerst Edge Impulse installeren op de Raspberry Pi-computer. Vanuit die edgeimpulse-daemon software moet je dit bestand downloaden. Maar maak je geen zorgen, Edge Impulse heeft een volledige pagina gewijd aan het installeren van Edge Impulse op de Raspberry Pi. Er zijn een paar afhankelijkheden die je eerst moet installeren. Werp eens een blik op [2], alles is vrij eenvoudig. Het proces is goed beschreven.

OK, dus nadat je Edge Impulse hebt geïnstalleerd op de Raspberry Pi-computer, kan de pret beginnen. Vergeet niet om de Raspberry Pi verbonden te houden met het internet.

Voer het commando edge-impulse-linux-runner uit vanaf de Raspberry Pi-terminal. Dit start een wizard die je vraagt om in te loggen en een Edge Impulse-project te kiezen. Als je later tussen projecten wilt wisselen, voer dan dat commando opnieuw uit met de optie --clean. Dit commando zal automatisch het AI-model van je project compileren en downloaden en vervolgens starten op je Raspberry Pi. Laat de knopen zien aan de camera die is aangesloten op je Raspberry Pi en hij zou ze moeten tellen. Dat is goed! In het volgende zullen we het systeem aanpassen met Python en een spraaksynthesizer die, na het tellen, het aantal knopen noemt.

Model implementeren in Python

In de bovenstaande opstelling werkt alles zoals bedoeld in het Edge Impulse-model. Om het te laten werken voor jouw speciale doel –



Figuur 3. Data-acquisitie: voorbeeld 1.



Figuur 4. Data-acquisitie: voorbeeld 2.



Figuur 5. Data-acquisitie: voorbeeld 3.

bijvoorbeeld om een geluidsalarm te laten klinken of een LED te laten oplichten wanneer '2 of meer' knopen zijn geteld – moet je iets anders bedenken! Hier komt Python 3 je te hulp. *Linux-sdkpython* moet geïnstalleerd zijn op je Raspberry Pi-computer. De Edge Impulse SDK Software Development Kit (SDK) is beschikbaar voor veel omgevingen, waaronder Python, Node.js, C++ enzovoort. Bekijk daartoe de SDK Python-pagina [3]. Zodra *linux-sdk-python* is geïnstalleerd, ga je naar de map *linuxsdk-python/examples/image* en voer je het Python-bestand voor beeldherkenning uit. Pas op: vergis je niet. In de voorbeeldmap staan drie submappen – één voor audiodata, één voor beelddata en één voor eigen data. In de beelddata-map is het video-classificatiebestand ook beschikbaar voor video-invoerdata. De map voor eigen data is voor het aanpassen van andere soorten gegevens (alleen voor experts!).

Voer nu deze opdracht uit:

python3 classify-image.py /home/bera/downloads/model.eim

Het modelbestand *.eim moet worden geladen vanuit de directory



Figuur 6. Vier knoppen worden gemist door ontoereikende scherpstelling en verlichtina.





Figuur 7. Na het verwijderen van één knoop telde het model ze in eerste instantie correct.

waar het zich bevindt. Je kunt het desgewenst ook naar de SDK-directory kopiëren!

Zo moet je het Python-bestand laden met het gedownloade model. eim-bestand. Het programma vindt automatisch de cameramodule (aangesloten op USB- of Cam-poort) en start. In de linkerbovenhoek wordt een klein 120×120-cameravenster geopend en de geïdentificeerde knopen worden met een kleine rode stip gemarkeerd. De geïdentificeerde nummers worden getoond in de terminal. Zorg ervoor dat er voldoende licht is en dat de camera goed is scherpgesteld op de knopen. Dit is vooral belangrijk voor goedkope camera's. Als je het model op je smartphone uitvoert, worden veel betere beelden geproduceert en wordt veel sneller geteld. Zorg desondanks voor goed licht en scherpstelling voor het beste resultaat. In de volgende screenshots van de Raspberry Pi-computer is linksboven het kleine beeldvenster zichtbaar waarin de knopen worden

geïdentificeerd en geteld door het model. Zie de duidelijk zichtbare rode markering op alle knoppen!

In figuur 6 worden vier knopen gemist vanwege ontoereikende scherpstelling en verlichting. De hier gebruikte camera kan ook niet op een statief worden vastgezet! Daarom raad ik aan om de camera op een statief te bevestigen (vergelijkbaar met een microscoop). Zorg ervoor dat je met de camera recht op de knopen kijkt. In figuur 7 heb ik één knoop verwijderd en het model telde in eerste instantie goed. Er werden twee knopen herkend, maar op het moment dat ik op de printscreen-knop drukte, verliep de uitlijning en ging de telling mis. Zorg er ook voor dat de camera een lange kabel (lintkabel - zie mijn prototype in figuur 8) heeft voor de nodige bewegingsvrijheid. Zo'n kabel is verkrijgbaar bij Amazon. Maar als de camera eenmaal op een statief is gemonteerd bij voldoende licht/daglicht, zal hij prima werken.



Figuur 8. Mijn prototype.

Aanpassing van het model

Werp eens een blik op het bestand *classify-image.py*. Het is een eenvoudig Python-bestand dat met weinig moeite aan individuele behoeften kan worden aangepast. In dit Python-bestand heb ik een *espeak*-module toegevoegd zodat op het moment dat een knoop of knopen worden gedetecteerd, het aantal gedetecteerde knopen wordt uitgesproken. Om *espeak* op je Raspberry Pi te installeren, moet je het volgende commando uitvoeren:

sudo apt-get install espeak

Zie **listing 1** met het Python-bestand inclusief mijn aanpassingen. Espeak is een zelfstandige tekst-naar-spraak-module voor Python. Voor het gebruik is geen internetverbinding nodig.

Aangepaste uitvoering

Nu heb je het Python-programma aangepast. Als je het nu uitvoert, zal het de knopen vinden (linksboven wordt een klein 120×120display geopend); de gedetecteerde aantallen zullen worden weergegeven in het terminalvenster en via de luidspreker tekstueel worden genoemd: *"Found five buttons / Found two buttons"* enzovoort. Als je een relais wilt aansturen, een LED wilt laten branden of iets dergelijks, moet je de GPIO-bibliotheek van Python importeren en de betreffende GPIO activeren voor de gewenste actie. In het geval van een relais moet je echter een drivertransistor gebruiken die de stroom kan verwerken die het relais nodig heeft.

Nasleep

Edge Impulse startte in 2019 met als doel ontwikkelaars in staat te stellen de volgende generatie intelligente apparaten te maken. Sindsdien zijn er AI-gestuurde programma's en apparaten verschenen voor ESP32, Jetson Nano, Raspberry Pi, Orange Pi, Maixduino, OpenMV, Nicla Vision en nog veel meer. Deze trend zal de komende tijd aanhouden! De dagen van supercomputers of 'grote' computermerken zijn voorbij. Kleine modulaire apparaten met een gering stroomverbruik nemen hun plaats in toenemend tempo in. En wie weet – misschien hebben we binnenkort de ingebouwde wijsheid kant-en-klaar in een doosje!

230575-03

Vragen of opmerkingen?

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

......

Listing 1. Python-programma om het aantal knopen uit te spreken met het espeak-tool.

```
#!/usr/bin/env python
                        # Device Specific patches - taken care by the software
import device_patches
import cv2 #import Computer Vision
import os
import sys, getopt
import signal
import time
from edge_impulse_linux.image import ImageImpulseRunner
import subprocess #this one have been added by Bera
       elif "bounding_boxes" in res["result"].keys():
                   print('Found %d bounding boxes (%d ms.)' % (len(res["result"]["bounding_boxes"]),
                   res['timing']['dsp'] + res['timing']['classification']))
                   if (len(res["result"]["bounding_boxes"])>0):
                           exitCode = subprocess.call(["espeak","-ven+f3","-a200"," Found %d Buttons" %
                           len(res["result"]["bounding_boxes"]) ]) #This one have been added by Bera
```



Over de auteur

Somnath Bera, een werktuigbouwkundig ingenieur van Jalpaiguri Govt. Engg. College, India, werkt als General Manager bij NTPC, de grootste energieproducent van het land. Hij heeft een grote passie voor elektronica, wat blijkt uit zijn meer dan 60 innovatieve projecten op Elektor Labs, waarvan er meer dan 10 zijn gepubliceerd in Elektor Magazine. Zijn projecten zijn vaak gericht op het oplossen van problemen op het gebied van afval en het beheer van natuurlijke hulpbronnen. Somnath maakt graag gebruik van innovatieve benaderingen en platforms zoals Arduino, Raspberry Pi en ESP32 in combinatie met verschillende soorten sensoren en draadloze systemen om efficiënte en kosteneffectieve oplossingen te creëren.

Gerelateerde producten

- Raspberry Pi 4 B (1 GB RAM) www.elektor.nl/18966
- G. Spanner, Machine Learning with Python for PC, Raspberry Pi, and Maixduino (E-book, Elektor) www.elektor.nl/20150
- > D. Situnayake, Jenny Plunkett, Al at the Edge (O'Reilly) www.elektor.nl/20465

WEBLINKS

- [1] Edge Impulse: https://edgeimpulse.com/
- [2] Installeren van Edge Impulse op een Raspberry Pi 4: http://tinyurl.com/ysc6mtuz
- [3] Linux Python SDK: http://tinyurl.com/2bat4w6z
- [4] Projectpagina op Elektor Labs: https://www.elektormagazine.com/labs/count-speak-number-of-buttons



Oplossingen voor de lastigste uitdagingen bij embedded ontwikkeling

Stuart Cording (voor Mouser Electronics)

De ontwikkeling van embedded systemen biedt engineers veel mogelijkheden, maar brengt nog meer uitdagingen met zich mee! Deze zijn zo divers en gevarieerd als het maar zijn kan. De unieke combinatie van elektronica. software en systemen biedt mogelijkheden tot speuren in het analoge domein en problemen oplossen op het gebied van digitale timing, evenals het onderzoeken van complexe problemen met regelkringen of het beoordelen van de kwaliteit van sensormetingen. Fabrikanten van testapparatuur hebben dit allang onderkend en reageren met slimme functies. nieuwe functionaliteiten. connectiviteitsmogelijkheden en smartphone-apps om het leven van de ontwikkelaar te vergemakkelijken.



Figuur 1. Het B&K Precision Model 2194 biedt een indrukwekkende geheugendiepte van 14 Mpts. De zoekfunctie maakt het gemakkelijk om signaalafwijkingen te vinden.

Moderne oscilloscopen geven meer weer dan alleen de spanning als functie van de tijd. Ze voeren ook complexe wiskundige berekeningen uit op inkomende signalen om analyse mogelijk te maken, seriële datasignalen te decoderen en op complexe signalen te triggeren. Dan zijn er ook nog de op FPGA-technologie gebaseerde 'softe' oscilloscopen die meer weg hebben van een minilab. Deze bieden het standaardscala aan meetmogelijkheden, maar kunnen dankzij hun programmeerbaarheid specifiek worden afgestemd voor het uitvoeren van nog veel meer taken. Ze zijn een goed alternatief voor iedereen die graag zijn laptop als scherm gebruikt, maar te weinig ruimte heeft op zijn werkbank.

Zoeken naar dwergpulsen met oscilloscopen

Met de komst van de digitaliserende oscilloscoop is het zoeken naar de oorzaken van problemen veel eenvoudiger geworden. Deze nieuwe methode maakt namelijk een foto van het CRT-scherm. De huidige apparatuur is eenvoudig aan te sluiten op netwerken en computers, waardoor de vastgelegde signalen beoordeeld en gedeeld kunnen worden. Dit vereenvoudigt de configuratie en het geautomatiseerd testen. In dit opzicht lijkt de B&K Precision Model 2194 [1] veel op andere vierkanaals-oscilloscopen op de markt (figuur 1). Zoals met alle belangrijke investeringen is het de moeite waard om de handleiding door te lezen om alle mogelijkheden te ontdekken. De 2194 met zijn respectabele 100 MHz ingangsbandbreedte, 1 GSa/s maximale bemonsteringsfrequentie en 7 inch TFT-display van 800 × 480 pixels is net zo goed uitgerust als vergelijkbare apparaten, maar dan voor de helft van de prijs. Met zijn gewicht van 2,6 kg en afmetingen van ongeveer 31×15×13 cm is hij eenvoudig verplaatsbaar en ruimtebesparend. De BNC-connectoren en USB-aansluiting voor opslagmedia aan de voorkant zijn gemakkelijk toegankelijk. Aan de achterkant bevinden zich de USB-aansluiting voor de besturing, Ethernet en de BNC-triggeruitgang (ook geschikt als pass/fail-uitgang voor automatisch testen). Het apparaat is daarnaast voorzien van een bevestigingspunt voor kabelsloten voor laptops.





Figuur 2. Dwergpulsen veroorzaakt door problemen met de driverschakeling kunnen worden gebruikt als trigger of worden gevonden in opgenomen signalen met behulp van de zoekfunctie van de 2194.

Figuur 3. De Moku:Go lijkt meer op een minilab dan op een oscilloscoop en biedt acht verschillende testapparaten in één unit net iets groter dan uw hand.

De 2194 beschikt tevens over 38 verschillende meetparameters en statistieken voor signaalanalyse, waaronder aantal, standaardafwijking, stijg- en daaltijden en piek-tot-piek, en hij decodeert ook I²C, SPI, UART, CAN en LIN.

Een updatesnelheid van 100.000 wfms/s helpt bij het vinden van storingen en andere onregelmatigheden. Ook de geheugendiepte is indrukwekkend. Dankzij de beschikbare 14 Mpts kunnen gedurende langere perioden signalen worden vastgelegd en, met behulp van de zoomfunctie, worden geanalyseerd op onregelmatigheden. Het zoeken naar anomalieën in zulke enorme datasets kan echter een uitdaging zijn. Dankzij de zoekfunctie van de 2194 kan een scala aan zaken worden gevonden, waaronder flanken, hellingen, pulsen en intervallen.

Een andere uitdaging voor ontwikkelaars is het vinden van digitale signalen die niet goed worden uitgestuurd. Deze staan bekend als dwergpulsen (runt pulses, figuur 2) en treden op wanneer de driver onvoldoende stijgsnelheid (slew rate) heeft om binnen de beschikbare tijd het vereiste logische niveau te bereiken. Dwergpulsen kunnen op de 2194 'gewoon' worden gezocht, maar kunnen ook als trigger worden gebruikt. Hiervoor moet zowel een hoog als een laag triggerniveau worden gedefinieerd. Mocht een signaal de ene drempel overschrijden, maar de andere niet, dan legt de trigger de golfvorm vast om deze te analyseren, zodat moeilijk te vinden sporadische storingen eenvoudiger opgespoord kunnen worden.

Liquid Instruments Moku:Go

Voor wie vaak onderweg is of in het veld werkt, kan een traditionele oscilloscoop een zwaar en onhandelbaar instrument zijn. Er zijn echter alternatieven als u bereid bent uw laptop als visuele interface te gebruiken. Eén zo'n tool is de Liquid Instruments Moku:Go [2], een compact apparaat van 24×3,8×13 cm met een gewicht van slechts 750 gram (**figuur 3**). De Moku:Go is dankzij het FPGA-gebaseerde ontwerp eigenlijk een combinatie van acht instrumenten, waaronder een 30 MHz oscilloscoop en real-time spectrumanalyser, 20 MHz golfvormgenerator, een PID-controller en een willekeurige golfvormgenerator, om er maar een paar te noemen.

De Moku:Go MO-unit komt met twee analoge ingangen, twee analoge uitgangen en 16 digitale I/O's. Hij is eenvoudig aan te sluiten op zowel Windows- als Mac-machines via USB-C. De M1 voegt een tweekanaals programmeerbare voeding (PSU) toe voor -5 V tot +5 V en 0 V tot 16 V bij 150 mA, samen met de benodigde kabels. En de M2 heeft daarnaast nog twee programmeerbare PSU's van 0,6 V tot 5 V bij maximaal 1 A per kanaal, compleet met Ethernet-connectiviteit.

Hoewel programmeerbare PSU's misschien niets bijzonders lijken, komen ze zeker van pas als u beseft dat de opgenomen stroom kan worden gebruikt als parameter in testopstellingen. Dit wordt gedemonstreerd in de application note "Converter Evaluation Using TI TPS63802EVM" [3], waarin wordt getoond hoe het rendement van een buck/boost-converter kan worden gemeten. De beide PSU's in combinatie met de twee analoge kanalen zijn meer dan voldoende om het rendement te

meten; het Math-kanaal levert, wanneer voorzien van de belastingsweerstand, heel handig het vermogen in watt. Geautomatiseerde meting is ook mogelijk met Python, MATLAB en LabView programmeer-API's. Het is lastig om in een paar bladzijden de Moku:Go uitputtend te beschrijven. Naast de indrukwekkende application notes [4] mag ook de bij de tool behorende app niet onvermeld blijven [5]. Ten eerste is de GUI schoon en netjes en dankzij de demomodus kunt u alle verschillende functies uitproberen. Dit geeft een indruk van hoe de X- en Y-as in- en uitzoomen en hoe de kanalen en rekenfuncties zijn geconfigureerd. Als u twijfelt over de deze tool iets voor u is, moet u dan eerst de app uitproberen en pas dan beslissen.

Componenten testen en sorteren

Het ontwerpen van voedingsschakelingen vereist een diepgaande kennis van de gebruikte spoelen en condensatoren. Bij uiterst accurate voedingen moet elk onderdeel wordt getest en gesorteerd. Ongeacht of uw uitdaging zich in het ontwerplab of op de fabrieksvloer bevindt, de Teledyne LeCroy T3LCR Series [6] LCR-meters kunnen u daarbij helpen (**figuur 4**). Ze hebben een groot 3,5" TFT-scherm en kunnen eenvoudig worden geconfigureerd voor vierdraadsmetingen via het frontpaneel, of de USB- of



Figuur 4. De Teledyne LeCroy T3LCR kan in het ontwerplab worden gebruikt om componenten te karakteriseren, terwijl hij binnen een productieomgeving deze kan sorteren ('binning').





Figuur 5. De meters van de Extech 250W-serie maken een reeks omgevingsmetingen mogelijk, waaronder metingen van geluidsniveau, toerental, lichtsterkte, relatieve vochtigheid en luchtsnelheid.

RS-232C poorten. De drie T3LCR-modellen ondersteunen testfrequenties van 10 Hz tot 2 kHz, 100 kHz of 300 kHz en bieden elk een basisnauwkeurigheid van 0,05 %.

Functies zoals automatische niveauregeling (automatic level control, ALC) garanderen de constante testspanning voor MLCC's, terwijl de instelbare teststroom past voor metingen van zelfinducties. De instelbare ±2,5 V interne DC-voorspanning kan worden gebruikt om gelijktijdig AC en DC te simuleren voor het beoordelen van capaciteitsvariaties. In de modus List Measurement kunnen tot 10 geautomatiseerde meetparameters worden verzameld, zodat componenten kunnen worden gekarakteriseerd inclusief eventuele variatietrends. De achterzijde van de T3LCR is voorzien van een DB-25 connector voor de koppeling met een handler voor het kwalificeren en sorteren ('binning') van componenten met ondersteuning voor maximaal tien categorieën ('bins'). Er zijn zowel equivalente serieals parallel-modelmetingen beschikbaar voor weerstand, zelfinductie en capaciteit. Daarnaast kunnen nog een tiental parameters worden gemeten, waaronder dissipatiefactor (D), kwaliteitsfactor (Q), fasehoek en gelijkstroomweerstand.

Houd de omgeving in de gaten

Bij de ontwikkeling van embedded systemen is het vaak noodzakelijk om door de microcontroller verzamelde sensorgegevens te vergelijken met metingen van professionele testapparatuur. Soms vraagt de situatie om langetermijnmetingen, bijvoorbeeld wanneer milieuparameters worden gemeten. De 250W-serie Bluetooth Connected Environmental Meters van Extech [7] zijn hiervoor ideaal (**figuur 5**) en dekken alles, van toerentalmetingen, lichtsterkte en relatieve vochtigheid tot geluidsniveau en luchtsnelheid. Met hun afmetingen van 54×28×120...176 mm en hun grote en helder verlichte LC-displays liggen de apparaten goed in de hand en kunnen ze eenvoudig worden afgelezen.

De lasertachometer RPM250W meet toerentallen tussen 10 en 99.999 rpm met een nauwkeurigheid van 0,04%, terwijl de LT250W geschikt is voor het meten van lichtsterktes tot 100.000 lux. De compacte luchtstroommeter AN250W levert zijn resultaten in ft/min, m/s en knopen, samen met de temperatuur van de omgevingslucht. De SL250W voor het geluidsdrukniveau biedt A-gewogen 'menselijk gehoor'-frequentiemetingen met registratie van de minimale en maximale waarden. De RH250W ten slotte meet en registreert de relatieve vochtigheid en de temperatuur.

Elk apparaat biedt Bluetooth-connectiviteit, waardoor de units aangesloten kunnen worden op een iOS- of Android-apparaat en geschikt zijn voor gebruik met de Extech ExView-app. De app kan gelijktijdig gegevens verzamelen van maximaal acht meters uit de 250W-serie en kan worden geconfigureerd om een akoestische alarm te genereren op hoge/lage niveaus. Gegevens worden lokaal opgeslagen en kunnen worden gedeeld in.csv-format. Bovendien kunnen foto's van testlocaties worden ingesloten in PDF-rapporten met meetgegevens.

Een meetinstrument voor elke embedded ontwikkelaar

Hoewel het ontwikkelen van embedded systemen divers en leuk is, vereisen sommige uitdagingen slimme testapparatuur om de oorzaak van sporadische storingen te achterhalen, het werkelijke rendement van omvormers te bepalen of omgevingsmetingen te vergelijken met de resultaten van embedded sensoren. In andere gevallen is het cruciaal om de exacte waarde van de gebruikte componenten te kennen, of deze tijdens de productie te categoriseren. Hopelijk helpt een van de hier besproken testoplossingen u op weg, ongeacht wat uw embedded ontwikkelingsuitdaging is. ►

230750-03



Over de auteur

Stuart Cording is een freelance journalist die onder meer schrijft voor Mouser Electronics. Hij is gespecialiseerd in videocontent en is gefocust op technische diepgang en inzicht. Hierdoor is hij vooral geïnteresseerd in de technologie zelf, hoe deze past in eindtoepassingen en in voorspellingen over toekomstige ontwikkelingen.

Mouser Electronics is een geautoriseerde distributeur van halfgeleiders en elektronische componenten die zich richt op de introductie van nieuwe producten van haar toonaangevende partners.

WEBLINKS

- [1] B&K Precision Model 2194: https://eu.mouser.com/new/bk-precision/bk-2194-oscilloscope/
- [2] Liquid Instruments Moku:Go: http://tinyurl.com/Moku-Go
- [3] Converter Evaluation Using TI TPS63802EVM: http://tinyurl.com/AppNoteConverter
- [4] Application Notes van Liquid Instruments: https://www.liquidinstruments.com/blog/category/application-notes/
- [5] Windows & macOS apps van Liquid Instruments: https://www.liquidinstruments.com/products/desktop-apps/
- [6] Teledyne LeCroy T3LCR: http://tinyurl.com/TeledyneT3LCR
- [7] Extech 250W: https://eu.mouser.com/new/extech/extech-250w-meters/

ESP32 Terminal

handheld met aanraakscherm

Johan van den Brande (België)

De Elecrow ESP32 Terminal [1] is een handheld apparaat op basis van een ESP32-3 met een 3,5" 480 × 320 capacitief TFT-aanraakscherm dat talloze mogelijkheden biedt. Het apparaat kan een mooie aanvulling zijn op je projecten als ie een aanraakscherm met verschillende soorten interfaces nodig hebt.

Het display op basis van een ILI9488 displavdriver heeft een 16-bits kleurdiepte; de aanraakfuncties worden verzorgd door een FT6236. Beide chips worden goed ondersteund door de Arduino-community.

ESP32-S3 MCU

Als ik een blik onder de motorkap werp, blijf ik verbaasd over de ongelooflijke hoeveelheid rekenkracht die in moderne MCU's verstopt is. De ESP Terminal wordt aangedreven door een Espressif ESP32-S3 [2], die een 32-bit dual-core XTensa LX7 MCU bevat die op 240 MHz draait. Vergelijk dat eens met de originele Arduino UNO met zijn 8-bit Microchip ATMega328 op 16 MHz!

De MCU heeft 512 KB SRAM aan boord, samen met 8 MB PSRAM. PSRAM staat voor pseudo-statisch RAM, een soort extern statisch RAM dat in het geval van de ESP32-S3 via een SPI-interface is verbonden met de MCU.

Wat betreft draadloze connectiviteit hebben we 2,4 GHz WiFi (2,4 GHz 802.11 b/g/n) en Bluetooth 5 LE. Ik was me hiervan niet bewust, maar Bluetooth 5 kan een groot bereik hebben en deze module claimt dit te ondersteunen. De long-range modus breidt het bereik uit van 10...30 meter tot meer dan een kilometer.

Verbinding maken met de fysieke wereld

De ESP32 Terminal heeft een accu-aansluiting en een ingebouwde LiPo-lader. Je kunt de USB C-connector gebruiken om het apparaat van stroom te voorzien, maar ook om de LiPo-accu op te laden. Er is ook een micro-SD kaartslot, handig als je een paar afbeeldingen of andere (grafische) zaken zoals webpagina's wilt opslaan voor je applicatie.

Aan de zijkanten van de module zitten in totaal vier 'Crowtail'-poorten [3]. Dit zijn 4-draads interfaces die compatibel zijn met Grove van Seeed Studio. Er is één digitale, één analoge, één seriële (UART) en één I²C-connector, genoeg voor eenvoudige projecten. Als je meer nodig hebt, kun je de I²C-poort als breakout gebruiken.

De ESP32 Terminal programmeren

Het uploaden van firmware gebeurt via de USB-C kabel, via een seriële verbinding waarvoor een CH340 USB/serieel-converter [4] wordt gebruikt. Het is me niet gelukt om dit met mijn MacBook aan de praat



Figuur 1. De ESP32 Terminal in zijn zwarte behuizing toont hier weergegevens.



Figuur 2. De vier 'Crowtail'-poorten zijn zichtbaar aan de achterzijde. te krijgen, waarbij ik de officiële driver probeerde te installeren. Gelukkig had ik mijn oude Linux-laptop nog, waarop Ubuntu draait – en dat werkt perfect! Het eerste wat ik wilde uitproberen was een of andere vorm van Python op het apparaat te draaien. Dat bleek lastig. Hoewel de documentatie zegt dat het apparaat geprogrammeerd kan worden met Python en Micro-Python, kon ik geen kant-en-klare downloadbare firmware vinden.

Bij het doorspitten van hun website vond ik talloze voorbeelden en een tutorial voor Arduino [5], dus ben ik dat gaan gebruiken voor mijn demotoepassingen.

UI-ontwerp

Volgens de webpagina van de ESP32 Terminal is het apparaat LVGL-gecertificeerd. LVGL [6][7] staat voor Light and Versatile Graphics Library, en is een open-source embedded grafische bibliotheek voor het maken van UI's voor verschillende MCU's en displaytypes. Hoewel de bibliotheek zelf open

Figuur 3. Met een DTH-11 sensor en een beetje code bouw je een eenvoudig weerstation.



source is, bestaat er ook een closed source drag&drop layout-editor, Squareline Studio [8].

Ik heb de proefversie geïnstalleerd en hoewel het op het eerste gezicht een beetje overweldigend is, lukte het me al snel om een eenvoudige UI te maken voor het servoproject hieronder. Als het maken van UI's voor embedded apparaten je doel is, dan is het de moeite waard om wat tijd te investeren om dit product te leren kennen.

Twee kleine projecten

Ik heb besloten om twee voorbeeldprojecten te maken. Het eerste project is een klein weerstation, waarbij we onze eigen gebruikersinterface tekenen met behulp van lijn- en cirkelprimitieven. Het tweede project is een voorbeeld waarbij we een servo besturen vanuit een gebruikersinterface ontworpen met Squareline Studio.

Weerstation

Voor het weerstation gebruiken we een DHT-11 sensor die temperatuur en vochtigheid kan meten. Hij heeft een digitale één-draads interface en wordt goed ondersteund door Arduino. Voor dit project koos ik de *Adafruit DHT*-bibliotheek [9]. Deze maakt deel uit van een set sensorbibliotheken met gemeenschappelijke code en daarom moeten we ook de *Adafruit Unified Sensor Driver*-bibliotheek installeren [10].

In dit project heb ik besloten om mijn eigen UI van de grond af op te bouwen, door primitieven te gebruiken om tekst af te drukken en enkele lijngrafieken te tekenen. Hiervoor gebruikte ik de *LovyanGFX LCD and e-Ink graphics*-bibliotheek [11][12], een andere open-source grafische bibliotheek.

Na het initialiseren van het scherm, waarvoor ik gewoon een van de voorbeelden van de ESP32 Terminal website heb gekopieerd, lezen we de DHT-11 sensor uit. Deze geeft ons drie waarden: temperatuur, luchtvochtigheid en de hitte-index. De hitte-index is een temperatuur die is aangepast aan de actuele vochtigheid. Deze drie waarden worden weergegeven op eenvoudige wijzerplaten, getekend met twee cirkels en een lijn als wijzer. We verversen het scherm elke twee seconden met een nieuwe waarde.

Servocontroller

Voor de servoregelaar wilde ik de tool Squareline Studio proberen om de UI te maken. Een schuifregelaar in de vorm van een cirkelboog is de enige widget die wordt gebruikt. Het bereik van de schuifregelaar is ingesteld van 0 tot 180, wat overeenkomt met de hoek van de servo.

Als je een UI voor de ESP32 ontwerpt met deze tool, moet je uitgaan van het sjabloon Arduino with TFT_eSPI en de kleurdiepte instellen op 16 bits bij een resolutie van 480×320 pixels. Nadat je de gegenereerde code hebt geëxporteerd, moet je deze uitpakken in een map met de naam ui in de map libraries van je Arduino. Normaal gesproken vind je deze libraries-map in dezelfde map als waar je Arduino-sketches staan. Het kostte me wat tijd om dit uit te vinden.

De servo wordt aangestuurd door de ESP32Servobibliotheek [13]. De standaard Arduino servo-bibliotheek werkt niet met de ESP32.

De code is elementair: in de functie loop lezen we de hoek van de 'ronde' schuifregelaar, die een getal tussen 0 en 180 retourneert, en sturen die waarde naar de functie servo.write.

Je kunt de broncode van beide projecten vinden op de Elektor-website [14]. Ik heb de map libraries niet in de download opgenomen, omdat die (veel) te groot is. Het zelf installeren van de afhankelijkheden is niet ingewikkeld.

Veel mogelijkheden

De ESP32 Terminal van Elecrow is niet zo gemakkelijk onder de knie te krijgen als ik had gehoopt, maar hij biedt veel mogelijkheden. Als je nodige de tijd wilt investeren om de LVGL-bibliotheek of een andere UI-bibliotheek te leren, en je bent vertrouwd met Arduino, dan kun je met de ESP32 Terminal een gebruikersinterface met aanraakbediening toevoegen aan je eerstvolgende project.

Het zou gemakkelijker zijn geweest als MicroPythonfirmware als download beschikbaar was geweest op de product-website. Ik heb wat rondgezocht op internet,



maar kon niets vinden. Het is zeker mogelijk om je eigen firmware te compileren, en misschien is dat een leuk idee voor je volgende project? 🖊

230755-03

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.



WEBLINKS

- [1] Elecrow ESP32 Terminal:
- http://www.elektor.nl/esp-terminal-esp32-s3-based-development-board-with-3-5-capacitive-tft-touch-display
- [2] ESP32-S3 datasheet: https://www.espressif.com/sites/default/files/documentation/esp32-s3_datasheet_en.pdf
- [3] 'Crowtail'-poorten: https://www.elecrow.com/wiki/index.php?title=Main_Page#Crowtail
- [4] CH340 USB/serieel-converter: https://cdn.sparkfun.com/datasheets/Dev/Arduino/Other/CH340DS1.PDF
- [5] Tutorial voor Arduino:

https://www.elecrow.com/wiki/index.php?title=Lesson01_Introducing_the_ESP32_Display_series_and_environment_configuration [6] LVGL: https://lvgl.io/

- [7] LVGL-documentatie: https://docs.lvgl.io/latest/en/html/index.html
- [8] Squareline Studio: https://squareline.io/
- [9] DHT-11 bibliotheek: https://github.com/adafruit/DHT-sensor-library
- [10] Unified Sensor bibliotheek: https://github.com/adafruit/Adafruit_Sensor
- [11] LovyanGFX LCD and e-Ink graphics bibliotheek: https://github.com/ropg/LovyanGFX
- [12] LovyanGFX-documentatie: https://lovyangfx.readthedocs.io/en/latest/index.html
- [13] ESP32 servo bibliotheek: https://www.arduino.cc/reference/en/libraries/esp32servo/
- [14] Software-download bij dit artikel: https://www.elektormagazine.com/news/elecrow-esp32-terminal-review
- [15] Günter Spanner, "MicroPython for the ESP32 and Friends (Part 1): Our First Programs," 2021: https://www.elektormagazine.com/articles/micropython-esp32-microcontroller



Figuur 4. De ronde schuifregelaar regelt de stand van de servo.

Aan de slag met Zephyr RTOS

krachtig - maar lastig...

Clemens Valens (Elektor)

Zephyr is a klein, schaalbaar real-time operating systeem (RTOS), ontworpen voor apparaten met relatief weinig resources en voor verschillende architecturen. Gesteund door de Linux Foundation (zie [1]) is Zephyr een open source-samenwerkingsproject met als doel het beste RTOS in zijn klasse te ontwikkelen. Het Zephyr RTOS is de laatste jaren steeds meer onder de aandacht gekomen voor embedded systemen en steeds meer microcontrollers en ontwikkelingsboards worden ondersteund door Zephyr. Hoogste tijd dus om er een nadere blik op te werpen.

Zephyr is schaalbaar, en daardoor bruikbaar voor een breed scala aan microcontrollers met al hun beperkingen op het gebied van resources. Deze schaalbaarheid wordt bereikt door een modulaire architectuur waardoor ontwikkelaars alleen die onderdelen die ze nodig hebben hoeven toe te voegen om de grootte van het systeem te beperken. De Zephyr-website stelt dat het systeem kan werken op systemen met slechts 8 KB geheugen, maar schaalbaar is tot geheugens van meerdere Gigabytes.

Brede hardware-ondersteuning

Zephyr ondersteunt een breed scala architecturen zoals ARM, x86, RISC-V, Xtensa, MIPS en meer. Ook FPGA's worden ondersteund met de Nios2- en MicroBlaze-softcores. Op het moment van schrijven kent Zephyr meer dan 600 bruikbare boards waaronder Arduino UNO R4 Minima, GIGA R1 WIFI en Portenta H7, diverse uitvoeringen van de ESP32, beide versies van de BBC micro:bit, Raspberry Pi Pico (en zelfs Raspberry Pi 4B+), nRF51- en nRF52-boards, de NXP MIMXRT1010-EVK familie en de STM32 Nucleo- en Discoveryfamilies. En dat zijn alleen nog de bekendste namen die regelmatig in Elektor voorbijkomen, er zijn er nog een heleboel meer. Naast processorboards ondersteunt Zephyr ook veel uitbreidingsboards (shields) en zijn er drivers beschikbaar voor allerhande interfaces en meer dan 150 sensoren.

Multitasking, netwerken en energiebeheer

Een real-time operating system (RTOS) zoals Zephyr biedt mogelijkheden zoals preemptive multitasking, inter-thread communicatie en real-time klok-ondersteuning. Het OS heeft verder ondersteuning voor netwerktechnologieën en -protocollen zoals TCP/IP, Bluetooth en IEEE 802.15.4 (zoals gebruikt in Zigbee), MQTT, NFS en LoRaWAN. Deze netwerktechnologieën, samen met krachtige power management-technieken in Zephyr, maken het geschikt voor energiezuinige IoT-toepassingen en batterijgevoede apparatuur. De beschikbare bibliotheken en middleware maken het eenvoudig om veelvoorkomende functies zoals communicatieprotocollen, bestandssystemen en apparaatdrivers te implementeren. Daarnaast is Zephyr ontwikkeld in overeenstemming met met beveiligingscertificaten zoals ISO 26262, waardoor het ook toepasbaar is in toepassingen die een goede beveiliging vereisen.

Geïnspireerd door Linux

Zephyr is geen Linux, maar het maakt wel gebruik van concepten, technieken en tools die ook in Linux toegepast worden. Kconfig wordt bijvoorbeeld gebruikt om het OS te configureren, en hardwareeigenschappen en -configuraties worden vastgelegd volgens de Device Tree Specificatie (DTS) [2]. Daarom zullen Linux-ontwikkelaars zich snel thuis voelen als ze met Zephyr aan de slag gaan.

Open-source

Last but not least is Zephyr vrijgegeven onder de Apache 2.0-licentie die zowel commerciële als niet-commerciële toepassingen toelaat. De gebruikersgemeenschap biedt ondersteuning en documentatie. Je kunt daar ook aan bijdragen.

Zephyr uitproberen

Zephyr uitproberen heeft meerdere jaren op mijn to-do lijst gestaan, maar mijn eerste ervaringen waren niet erg bemoedigend waardoor ik toen niet verder gegaan ben. Destijds was een van de grote problemen (naast het OS überhaupt zonder foutmeldingen te kunnen compileren) dat er een programmeermodule nodig was om de





Figuur 1. Het compacte BBC micro:bit board is een prima kandidaat om Zephyr RTOS te testen. Wie had gedacht dat dit kleine board, oorspronkelijk bedoeld om tienjarigen te leren programmeren met MakeCode, een Scratch-achtige grafische programmeertaal, ook een prima hulpmiddel zou zijn voor ervaren embedded software-ontwikkelaars die een professioneel, real-time operating system willen leren kennen?

targetprocessor te programmeren, waardoor het minder toepasbaar was voor hobbyisten en semi-professionals. Dankzij Arduino en zijn bootloader zijn we gewend geraakt aan het werken zonder speciale programmeerhardware waardoor deze eis aanvoelde als een flinke stap terug.

Keuze van een board

De ontwikkelingen hebben sinds die tijd niet stilgestaan. Zoals al opgemerkt ondersteunt Zephyr inmiddels meer dan 600 microcontroller boards en er is dus een goede kans dat je al een of meer compatibele boards beschikbaar hebt. Bij het bestuderen van de lijst ontdekte ik dat ik ruim een dozijn verschillende bruikbare boards had liggen.

Leve de BBC micro:bit!

Ik heb de meeste ervan geprobeerd en heb uiteindelijk de BBC micro:bit gekozen voor mijn experimenten (**figuur 1**, in Zephyr bekend als bbc_microbit of bbc_microbit_v2, afhankelijk van de versie van het board). In vergelijking met andere boards heeft de BBC micro:bit, naast dat ik er een had liggen, het voordeel dat het misschien wel de beste Zephyr-ondersteuning heeft, wat betekent dat drivers voor alle interfaces op het board beschikbaar

zijn en ondersteund worden met voorbeelden. En misschien wel het belangrijkst: programmeren en debuggen kan zonder extra hardware of software.

De populaire ESP-WROOM-32 (in Zephyr bekend als esp32_ devkitc_wroom) is ook een geschikte kandidaat, maar debuggen vereist wel een extern tool.

Arduino GIGA R1 WIFI zou ook een goede keuze zijn maar daarvan wordt de standaard bootloader overschreven als Zephyr gebruikt wordt. Dat kan weliswaar hersteld worden, maar is wat mij betreft een ongewenst bijverschijnsel.

Arduino UNO R4 Minima heeft (net als veel andere boards, waaronder de Raspberry Pi Pico), officieel een SWD-programmeermodule nodig, maar ik vond een workaround door *dfu-util* te gebruiken (zie verderop). Net als bij de GIGA R1 wordt ook hier de Arduino bootloader overschreven door Zephyr.

...of gebruik een emulator

Als je geen bruikbaar board bij de hand hebt en je toch graag Zephyr wilt proberen, dan is de ingebouwde emulator-ondersteuning voor QEMU een mogelijkheid (alleen beschikbaar op Linux/macOS). Hiermee kun je toepassingen virtueel uitvoeren en testen. Renode van Antmicro werkt vergelijkbaar, maar ik heb dat niet uitgeprobeerd.

Zephyr Installeren

Ik heb Zephyr geïnstalleerd op een computer met Windows 11 – ik heb Linux en macOS niet geprobeerd. Uitgebreide instructies voor de installatie zijn online beschikbaar op [3]. De benodigde stappen worden duidelijk aangegeven en verdere uitleg is niet nodig. Ik gebruikte een virtuele Python-omgeving zoals aanbevolen. Dat betekent wel dat je het commando om de virtuele omgeving te activeren zult moeten noteren omdat je het elke keer dat je de software wilt gebruiken nodig hebt. Bij gebruik van Windows PowerShell voer je het script *activate.ps1* uit; onder Command Prompt is dit het *activate.bat* batchbestand. Windows PowerShell is beter in het verwerken van compiler- en linker-uitvoer (**figuur 2**).



Figuur 2. Het Zephyr build-tool west is bedoeld om in een terminal te gebruiken. cmd.exe van Windows kan gebruikt worden maar is geen echte terminal. Windows PowerShell, ook bekend als Terminal, is daarvoor beter geschikt.



	🔿 🛛 Administrator: Windows Powe × + ~	- 0	×
USB	Merged configuration 'D:/dev/zephyr/zephyrproject/zephyr/samples/hello_world/prj.conf' Configuration_saved to 'D:/dev/zephyr/zephyr/samples/hello_world/prj.conf'		
	Kconfig header saved to 'D:/dev/zephyr/ McCOM11-TeraTerm VT -) ×	
	Found GnuLd: c:/users/enerv/zephyr-s File Edit Setup Control Window Help		m-z
EDU	ephyr-eabi/bin/ld.bfd.exe (found versio*** Booting Zephyr OS build zephyr-v3.5.0-1392-ga3ff19a39eb1 ***		
· 1 ·	The C compiler identification is GNUHello World! bbc_microbit		
Juna	The CXX compiler identification is G		
	The ASM compiler identification is G		
	Configuring done (29 Ss)		
	Generating done (0.25)		
	Build files have been written to: D:		
	←[92m west build: building applicatio		
	[1/132] Generating include/generated/ve		
	Zephyr version: 3.5.99 (D:/dev/zephy		
	[132/132] Linking C executable zephyr.elf		
	Memory region Used Size Region Size % age Used		
	PLASH. 2320 D 230 KD 0.07% DAM- 5///18 P 16 //P 22 25%		
SEGGER			
www.segger.com	(.venv) PS D:\dev\zephyr\zephyrproject\zephyr> west flash		
	west flash: rebuilding		
Target	ninja: no work to do.		
-	west flash: using runner pyocd		
	runners.pyocd: Flashing file: D:\dev\zephyr\zephyrproject\zephyr\build\zephyr\zephyr.hex		
	00011522 I Loading D:\dev\zephyr\zephyrpoject\zephyr\build\zephyr\zephyr\setuild\zephyr\zephyr.hex [load_cmd]		
	[0.03899] T Erzsed 23552 bytes (23 sectors) programmed 23552 bytes (23 pages) skipped 0 bytes (0 pages) at 10	<u>АЦ 68/</u>	: Г1
uur 3. In veel (maar niet alle)	oader]	04 KD/S	
	(venu) PS D:\dev\zenhvr\zenhvrproiect\zenhvr>		

situaties is een JTAG- or SWDprogrammer/debugger noodzakelijk voor Zephyr.

Figuur 4. Het Hello World-voorbeeld is niet erg uitgebreid. Als je de seriële terminal te laat opent, zul je de uitvoer misschien zelfs missen.

Zephyr bestaat uit twee delen – het OS zelf en een SDK (Software Development Kit) met daarin een verzameling MCU-toolchains (21 op het moment van schrijven). Het OS en de SDK hoeven niet op dezelfde locatie geïnstalleerd te zijn. In totaal nam het pakket bij mij ongeveer 12 GB ruimte in beslag op de harde schijf. Door toolchains die je niet gebruikt te verwijderen kunt je wat ruimte vrijmaken. Na de installatie kun je proberen of alles werkt door een van de voorbeelden te compileren en dit naar je board over te brengen met het onderstaande commando. Vervang daarbij <my_board> door de naam van het board dat je gebruikt, bijvoorbeeld arduino_uno_r4_minima:

west build -p always -b <my_board> samples/basic/blinky

Als je het pad naar het voorbeeld niet wilt aanpassen, moet je het commando uitvoeren in de map waar het staat (waarbij (.venv) aangeeft dat je in een virtuele omgeving werkt):

(.venv) <my_path_to_zephyr>\zephyrproject\zephyr

Als het voorbeeld zonder foutmeldingen compileert, kun je het naar je board overbrengen met:

west flash

waarna de 'default' LED op je board begint te knipperen met een frequentie van 0,5 Hz.

Zoals eerder opgemerkt heb je voor het flashen mogelijk een externe programmeermodule zoals een J-Link-adapter of een vergelijkbare (JTAG- of SWD-compatibele) programmer nodig (**figuur 3**) waarbij de bijbehorende driversoftware toegankelijk moet zijn (dat wil zeggen in het zoekpad – zoals %PATH% in Windows). Als dit niet het geval is, zul je een foutmelding krijgen (maar die zijn vaak lang en nogal cryptisch).

Op de BBC micro:bit V2 moest ik de eerste keer het HEX-bestand met de hand naar het board kopiëren volgens de standaard micro:bit-procedure. Daarna werkte de flash-methode prima. Het bestand zephyr.hex is te vinden in zephyrproject\zephyr\build\zephyr\.

Het standaard flash-commando voor de Arduino-boards UNO R4 Minima en GIGA R1 WIFI vereist dat het *dfu-util* hulpprogramma te vinden is in het zoekpad van het OS (voordat je de virtuele omgeving start, als je die gebruikt). Dit hulpprogramma is te vinden in de Arduino IDE, maar waar dit precies op je computer staat, zul je zelf moeten uitzoeken. (Standaard staat het in %HOMEPATH%\ AppData\Local\Arduino15\packages\arduino\tools\dfu-util\<je meest recente Arduino versie>). Het board moet daarna eerst in DFU-modus gezet worden door twee keer snel na elkaar op de resetknop te drukken. Als de LED begint te 'pulseren' of te 'ademen' kan het programma geflasht worden.

Blinky-compatibiliteit

Ik heb in het eerste voorbeeld de Arduino UNO R4 Minima als board gebruikt om Blinky te testen en niet de BBC micro:bit waar ik eerder zo enthousiast over was. De reden daarvoor is dat, ondanks dat er 25 LED's (de aan/uit-LED niet meegeteld) beschikbaar zijn, er geen LED op zit die compatibel is met het Blinky-voorbeeld. De ESP Wroom 32 heeft die ook niet, maar de R4 Minima wel.

De GIGA R1 werkt ook met het Blinky-voorbeeld. De MCU op dit board heeft twee rekenkernen (Cortex-M7 en -M4) en Zephyr laat je de keuze welke te gebruiken door ofwel *arduino_giga_r1_m4* ofwel *arduino_giga_r1_m7* te kiezen voor het build-commando. Je kunt bewijzen dat de kernen inderdaad onafhankelijk werken door het Blinky-voorbeeld twee keer te laden, een keer voor de -M4 en een keer voor de -M7. De GIGA heeft een RGB-LED en Blinky zal op elke kern een andere kleur gebruiken: blauw op de M4 en rood op de M7. Om nog duidelijker onderscheid tussen beide Blinkies te maken kun je de knippersnelheid in één van de twee aanpassen (verander in *samples\basic\blinky\src\main.c* de waarde van SLEEP_TIME_MS).

Hello World

Voor boards zonder Blinky-LED is er een *hello_world* voorbeeld dat een welkomsttekst als uitvoer produceert op de seriële poort.

west build -p always -b <my_board> samples/hello_world west flash

Dit voorbeeld werkt op zowel de BBC micro:bit als op de ESP-WROOM-32 modules. Om de uitvoer te zien, open je een seriële monitor op je computer. De gebruikelijke baudrate is 115.200 baud (115200,n,8,1). Mogelijk moet je het board na het verbinden resetten omdat de tekst slechts eenmalig geprint wordt en je het mogelijk gemist hebt (**figuur 4**).

Op de R4 Minima en de GIGA R1 verschijnt de seriële uitvoer op pin 1 en niet, zoals je in je onschuld zou verwachten op de USB-C poort zoals dat in de Arduino IDE het geval is. Dat komt omdat de USB hier een interface van de MCU is en geen externe chip; omdat Zephyr modulair en schaalbaar is moet USB-ondersteuning – net als voor elke interface – expliciet ingeschakeld worden voor een project voordat deze gebruikt kan worden. Dit gebeurt in de configuratiebestanden van het project. Verderop meer hierover.

Bij boards met een ingebouwde serieel/USB-converter zul je eerst de juiste seriële poort moeten vinden (meestal is dit poort o als de MCU er meerdere heeft) en deze via een externe serieel/USB-converter met uw computer moeten verbinden.

lets moeilijker

Al het je gelukt is om *Blinky* en *hello_world* aan de praat te krijgen op je board, heb je een goede uitgangspositie om echte toepassingen te gaan ontwikkelen in Zephyr. Als slechts één van de voorbeelden werkt en je het andere ook aan de praat wilt krijgen, kan het iets ingewikkelder worden.

De BBC micro:bit was mijn eerste keus voor mijn Zephyr-experimenten, ondanks het feit dat het Blinky-voorbeeld hier niet werkt. Maar dat is niet zo erg omdat voor het board ook een paar bruikbare voorbeelden beschikbaar zijn (in de *bbc_microbit-submap* van de *samples\boards* map) waarvan er eentje (*display*) veel interessanter is dan de enkele knipperende LED van Blinky. Er zijn ook voorbeelden voor andere boards, maar slechts weinig in verhouding met het totaal aantal ondersteunde boards (minder dan 5%). Bovendien betreffen veel van deze voorbeelden nogal geavanceerde of ongebruikelijke toepassingen.

Als je Blinky voor de BBC micro:bit (of de ESP-WROOM-32 of een ander incompatibel board) wilt compileren, krijg je een lastig te begrijpen foutmelding voorgeschoteld. Deze probeert je te vertellen dat ledo een onbekende entiteit is. ledo is de standaard Blinky LED (vergelijkbaar met LED_BUILTIN bij Arduino). Omdat de micro:bit een uitbreidingspoort heeft waar je onder andere LED's op kunt aansluiten, gaan we een van de pinnen van de poort te definiëren als ledo.

Maar voordat we dat doen, maken we eerst een backup van de *samples/basic/blinky* map of, beter nog, maken we een map met een nieuwe naam aan en gebruiken deze. Hieronder gebruiken we *samples/basic/blinky*, maar je moet natuurlijk de naam gebruiken van de nieuwe map.

De device tree

Het definiëren van een ledo brengt ons bij de device tree, die we al eerder kort hebben genoemd. De device tree bestaat uit een of meer tekstbestanden en beschrijft onder andere de interfaces en het beschikbare geheugen op een board of controller. In Zephyr hebben deze bestanden een .dts- of .dtsi-extensie (de 'i' staat hier voor 'include') en deze bestanden mogen zelf weer andere bestanden invoegen. Processor-gerelateerde .dtsi-bestanden staan in de dts-map, board gerelateerde .dtsi-bestanden in de boards-map. Beide mappen zijn weer onderverdeeld naar de processor-architectuur.

Om DTS(I)-bestanden comfortabel te lezen, kun je de Device Tree plug-in voor Visual Studio Code [4] gebruiken. Deze biedt syntax highlighting en code folding waardoor de bestanden leesbaarder



Figuur 5. Een gedeelte van het nrf51822.dtsi-bestand. De miniweergave rechts geeft een indruk van de omvang van het hele bestand.



Figuur 6. Dit deel van de device tree laat onder andere de l²C-busdefinitie van het BBC micro:bit board zien, niet van de processor. Dit is een deel van het bbc_microbit.dts-bestand.

zijn (DTS bestanden gebruiken een C-achtige syntax). **Figuur 5** toont een deel van het .*dtsi*-bestand voor de nRF51822 die het hart van het BBC micro:bit V1-board vormt. Dit bestand hoort bij de DTS-bibliotheek van het board. Een voorbeeld: de status van uart0 staat op "disabled". Deze status wordt overschreven in het DTS-bestand van het board, waar deze "okay" is, wat betekent dat uart0 te gebruiken is. Hetzelfde geldt voor gpio0 en i2c0.

I²C in de device tree

Een ander fragment van het .*dts*-bestand voor de BBC micro:bit is te zien in **figuur 6**. Het toont de device tree voor de I²C-bus. Er zijn een of twee sensoren op de bus aangesloten bij de micro:bit (afhankelijk van de variant van de micro:bit V1) en ze zijn in de tree gedefinieerd als mma8653fc en lsm303agr (de laatste bevat twee sensoren, te zien omdat hij twee keer in de tree voorkomt). De eerste heeft status "okay", terwijl de andere twee op "disabled" staan. Dit klopt voor mijn board omdat dat er eentje is uit de eerste micro:bit V1-generatie.

Zoals het fragment laat zien, is deze sensor compatibel met de FXOS8700 en de MMA8653FC, het adres op de I²C-bus is 0x1d, en er zijn twee int (interrupt-)signalen gedefinieerd die verbonden



M	COM	11 - Tera	Term VT					-	×
File	Edit	Setup	Control	Window	Help				
	00000000000000000000000000000000000000	12222222222222222222222222222222222222	20000110000000000000000000000000000000	398111 3577502111 35775022111 35068450661 3568702111 3568702111 3568702111 3568702111 3568702111 368702111 3788 3776531 3788702111 3788702111 3788702111 3788702111 3788702111 3788702111 3788702111 3788702111 378770211 378770211 378770211 378770211 3787702000000000000000000000000000000000	<u>\$</u> \$ \$\$\$\$\$\$\$\$\$\$\$		<pre>temp_nrf5_channel_get: Temperature:21,0 temp_nrf5_sample_fetch: sample: 84 temp_nrf5_channel_get: Temperature:21,0 temp_nrf5_sample_fetch: sample: 84 temp_nrf5_channel_get: Temperature:21,0 temp_nrf5_sample_fetch: sample: 84 temp_nrf5_channel_get: Temperature:21,0 temp_nrf5_channel_get: Temperature:21,0 temp_nrf5_channel_get: Temperature:21,0 temp_nrf5_channel_get: Temperature:21,0 temp_nrf5_channel_get: Temperature:21,0 temp_nrf5_channel_get: Temperature:21,0 temp_nrf5_sample_fetch: sample: 84 temp_nrf5_channel_get: Temperature:21,0 temp_nrf5_sample_fetch: sample: 84 temp_nrf5_channel_get: Temperature:21,0 temp_nrf5_sample_fetch: sample: 84 temp_nrf5_channel_get: Temperature:21,0 temp_nrf5_channel_get: Temperature:21,0 temp_nrf5_channel_get: Temperature:21,0 temp_nrf5_sample_fetch: sample: 84 temp_nrf5_s</pre>		
Ĕĕ	0:0 0:0	2:40 2:40	.399 .399	047] 200]	<qpa <qpa <qpa <qpa ></qpa </qpa </qpa </qpa 	temp_nrf5: temp_nrf5:	temp_nrf5_sample_fetch: sample: 84 temp_nrf5_schanneI_get: Temperature:21,0		

Figuur 7. De uitvoer van de samples/sensor/fxos8700-demo, uitgevoerd op de BBC micro:bit.

zijn met GPIO pinnen 27 en 28. Als je het wilt testen is er een demoprogramma beschikbaar:

west build -p always -b bbc_microbit samples/sensor/ fxos8700 west flash

Let op dat dit niet opgaat voor de BBC micro:bit V2 omdat deze een ander type sensor definieert in de device tree.

De uitvoer van dit demoprogramma is te zien in figuur 7 – maar laten we niet afdwalen.

Overlay van de device tree

Terug naar onze LED, led0. In de device tree van ons board komt zoals verwacht geen led0 voor, dus die moeten we toevoegen. We zouden het direct in de device tree van het board kunnen schrijven, maar dat zou niet correct zijn omdat het board geen led0 heeft. De correcte manier om een device tree uit te breiden is het maken van een overlay-bestand. De inhoud van zo'n overlay-bestand wordt aan de device tree toegevoegd. Secties die al in de device tree aanwezig zijn worden uitgebreid (in het geval van een nieuw item) of overschreven (als het item al bestaat in de device tree); nieuwe secties worden toegevoegd.

Overlay-bestanden moeten in de projectmap staan, in een submap genaamd *boards*. Als deze map bestaat, wordt door het build-proces gezocht naar een bestand met de naam <*my_board*>.overlay. In mijn geval is de bestandsnaam *bbc_microbit.overlay* (*bbc_microbit_v2.overlay* voor gebruikers van versie 2). **Figuur 8** toont de inhoud van dit bestand.

Toevoegen van de Blinky-LED

Zephyr heeft een standaard device tree-item gereserveerd voor LED's: leds, dus hebben we daar een node (tak) voor gemaakt (je kunt elke naam kiezen die je wilt, maar uiteraard kies je leds als het een overlay voor de bestaande leds-node is). Deze node wordt toegevoegd aan de root (het hoogste niveau) van de device tree; daarom begint de naam met een schuine streep: '/', dat geeft de root aan in DT-terminologie. De volgende regel geeft aan dat deze node compatibel is met de gpio-leds driver die is ingebouwd in Zephyr (de interface voor deze driver is te vinden in zephyr\include\zephyr\drivers\led.h).

Onderliggende nodes

Nu volgt een lijst van onderliggende LED-nodes (child nodes). Omdat ik maar één LED wil aansturen is er ook maar één onderliggende node nodig, led_0, met als naam led0. Een naam geven aan een node is niet verplicht, maar maakt het mogelijk er elders in de tree naar te verwijzen, iets dat we een paar regels verderop zullen doen. Daarnaast kan de naam gebruikt worden door de ontwikkelaar om te verwijzen naar een node en onderdelen daarvan.

In een child node worden de eigenschappen van een device gedefinieerd. In het geval van de een LED is alleen de bijbehorende GPIO-pin verplicht, optioneel kan nog een label (naam) toegekend worden. Labels worden gebruikt als documentatie: voor mensen leesbare informatie in de toepassing, labels hebben verder geen functie.

Ik kies hier GPIO-pin 1, die verwijst naar het grote pad nummer 2 van de micro:bit-uitbreidingsconnector. Bij gebruik van de BBC micro:bit V2 kies je hier 4 als GPIO-pin (in plaats van 1).

Een alias maken

De volgende stap is noodzakelijk omdat het Blinky-voorbeeld het gebruikt. Het betreft het aanmaken van het led0-alias voor onze LED. Je zou verwachten dat toekennen van het gelijknamige label aan de child node volstaat, maar dat is niet zo omdat Blinky de DT_ALIAS macro gebruikt om de LED child node te benaderen. Daarom moeten we iets hebben dat deze macro kan verwerken



Figuur 8. Dit device tree overlay-bestand maakt de BBC micro:bit V1 geschikt voor het Blinky-voorbeeldprogramma.

en dat is een alias, en die komt in het aliases blok. Als Blinky de DT_NODELABEL macro gebruikt zou hebben, zou een alias overbodig zijn omdat DT_NODELABEL het led0 child node-label direct kan gebruiken. Ik begrijp dat het bestaan van labels en aliassen met dezelfde naam verwarrend kan zijn, maar dat is voor de uitleg van dit voorbeeld noodzakelijk.

Zephyr-macro's

Macro's zijn niet populair in C/C++, maar in Zephyr worden ze veel gebruikt. Macro's zoals DT_ALIAS en DT_NODELABEL laten het de applicatie- en project-configuratie-tools toe om informatie uit de device tree te halen en worden daarom veel toegepast. Je kunt de beschrijving ervan vinden in het ephyr-handboek, in het hoofd-stuk "Devicetree API" [5].

Aardig om te weten is dat veel (of misschien wel alle?) Zephyrmacro's verwachten dat de argumenten met kleine letters zijn geschreven, waarbij alle karakters die geen letters ('a' tot 'z') of cijfers ('o' tot '9') zijn, worden vervangen door underscores ('_). Dit wordt lowercase-en-underscore-compatibel genoemd. Als ik de eerder beschreven LED child node LED-0 had genoemd in plaats van led0 zou het argument voor de DT_NODELABEL-macro led_0 moeten zijn: DT_NODELABEL(led_0) omdat '-' geen letter of cijfer is, en alleen kleine letters zijn toegestaan. Met andere woorden: voor de ontwikkelaar die device tree-macro's gebruikt, geldt de underscore als wildcard ('joker'). Daarom kan led_0 in een toepassing verwijzen naar led_0, led-0, Led_0, LED-0 en ledé0 (en elke andere denkbare variant) in de device tree. Met dit in gedachten is het aanbevolen om de documentatie van Zephyr-macro's zorgvuldig te bestuderen.

Gij zult geen fouten maken

Bedenk dat fouten in de device tree tot gevolg hebben dat de compiler 'FATAL ERROR' zal melden zonder enige nadere tekst of uitleg!

'Nieuwbouw'

Bij het experimenteren met de device tree of met je toepassing zul je het project waarschijnlijk vaak compileren. Om dat proces te versnellen kun je -p always (de 'p' staat voor 'pristine' wat zoveel als 'ongerept' betekent) weglaten uit het build-commando. Hierdoor wordt niet alles van de grond af opnieuw gecompileerd. Anderzijds, als je verschillende voorbeelden na elkaar wilt testen, kun je de optie beter laten staan omdat er anders voortdurend meldingen verschijnen dat de build-folder niet bij het gekozen project hoort. Het flash-commando voert automatisch eerst het laatst gebruikte build-commando uit waardoor het volstaat om direct het flashcommando te gebruiken na een wijziging.

Gebruik een device driver

Het Blinky-voorbeeld roept gpio_pin_toggle_dt() aan om de toestand van de LED te wijzigen. Dit is logischerwijze een functie van de GPIO-driver. Dat is helemaal in orde, maar Zephyr heeft ook een verzameling specifieke LED-drivers. Een LED-driver maakt het programma niet alleen beter leesbaar, maar ook flexibeler en beter uitwisselbaar omdat de LED-driver kan worden aangepast zonder wijzigingen aan de toepassing. Hier komen de schaalbaarheid en de modulariteit van Zephyr tot uiting.



Figuur 9. De GUI van het Kconfig project-configuratietool. Er zijn talloze opties.

Kconfig heeft een GUI

Het integreren van de LED-driver in ons programma omvat meerdere stappen. Als eerste moet het project zodanig geconfigureerd worden dat de driver meegenomen wordt. Hiervoor gebruiken we Kconfig, het kernel build-time configuratiesysteem dat ook in Linux gebruikt wordt. Er zijn meerdere manieren om er mee te werken, waarvan een grafische gebruikersinterface (GUI) er een is. In Zephyr start je deze als volgt:

west build -t guiconfig

Het duurt even voor de GUI gestart is, maar als het eenmaal zover is zie je iets als in **figuur 9**. Kconfig laat veel informatie zien over het project. Let op het project-under-development gedeelte: om er zeker van te zijn dat Kconfig daadwerkelijk je project bewerkt, voer je éénmalig een pristine build uit (met de -p always vlag) voordat je de Kconfig GUI start.

Zo veel configuratie-opties...

Neem de tijd om de configuratiemogelijkheden te onderzoeken. Vouw vertakkingen open door op de + te klikken. Selecteer opties door erop te klikken, dan verschijnt onderin het venster aanvullende informatie. Merk op dat floating-point ondersteuning voor printf() een configuratie-optie is, net als C++ ondersteuning. Op vergelijkbare wijze kun je onder *Build and Link Features* allerlei optimalisatie-instellingen voor de compiler vinden.

Er zijn talloze configuratiemogelijkheden maar die waarin we hier geïnteresseerd zijn, is het *Device Drivers*-gedeelte. Vouw dit open en scrol omlaag om een indruk te krijgen van wat allemaal mogelijk is. De LED-driver staat ongeveer halverwege: *Light-Emitting Diode* (*LED*) *Drivers*. Selecteer de optie en laat de keuzes die erbij horen ongemoeid (**figuur 10**). Klik op *Save* en onthoud het pad naar het

Save	Save as	Save minimal (advanced)	Open	Jump to			
Show name	Show all	Single-menu mode					
Fop) -> Device I	Drivers			Name			
puon				INdrife			
	drivers			GNSS			
I KGenera	al-Purpose In	put/Output (GPIO) drivers		GPIO			
Hardw	are Informati	on drivers		HWINFO			
HW sp	inlock Suppo	rt		HWSPINLOC	K		
🗄 🗰 İnter-lı	ntegrated Cire	cuit (I2C) bus drivers		12C			
Inter-I	C Sound (I2S)	bus drivers		125			
Improv	ved Inter-Inte	grated Circuit (I3C) bus drivers		I3C			
Interrupt	controller driv	/ers					
Inter-P	rocessor Mai	lbox (IPM) drivers		IPM			
Keybo	ard scan drive	ers		KSCAN			
E KLight-I	Emitting Dioc	le (LED) drivers		LED			
% (90)) LED initializ	ation priority (NEW)		LED_INIT_PR	IORITY		
_ 🔀 GP	10 LED driver	(NEW)		LED_GPIO			
Light-l	Emitting Diod	le (LED) strip drivers		LED_STRIP			
	rivers [EXPER	IMENTAL]		LORA			
Multi-	Channel Inter	-Processor Mailbox (MBOX) dri	vers	MBOX			
Manag	gement Data I	nput/Output (MDIO) drivers		MDIO			
Memo	ry controller	drivers [EXPERIMENTAL]		MEMC			

Include LED drivers in the system configuration.

Kconfig definition, with parent deps. propagated to 'depends on'

At drivers/led/Kconfig:6

Included via D:/dev/zephyr/zephyrproject/zephyr/Kconfig:8 -> Kconfig.zephyr:49 > drivers/Kconfig:48 Menu path: (Top) -> Device Drivers

Modified

Figuur 10. Selecteer Light-Emitting Diode (LED) Drivers en laat de onderliggende instellingen ongemoeid.

```
#include <zephyr/kernel.h>
#include <zephyr/device.h>
#include <zephyr/drivers/led.h>
#define SLEEP_TIME_MS (1000) /* 1000 msec = 1 sec */
int main(void)
    printf("\nBlinky with LED device driver\n");
    const struct device *const led_device = DEVICE_DT_GET_ANY(gpio_leds);
if (!led_device)
         printf("No device with compatible 'gpio-leds' found\n");
    else if (!device_is_ready(led_device))
         printf("LED device %s not ready\n", led_device->name);
         return 0;
    while (1)
             result = led on (led device, 0);
         if (result<0)
             printf("led_on failed\n");
return 0;
         k msleep(SLEEP TIME MS);
         result = led_off(led_device,0);
if (result<0)</pre>
         {
             printf("led_off failed\n");
         k msleep(SLEEP TIME MS);
    3
    return 0;
```

Figuur 11. Het Blinky-programma aangepast voor het gebruik van een LED device driver. Er is geen board-afhankelijke code waardoor het programma zou moeten werken op elk board met een gpio_leds (or gpio-leds) driver in de device tree.

configuratiebestand dat onderin het venster staat. Het is leerzaam om dit bestand eens te bekijken, al is het maar om te zien dat er heel veel informatie in staat. Sluit vervolgens de GUI af.

Vanaf nu mag je de -p always vlag niet meer gebruiken in het build-commando omdat dit de zojuist gedane wijzigingen ongedaan zal maken. Verderop zal ik je verklappen uit hoe je deze wijzigingen permanent kunt maken.

Blinky met de LED device driver

Nu kunnen we het nieuwe Blinky-programma schrijven – zie **figuur 11**. Het start met het invoegen (*#include*) van de device en de LED driver files. Dan, in het hoofdprogramma (main), passen we de **DEVICE_DT_GET_ANY** macro toe om uit de device tree een verwijzing naar de LED op te halen. Merk op dat het gpio_leds argument voldoet aan de lowercase- en underscore-regel zodat deze overeenkomt met de gpio-leds waarde van de compatible eigenschap in de leds-node van de device tree (zoals hiervoor uitgelegd). Als geen overeenkomst gevonden wordt, bijvoorbeeld door een type- of andere fout, zal de nieuwe Blinky de melding "No device with compatible gpio-leds found" weergeven. Deze melding zal ook verschijnen als de status van een device nog op "disabled" staat. Het gebruik van het woord 'compatible' als zelfstandig naamwoord in Zephyr is even wennen. De melding van hiervoor hoeft niet te betekenen dat er geen compatibel device bestaat, maar dat er geen device is met een eigenschap die compatible heet en met de waarde gpio-leds (of bijvoorbeeld gpio_leds: de underscore ' 'vervangt nu eenmaal alles behalve 'a' tot 'z' en '0' tot '9').

Een tweede controle verifieert of het device correct geïnitialiseerd wordt. Er van uitgaande dat dit het geval is, gaan we verder. In de eindeloze while()-lus wordt de LED in- en uitgeschakeld door middel van de led on en led off commando's die de driver [6] kent. De parameter o duidt het eerste (en hier enige) device dat door de macro DEVICE_DT_GET_ANY gevonden is, hier led0.

Controleer de geretourneerde waarden

Omdat we een device driver gebruiken in plaats van een GPIO-pin via hardware-registers direct aan te sturen, is het een goede gewoonte om de geretourneerde waarden van de driverfunctie-aanroepen te controleren omdat er redenen kunnen zijn waardoor deze een foutmelding geven. Een driver moet verplicht bepaalde functies en callbacks aanbieden maar er zijn ook optionele functies mogelijk. Een LED-driver moet bijvoorbeeld led_on en led_off hebben, maar led_blink is optioneel. Als je project led_blink aanroept zal er niets gebeuren omdat deze functie niet geïmplementeerd is. De functie bestaat, maar is leeg en de retourwaarde zal dit melden. Het is daarom een goede gewoonte om waar mogelijk de retourwaarde van een aangeroepen functie te controleren in je software. Bouw en laad het programma als volgt (zonder de –p always vlag):

west build -b bbc_microbit samples/basic/blinky west flash

Configureren van het project

Als de LED gaat knipperen met een frequentie van 0,5 Hz hebben we een werkend programma. Om dat zo te houden moeten we de huidige, aangepaste configuratie permanent maken.



Figuur 12. De BBC micro:bit ondersteunt gdb-debuggen zonder aanpassingen, er is geen extra hardof software nodig.

Hiertoe open je het *prj.conf*-bestand in de map van het aangepaste Blinky-programma en voeg je de volgende regel toe (anders dan in de device tree-bestanden met hun C-achtige syntax, gebruikt Kconfig een Python-syntax voor de configuratiebestanden):

CONFIG_LED=y

Om te controleren het werkt, voer je een pristine build van je project uit en laad je het programma in het board.

Debuggen

Als je board de optie biedt (zoals de BBC micro:bit) of als je een geschikt debug-tool bezit, kun je het programma debuggen met:

west debug

Dit commando start een gdb-server en opent een terminal (**figuur 12**). Op internet is informatie te vinden hoe je gdb kunt gebruiken, dit valt helaas buiten het kader van dit artikel.

Zephyr versus Arduino

Nu je hebt gezien wat er nodig is om met Zephyr OS aan de slag te gaan, vraag je je misschien af waarom je het zou gaan gebruiken. Is het niet veel simpeler om gewoon Arduino te gebruiken? Net als Zephyr ondersteunt Arduino meerdere processor-architecturen en honderden board-types, en er zijn duizenden drivers en bibliotheken beschikbaar voor Arduino. Als een toepassing of een bibliotheek de standaard Arduino-API gebruikt kan deze ook eenvoudig bruikbaar gemaakt worden voor andere ondersteunde platforms met vergelijkbare interfaces en periferie, precies zoals een Zephyr-toepassing. Beide zijn bovendien open source. Dus? Zephyr is bedoeld als zeer robuust RTOS van industriële kwaliteit en met mogelijkheden zoals task scheduling, memory management en device drivers. Daarnaast is Zephyr ontworpen om een breed scala aan complexiteit te ondersteunen, van eenvoudige IoT-apparaten tot zeer complexe embedded systemen. Het biedt meer flexibiliteit, maar vraagt ook uitgebreide kennis en begrip van embedded systeemontwerpen.

Arduino biedt ook real-time mogelijkheden, maar is geen RTOS maar een framework voor single-threaded toepassingen en de

focus ligt op toegankelijkheid en eenvoud van gebruik. Arduino abstraheert veel hardwarespecifieke details waardoor het makkelijk toegankelijk is voor beginners. Het kan eventueel gebruikt worden boven op een RTOS zoals Mbed OS voor complexe toepassingen en er wordt gewerkt aan de mogelijkheid om de Arduino Core API te gebruiken onder Zephyr ([7]).

Het is aan jou om te beslissen of je Zephyr al dan niet nodig hebt voor je volgende project. Je kunt het in elk geval eens proberen: ervaring met Zephyr staat in elk geval goed op het CV van een embedded software-ontwikkelaar.

Meer weten?

Tot zover dit artikel. Je hebt gezien dat Zephyr OS complex is en een redelijk steile leercurve heeft. In dit artikel heb ik geprobeerd deze curve wat minder steil te maken. Uiteraard is er nog veel meer te vertellen en te leren over Zephyr – denk vooral niet dat je nu alles weet. In [8] en [9] vind je links naar bronnen die een goed vervolg kunnen zijn voor geïnteresseerden. ►

vertaling: Adrie Kooijman — 230713-03

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.



Over de auteur

Na een carrière in maritieme en industriële elektronica begon Clemens Valens in 2008 als hoofdredacteur bij Elektor Frankrijk. Sindsdien heeft hij diverse functies gehad en recentelijk is hij begonnen op de afdeling productontwikkeling. Zijn interesse gaat vooral uit naar signaalverwerking en geluidsproductie.





Gerelateerde producten

- > BBC micro:bit V2 www.elektor.nl/19488
- Dogan Ibrahim, Get Started with the NXP i.MX RT1010 Development (Bundle) www.elektor.nl/20699
- > Warren Gay, FreeRTOS for ESP32-Arduino, Elektor 2020 www.elektor.nl/19341







Volg ons Zephyr-webinar voor een praktische inleiding tot IoT-projecten met Zephyr

Meer informatie op elektormagazine.com/webinars

WEBLINKS

- [1] Over het Zephyr-project een project van de Linux Foundation: https://zephyrproject.org/learn-about
- [2] Device Tree-specificaties: https://devicetree.org/specifications
- [3] Beginnen met Zephyr: https://docs.zephyrproject.org/latest/develop/getting_started/index.html
- [4] Devicetree syntax highlighting plug-in voor Visual Code Studio: https://github.com/plorefice/vscode-devicetree
- [5] Devicetree-API: https://docs.zephyrproject.org/latest/build/dts/api/api.html
- [6] De LED-periferie: https://docs.zephyrproject.org/latest/hardware/peripherals/led.html
- [7] Arduino met Zephyr gebruiken: https://dhruvag2000.github.io/Blog-GSoC22
- [8] Eli Huges whitepaper "From Hardware Concept to Zephyr Bring Up": https://zephyrproject.org/white-papers
- [9] USB seriële console-uitvoer in een Zephyr RTOS-applicatie: https://gnd.io/zephyr-console-output

Zij vertrouwen ons, u ook?



Bekroonde ethiek

een vraaggesprek met Alexander Gerfer, CTO van Würth Elektronik eiSos, over het faciliteren van innovatie en bewust gedrag

Vragen van Shenja Panik (Ethics in Electronics / Elektor)

Alexander Gerfer, CTO van Würth Elektronik eiSos, deelt zijn inzet voor ethiek en innovatie in een exclusief vraaggesprek met Shenja Panik. Aan de orde komen de initiatieven van het Hightech Innovation Center voor milieuverantwoordelijkheid, eerlijkheid en voorspelbaarheid, en ook hoe ethiek is geïntegreerd met zakelijk succes bij Würth Elektronik.

Shenja Panik: Gelukgewenst, meneer Gerfer, met het winnen van de 2023 Ethics in Electronics Award voor uw onvermoeibare inzet voor duurzaamheid, groene technologie en verticale landbouw. Bedankt dat u ons vandaag heeft uitgenodigd in uw Hightech Innovation Center in München en dat u de tijd heeft genomen om enkele vragen te beantwoorden. Kunt u uw rol op het gebied van ethiek in elektronica beschrijven? Alexander Gerfer: Om die uit te leggen, is het belangrijk om naar mijn achtergrond te kijken: opgegroeid op een boerderij met koeien en kippen, moest ik al heel vroeg verantwoordelijkheid dragen. Een ander cruciaal aspect is het gezegde: wat gij niet wilt dat u geschiedt, doe dat ook een ander niet. Dit zijn de basisprincipes van mijn opvoeding. Bij Würth Elektronik staan onze klanten altijd centraal. We discrimineren klanten niet op basis van de grootte van hun bedrijf of hun invloed. We onderhouden met iedereen een zeer coöperatieve relatie. We laten hen profiteren van onze expertise door middel van advies, voorbeeldontwerpen, online-presentaties, ontwikkelkits en softwaretools die helpen bij het dimensioneren. We vragen hiervoor of voor laboratoriummonsters van onze componenten geen geld. Iedereen is even belangrijk voor ons. We ondersteunen iedereen die kennis over selectie en toepassing kan gebruiken. We leveren en ondersteunen iedereen, van startups tot gevestigde bedrijven, van mensen die kleine hoeveelheden bestellen tot kopers van grote partijen. We hebben geen minimum bestelhoeveelheden. Dat is gebaseerd op onze bedrijfscultuur. Als start-up kun je tijdens de ontwikkeling falen vanwege een component van een cent. Dat weet ik uit eigen ervaring. Daarom streven we er altijd naar om problemen snel op te lossen - en trekken iemand die maximale omzet belooft niet voor.

Shenja Panik: Niet iedereen krijgt dezelfde kansen als een start-up, dus het is geweldig dat Würth Elektronik zich actief inzet voor gelijke mogelijkheden voor iedereen. Welke andere initiatieven of strategieën worden er in München geïmplementeerd om duurzaam en ethisch ondernemen te stimuleren?

Alexander Gerfer: In ons nieuwe Hightech Innovation Center in München testen we prototypes van elektronische apparaten en constructies op elektromagnetische compatibiliteit en doen we gerichte suggesties voor verbetering. EMC-certificering is immers vaak een grote hindernis voor ontwikkelaars. Eerlijkheid, voorspelbaarheid, gelijkheid, wederzijdse steun – dit zijn de principes voor het management binnen het bedrijf en voor hoe werknemers met elkaar omgaan.

Als onderdeel van de Würth Group dragen we ook proactief bij aan het behoud van ons milieu, zowel door onze producten als door milieuvriendelijk zakendoen. Tegenwoordig is het belangrijker dan ooit om zorg te dragen voor het milieu.

We denken altijd een stap vooruit. Neem bijvoorbeeld vertical farming: het kweken van voedsel in kassen met meerdere niveaus levert al een bijdrage aan het lokaal produceren van voedsel van hoge kwaliteit. Deze bijdrage moet echter aanzienlijk worden uitgebreid. Volgens schattingen van de Voedsel- en Landbouworganisatie van de VN (FAO) zullen er in 2050 10 miljard mensen op deze planeet leven. Dit betekent dat de wereldwijde landbouwproductie tegen 2050 met 50% moet toenemen. Toch is het landbouwareaal de afgelopen decennia afgenomen, van bijna 40% van het landoppervlak in 1991 tot slechts 37% in 2018. Planten zullen letterlijk naar de hemel moeten groeien als we willen dat iedereen goed gevoed wordt. Klimaatverandering en demografie vormen enorme uitdagingen en we moeten nu aan oplossingen werken

Figuur 1. Overhandiging van de Ethics-in-Electronics Award 2023 aan winnaar Alexander Gerfer.



– zelfs als ze in eerste instantie als inefficiënt en 'te duur' worden beschouwd.

Er is geen tekort aan goede ideeën om onze wereld beter te maken. Voor elk belangrijk idee is tegenwoordig echter elektronica nodig om het te implementeren. Hier leveren we een bijdrage als betrouwbare leverancier, maar nog belangrijker, als helpende hand voor kleine en middelgrote ondernemingen en ambitieuze start-ups. Hoewel we als bedrijf winst moeten maken, ligt de nadruk bij Würth Elektronik niet op winst op korte termijn, maar op de betekenis van het project.

Shenja Panik: Wat betekent deze prijs voor u en uw netwerk?

Alexander Gerfer: Voor mij persoonlijk is de prijs een bijzondere eer. We hebben hem ook met het team gedeeld, want ik doe het niet alleen. Het is een eer voor het hele team dat aan deze projecten werkt – maar bovenal is het een stimulans. We kunnen jammer genoeg niet stellen dat de wereld er de laatste tijd alleen maar beter op is geworden. Helaas zijn er op veel gebieden tekenen van confrontatie en helaas worden zelfs de meest fundamentele ethische principes terzijde geschoven. Het is daarom nog belangrijker dat we trouw blijven aan onze principes, onbaatzuchtig helpen, samen handelen, elkaar met respect behandelen, onze leefruimte behouden en proactief innovaties stimuleren ten voordele van de algemeenheid. We staan allemaal voor grote uitdagingen. We hebben enorme problemen op te lossen en het is het beste om die vanaf vandaag samen aan te pakken.

Shenja Panik: U hebt ons verteld over uw achtergrond en hoe die u tot dit punt heeft gebracht. Wat zijn nu uw belangrijkste inspiratiebronnen bij het nemen van ethische beslissingen?

Alexander Gerfer: Natuurlijk is er in de eerste plaats mijn persoonlijke ethische en waardenkompas dat ik van huis uit heb meegekregen. Het verantwoordelijkheidsgevoel voor de natuur en de medemens werd mij al op zeer jonge leeftijd bijgebracht. Daarom ligt bijvoorbeeld het onderwerp vertical farming met LED's me na aan het hart.

Belangrijk voor ons in de Würth Group zijn onze bedrijfswaarden, die onze houding en ons zakelijk gedrag op alle belangrijke gebieden bepalen. Wederzijds vertrouwen, voorspelbaarheid, eerlijkheid en oprechtheid, zowel intern als extern, zijn fundamentele principes die prof. dr. h. c. mult. Reinhold Würth al in de jaren '70 formuleerde.

De *Code of Compliance* van de Würth Group beslaat 32 pagina's en beschrijft algemene gedragsprincipes, richtlijnen voor de omgang met zakenpartners, informatie over het vermijden van belangenconflicten en de implementatie van de Code of Compliance.


Alvorens een beslissing te nemen, vragen mijn collega's of ik: is mijn beslissing of handeling in overeenstemming met de geldende wetten? Is de beslissing in overeenstemming met de bekende waarden en regels van Würth? Ervan uitgaande dat mijn collega's en familie op de hoogte zijn van mijn beslissing, zou ik dan een zuiver geweten hebben? Als iedereen op deze manier zou beslissen, zou ik dan met de gevolgen kunnen leven? Als mijn beslissing morgen in de krant zou staan, zou ik die dan kunnen rechtvaardigen?

Zoals ik al zei, is eerlijk zijn in je beslissingen en respectvol en dankbaar zijn voor elke bijdrage belangrijk, maar voorspelbaar en betrouwbaar zijn is net zo belangrijk.

Shenja Panik: Hoeveel doorzettingsvermogen is er nodig om ethische en duurzame initiatieven in een groot bedrijf te implementeren?

Alexander Gerfer: In onze dagelijkse werkzaamheden? Weinig tot geen, omdat we onze werknemers heel zorgvuldig selecteren. Compliance betekent voor ons niet alleen het naleven van alle toepasselijke regels, wetten en bedrijfsrichtlijnen, maar ook een overeenkomstige interne houding van medewerkers, wat een cruciale component is voor het duurzame zakelijke succes van Würth Elektronik.

Daarom verwachten we van onze managers, werknemers en zakenpartners dat ze nationaal en internationaal geldende wetten en regels naleven. Als onderdeel van onze waardencatalogus verplichten we ons bovendien tot het naleven van de normen, richtlijnen en standaarden die duidelijk zijn gedefinieerd in de Code of Compliance. Dit gedeelde waardenkompas is van toepassing op alle afdelingen van het bedrijf en wordt geïnternaliseerd door alle werknemers.

Tijdens het selectieproces beoordelen we sollicitanten niet alleen op hun professionele kwalificaties, maar vooral op hun persoonlijke houding.

Komen ze overeen met onze principes? Identificeren toekomstige werknemers zich met onze bedrijfswaarden, of hebben ze deze alleen uit het hoofd geleerd? Zijn ze meer gericht op duurzaam succes, of alleen op kwartaalcijfers? Zien ze collega's als partners of arrogant? Denken ze holistisch en milieubewust, of egocentrisch en opportunistisch?

Deze en nog veel meer vragen moeten beantwoord worden voordat iemand wordt aangenomen. Kennis kan worden opgedaan door on-the-job training en vaardigheden kunnen worden geleerd van ervaren collega's. De onmisbare basis zijn echter ethische richtlijnen die iedereen moet meenemen binnen de groep. De onmisbare basis wordt echter gevormd door ethische richtlijnen die iedereen in de groep moet inbrengen. Er is dus niet veel weerstand – de mensen die hier werken geloven in wat we doen en hoe we het doen. En we ervaren dagelijks dat onze medewerkers zeer goed omgaan met verantwoordelijkheid. Hier trekken bedrijfsleiding, leidinggevenden en medewerkers samen op. Ethiek wordt niet afgedwongen, maar geleefd binnen onze organisatie.

Shenja Panik: Het gaat er niet om van project naar project te gaan, maar om een bedrijfsfilosofie te handhaven. Jullie zijn allemaal betrokken bij de vraag hoe we van de wereld een betere plek kunnen maken.

Alexander Gerfer: Het zijn niet alleen onze eigen mensen die we motiveren, maar we motiveren ook mensen buiten ons bedrijf. Op basis van mijn eigen start-up ervaring en de uitdagingen die daarbij komen kijken, realiseerde ik me hoe belangrijk het is om ontwikkelaars al in een vroeg stadium de hand te reiken, vooral op het gebied van hardware, een van de meest uitdagende stappen in het begin. Een verkeerde beslissing kan zelfs het beste idee ondermijnen. Daarom bieden we hulp en ondersteuning. En we leren zelf tijdens dat proces... Onze overkoepelende visie is dus een volledig voorspelbaar begrip van de behoeften van de klant. Ethiek en zakelijk succes zijn voor ons niet tegenstrijdig, maar juist onafscheidelijk met elkaar verbonden.

vertaling: Willem den Hollander – 240013-03

Vragen of opmerkingen?

Hebt u vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar Shenja via shenja. panik@elektor.com.

Over Shenja Panik

Shenja Panik is een gepassioneerd sociologe die voor het content-team van EiE schrijft, vooral over ethiek binnen de elektronica-industrie. Haar werk draait om het verkennen en bevorderen van ethische overwegingen in het steeds veranderende technologische landschap.



Over Alexander Gerfer

Alexander Gerfer is de CTO van Würth Elektronik eiSos, een grote Europese fabrikant van elektronische componenten. Hij werkt al 27 jaar bij het bedrijf, bekleedde verschillende functies en bevordert technologische partnerschappen op gebieden zoals LED-lichtconcepten, energie-terugwinning en e-mobiliteit. Met zijn vakkennis is hij gepassioneerd over het ondersteunen en adviseren van innovatieve start-ups. Gerfer is ook auteur van technische boeken, docent en keynote spreker op internationale conferenties.

Sektor maart/april 2024 109



 \rightarrow C

o www.elektor.nl

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).

FNIRSI 1013D 2-kanaals Tablet Oscilloscoop (100 MHz)



The FNIRSI 1013D is a fully featured 2-channel tablet oscilloscope with a high-resolution 7-inch LCD screen (800 x 480 pixels). The oscilloscope has a real-time sampling rate of 1 GSa/s and an analog bandwidth of 100 MHz.

Prijs: € 159,95 Ledenprijs: € 143,96

🖵 www.elektor.nl/20644



Node-RED and Raspberry Pi Pico W

This book is a learning guide and a reference. Use it to learn Node-RED, Raspberry Pi Pico W, and MicroPython, and add these state-ofthe-art tools to your technology toolkit. It will introduce you to virtual machines, Docker, and MySQL in support of IoT projects based on Node-RED and the Raspberry Pi Pico W.



🕁 www.elektor.nl/20745

Oxocart Connect Innovator Kit



Prijs: € 89,95 Ledenprijs: € 80,96

🕁 www.elektor.nl/20718

SunFounder Kepler Kit (Ultimate Starter Kit voor Raspberry Pi Pico W)

☆



Ledenprijs: € 62,96

🕁 www.elektor.nl/20730

Temperatuurgeregeld soldeerstation ZD-931



Prijs: € 49,95 Ledenprijs: € 44,96

🕁 www.elektor.nl/20623

FNIRSI DSO152 Mini Oscilloscoop (200 kHz)



Prijs: € 39,95 Ledenprijs: € 35,96

्रि www.elektor.nl/20642



Project 2.0

correcties, updates en brieven van lezers

samengesteld door Jean-François Simon (Elektor)



Regelbare lineaire labvoeding Elektor januari/februari 2024, p. 26 (220457)

Er staat een fout in het schema van de dubbele regelbare labvoeding (p. 29). De negatieve spanningsregelaar onder in het schema moet een LM337 zijn en geen LM317. In de tekst van het wordt deze correct LM337 genoemd.

USB-Killer-detector Elektor november/december 2023, p. 32 (220549)

Is de USB-Killer-detector als opgebouwde schakeling leverbaar? N. D. (Duitsland)

Je kunt alle bestanden die nodig zijn om de USB-Killer-detector te bouwen vinden op GitHub (https://github.com/instantdevices/ USB-Killer-Detector). Hoewel ik misschien onderdelen kan leveren, ben ik nog op zoek naar hulp van de community vanwege een hardnekkig probleem waarbij een specifiek type 'killer' de detector te slim af is. Dus ik zou zeggen dat het apparaat nog niet helemaal marktrijp is. Je kunt me bereiken op instant.devices@yahoo.com. Carlos Guzman (auteur van het artikel)

Heb je een goed idee of waardevolle feedback voor Elektor? Neem contact met ons op via redactie@elektor.com - we horen graag van je!

cartee calman faatear van net artike

Installeren van een SMT-productielijn

Elektor november/december 2023, p. 108 (230593) In dit artikel presenteerde u de Whizoo Controleo3. Kunt u een leverancier noemen, indien mogelijk in Duitsland? *Wolf-Peter Kleinau (Duitsland)*

Dat artikel is geschreven door onze partners bij Opulo, een Amerikaans bedrijf. Whizoo is ook een in de VS gevestigd bedrijf en helaas zijn er, voor zover ik weet, geen wederverkopers in de EU. Bovendien suggereert het artikel van Opulo dat de 'Controleo3' een opgebouwde en 'out of the box bruikbare' reflow-oplossing is. Om precies te zijn was de Controleo3 eerst bekend als een bouwpakket om je eigen reflow-oven te bouwen, en het toeval wil dat Whizoo ook een volledig geassembleerde oven verkoopt voor ongeveer € 1200. Helaas is dit product alleen verkrijgbaar in de VS en Canada en wordt het geleverd met een 115V-oven. Als je voor Controleo3-producten kiest, kun je ze nog steeds als kit kopen in de VS, maar dan moet je apart BTW en invoerrechten betalen, wat lastig kan zijn, en je moet je zelf voor een oven zorgen en die in elkaar zetten, en dat is dus niet 'reflow-klaar'.

Er is geen eenvoudig antwoord op de vraag "Welke reflow-oven moet ik kopen?". Dit is een breed gebied en ik zou u willen aanraden om uitgebreid onderzoek te doen om te beslissen wat het beste aan je behoeften voldoet. Persoonlijk kan ik geen aanbevelingen doen, maar ik heb gehoord over het bedrijf Beta Layout. Van andere fabrikanten zijn twee populaire opties de T962-oven (in veel variaties) en de ZB3530HL. Op het internet circuleren ook manieren om de T962 te verbeteren voor een betere warmteverdeling of een betere gebruikersinterface.

Jean-François Simon (Elektor)

De Raspberry Pi 5

Elektor november/december 2023, p. 6 (230635) Hartelijk dank voor dit informatieve artikel. Voor de verschillende Raspberry Pi 4's die ik bezit, ben ik gewend om SD-kaarten te kopiëren om alles wat moeizaam was geïnstalleerd (zoals VS-code, Qt 5, CAN-bus enzovoort) gemakkelijk over te zetten naar andere apparaten. Kunt u mij en andere lezers vertellen hoe ik deze kan overzetten naar de Pi 5? Ik heb de gekopieerde SD-kaart in de Pi 5 geplaatst, maar het werkt niet. Hartelijk dank! *Hanspeter Schären (Duitsland)*

Bedankt voor je e-mail. Dit heeft te maken met de versie van het Raspberry Pi OS dat aanwezig is op de SD-kaarten die je hebt gebruikt met de Pi 4. De Pi 5 vereist de OS-versie Bookworm en kan niet overweg met eerdere versies. In jouw geval zul je dus helaas vanaf nul een nieuwe SD-kaart moeten maken, bijvoorbeeld met de Raspberry Pi Imager, die eenvoudig in het gebruik is. En je zult de software die je nodig hebt met de hand opnieuw moeten installeren. Daarna zou die SD-kaart met Bookworm erop in principe moeten werken met zowel de Pi 5 als de Pi 4 als je ooit van de ene naar de andere Raspberry moet overschakelen, maar je moet zelf uitproberen of alles werkt. Veel succes! *Jean-François Simon (Elektor*)

MEMS-microfoon

Elektor november/december 2023, p. 70 (230188)

We hebben een fout gemaakt bij het hertekenen van het schema (figuur 3, p. 72). Onze excuses daarvoor! De niet-inverterende ingangen van opamps U1...U6 zijn verbonden met Vref via $47k\Omega$ -weerstanden, en niet met massa (zie het fragment van het schema rechts). Het originele schema dat beschikbaar is in het *Hackaday.io*-project waarnaar in het artikel wordt gelinkt, is natuurlijk correct.



Koude-kathode-buizen

Elektor januari/februari 2024, p. 74 (230373) Ik heb gehoord dat sommige koude-kathode-buizen zwak radioactief zijn. Is dat waar? Bijvoorbeeld buizen van Elesta en andere zoals de ER21A, GT21, 5823, of de GR16?

Adolf Parth (Italië)

Je hebt gelijk. Sommige van deze buizen bevatten radioactief materiaal om het gas erin te ioniseren! Er bestaan verschillende websites op het internet, gemaakt door liefhebbers van radioactiviteit en geigertellers, die je meer details geven over welke specifieke buizen welk radioactief materiaal bevatten. Jean-Francois Simon (Elektor)

De functie van een kathode (koud of verhit) is meestal om elektronen uit te zenden. In het geval van verhitte kathodes zenden ze elektronen uit die meestal gemoduleerd worden door een rooster en opgevangen worden door een anode. Bij koude-kathode-buizen zenden ze meestal elektronen uit om een gas te ioniseren, ofwel voor overspanningsbeveiliging, voor weergave (neonen nixie-buizen) of om te schakelen, wat het geval is bij de koude-kathode thyratronbuizen die de heer Parth noemt.

Daarom worden kathodes vaak gecoat met stoffen die de emissie van elektronen bevorderen. In het geval van standaard buizen met verhitte kathode wordt meestal thorium gebruikt. In buizen met koude kathode worden een aantal zeldzame aardmetalen en verbindingen gebruikt. Sommige hiervan vertonen een lichte radioactiviteit. In het periodiek systeem behoren de zeldzame aardmetalen tot de lanthanidenreeks en thorium tot de verwante actinidenreeks. Enkele lanthaniden zijn radioactief en de meeste actiniden zijn dat ook. Thorium wordt ook gebruikt in de gloeikousjes van gaslampen, en deze zullen dan ook licht radioactief zijn. Veel gewone elementen bevatten voor een deel isotopen die radioactief zijn, zoals kalium.

David Ashton (auteur van het artikel)

Bron: https://en.wikipedia.org/wiki/Nixie_ tube#/media/File:ZM1210-operating_edit2.jpg



Motordriver breakout-board

Elektor september/oktober 2023, p. 56 (210657) Hallo allemaal! Is het mogelijk om met behulp van een NTC 10k Ω -sensor een temperatuurafhankelijke snelheidsregeling te maken voor een ventilator met de motordriver MP6619, die dan uitgangsspanningen levert van 3 V_{DC} bij 20 °C tot (bijvoorbeeld) 12 V_{DC} bij 40 °C? Hartelijk dank voor eventuele informatie of voorbeeldschakelingen. Of is er misschien nog een andere mogelijkheid om zo'n snelheidsregeling te bouwen?

Matthias Seesemann (Duitsland)

Het is mogelijk om dit met een MP6619 te doen, maar je hebt ook andere componenten nodig. De MP6619 op zich is slechts een H-brug, dus hij zet zijn uitgangstransistoren aan of uit afhankelijk van de status van de IN1-, IN2en ENABLE-ingangen. Om een uitgangsspanning van 3...12 V te krijgen, zou je PWM moeten gebruiken om de ingangen van de MP6619 aan te sturen. De EN-pin moet hoog worden getrokken met een pull-up weerstand, dan kun je IN2 hoog of laag maken afhankelijk van de draairichting die je nodig hebt en tegelijkertijd stuur je een PWM-signaal naar IN1 om de snelheid van de motor te variëren. Om het PWM-signaal te genereren kun je elke microcontroller gebruiken, Arduino of welke dan ook. Ik raad je aan een Arduino Uno R3 te gebruiken. Daarvoor vind je online genoeg codevoorbeelden. Die demonstreren ook hoe je een temperatuur uitleest met een NTC, en hoe je PWM-functies gebruikt op de Arduino. Misschien wil je ook enkele online-projecten met de 'fan speed controller' TC648 bekijken: ik denk dat deze precies zijn wat je nodig hebt. Misschien moet je een of twee weerstanden veranderen om het zo te laten werken als je wilt. Zet je project op ons Elektor Labs-platform (www.elektormagazine.com/labs), zodat andere leden van de Elektor-community commentaar kunnen geven en je kunnen helpen. Veel plezier! Jean-François Simon (Elektor)

Word lid van de Elektor Community



Neem **nu** een



lidmaatschap!

Toegang tot het compleet web-archief t/m 1960! 🗹 8x Elektor Magazine (Print) 🗹 8x digitaal (PDF) 🗹 10% korting in de Elektor Store, en exclusieve aanbiedingen 🗹 Toegang tot meer dan 5000 Gerberfiles



Ook verkrijgbaar Het digitale GREEN lidmaatschap!

- Toegang tot het compleet web-archief
- 🗹 10% korting in de Elektor Store
- 쭏 8x Elektor Magazine (PDF)
- ✓ Toegang tot meer dan 5000 Gerberfiles



www.elektormagazine.nl/Abonnement





Vergroot uw capaciteiten

Snelle tips, tools en artikelen voor inkoopprofessionals

mou.sr/purchasing-resources