

Neues LCR-Meter

50 Hz – 2 MHz
10 mΩ bis 100 MΩ

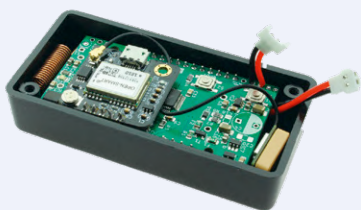
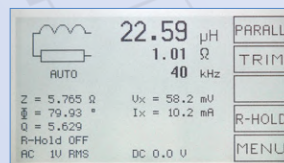
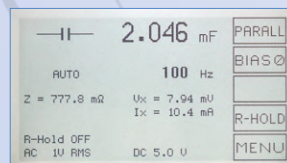
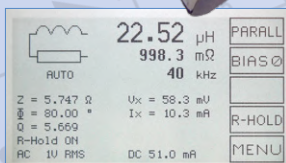
S. 6

Von Jean-Jacques Aubry

Interesse
am Bausatz?
elektor.de/1cr

In dieser Ausgabe

- > Differentieller Stromtastkopf für Oszilloskope
- > Weihnachtswunderkerze
- > Grundlagen: Batterie-Management
- > Interview: Die Zukunft des maschinellen Lernens
- > KiCad-Plugins und Add-ons
- > Zustandsautomaten mit 8-Bit-PICs
- > ESP32-Multitasking: Task-Ereignisse
- > Smartphone-Apps für Android und iOS
- > Breadboardansichten mit Fritzing
- > Entwicklung: Aktive Filter
- > Rennbahn-Rundenzähler und vieles mehr!



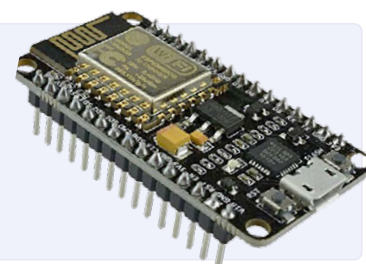
LoRa-GPS-Tracker
Offene Hardware und Software

S. 98



Review: Rigol DG2072 Generator
Elektronik-YouTuber
Philippe Demerliac
über Vor- und Nachteile

S. 62



DIY-Haus-Automatisierung
Integrieren Sie Ihre Türklingel
– mit Home Assistant und ESPHome

S. 20





UNSER SORTIMENT VON TECHNIKERN FÜR TECHNIKER

The best part of your project: www.reichelt.de/sortiment

Nur das Beste für Sie – von über 900 Markenherstellern.

Unsere Produktmanager sind seit vielen Jahren bei reichelt tätig und kennen die Anforderungen unserer Kunden. Sie stellen ein breites Spektrum an Qualitätsprodukten zusammen, optimal auf den Bedarf in Forschung & Entwicklung, Instandhaltung, IT-Infrastruktur und Kleinserienproduktion sowie auf Maker zugeschnitten.

Weller Lötstation WE1010

Die stärkste Lötstation in Ihrer Klasse!

ESD-sichere Lötstation mit hoher Temperaturstabilität und intuitiver Navigation. Die Standby-Funktion mit automatischem Setback schützt das Werkzeug und die Umwelt.

- leistungsstark - 70 W
- benutzerfreundlich - klares LC-Display
- kosteneffizient - geringe Betriebskosten

Weller®



BESTSELLER

Bestell-Nr.: WELLER WE1010

164,69

AUSGABE 01 | 2021

DER NEUE IST DA!

- mehr als 110.000 Artikel aus Elektronik & IT
- über 1.800 Seiten
- mehr als 10.000 Neuheiten



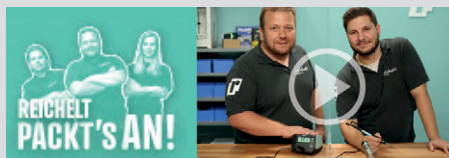
Jetzt anfordern ►



Löttechnik für jeden Einsatzzweck
finden Sie online!



Gleich entdecken ► www.reichelt.de/loeten



**LÖTEN
FÜR PROFS**



Jetzt auf YouTube/reichelt ►

- Top Preis-Leistungs-Verhältnis
- über 110.000 ausgesuchte Produkte

- Zuverlässige Lieferung – aus Deutschland in alle Welt.

Bestellservice: +49 (0)4422 955-333

www.reichelt.de

reichelt
elektronik – The best part of your project

Es gelten die gesetzlichen Widerrufsregelungen. Alle angegebenen Preise in € inklusive der gesetzlichen MwSt., zzgl. Versandkosten für den gesamten Warenkorb. Es gelten ausschließlich unsere AGB (unter www.reichelt.de/agb, im Katalog oder auf Anforderung). Abbildungen ähnlich. Druckfehler, Irrtümer und Preisänderungen vorbehalten. reichelt elektronik GmbH & Co. KG, Elektronikring 1, 26452 Sande, Tel.: +49 (0)4422 955-333

TAGESPREISE! Preisstand: 8. 10. 2020

51. Jahrgang, Nr. 576
November/Dezember 2020
ISSN 0932-5468

Erscheinungsweise: 9x jährlich
(6x Elektor-Doppelheft + 3x Elektor Industry Magazin)

Verlag

Elektor Verlag GmbH
Kackertstraße 10
52072 Aachen
Tel. 0241 95509190

Technische Fragen bitten wir per E-Mail an
redaktion@elektor.de zu richten.

Hauptsitz des Verlags

Elektor International Media
Postbus 11, 6114 ZG Susteren
Niederlande

Anzeigen

Margriet Debeij (verantwortlich)
Tel. 0241 95509174
Mobil: +49 170 5505396
E-Mail: margriet.debeij@elektor.com

Büra Kas

Tel. 0241 95509178
E-Mail: busra.kas@elektor.com

Es gilt die Anzeigenpreisliste ab 01.01.2020.

Distribution

IPS Pressevertrieb GmbH
Postfach 12 11, 53334 Meckenheim
Tel. 02225 88010
Fax 02225 8801199

Druck

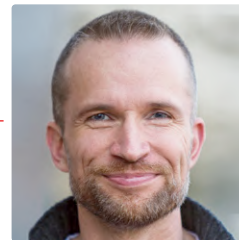
Pijper Media, Groningen (NL)

Der Herausgeber ist nicht verpflichtet, unverlangt eingesandte Manuskripte oder Geräte zurückzusenden. Auch wird für diese Gegenstände keine Haftung übernommen. Nimmt der Herausgeber einen Beitrag zur Veröffentlichung an, so erwirbt er gleichzeitig das Nachdruckrecht für alle ausländischen Ausgaben inklusive Lizenzen. Die in dieser Zeitschrift veröffentlichten Beiträge, insbesondere alle Aufsätze und Artikel sowie alle Entwürfe, Pläne, Zeichnungen einschließlich Platinen sind urheberrechtlich geschützt. Ihre auch teilweise Vervielfältigung und Verbreitung ist grundsätzlich nur mit vorheriger schriftlicher Zustimmung des Herausgebers gestattet. Die veröffentlichten Schaltungen können unter Patent- oder Gebrauchsmusterschutz stehen. Herstellen, Feilhalten, Inverkehrbringen und gewerblicher Gebrauch der Beiträge sind nur mit Zustimmung des Verlages und ggf. des Schutzrechtsinhabers zulässig. Nur der private Gebrauch ist frei. Bei den benutzten Warenbezeichnungen kann es sich um geschützte Warenzeichen handeln, die nur mit Zustimmung ihrer Inhaber warenzeichengemäß benutzt werden dürfen. Die geltenden gesetzlichen Bestimmungen hinsichtlich Bau, Erwerb und Betrieb von Sende- und Empfangseinrichtungen und der elektrischen Sicherheit sind unbedingt zu beachten. Eine Haftung des Herausgebers für die Richtigkeit und Brauchbarkeit der veröffentlichten Schaltungen und sonstigen Anordnungen sowie für die Richtigkeit des technischen Inhalts der veröffentlichten Aufsätze und sonstigen Beiträge ist ausgeschlossen.

© 2020 elektor international media b.v.

von Jens Nickel

Chefredakteur Elektor



Bauen und Lernen

Titelprojekt dieser Ausgabe ist ein LCR-Meter. Vom selben Autor gab es schon 2013 ein ähnliches Projekt in Elektor - die sechsteilige Serie wurde damals auch kommerziell zu einem großen Erfolg. Freilich steht die Zeit nicht still. Leistungsfähige Messgeräte sind inzwischen auch für Einsteiger-Budgets zu haben, wie man an vielen Oszilloskopen im Elektor-Store sehen kann. Und auch der Anteil derjenigen Leser, die große Elektor-Projekte 1:1 nachbauen möchten, mag sich in den Jahren verschoben haben. Wir wollten daher unsere Leser selbst entscheiden lassen, ob wir einen Bausatz auf den Markt bringen. Wenn Sie daran interessiert sind, ein Kit zu erwerben, dann können Sie uns Ihr (unverbindliches) Interesse unter www.elektor.com/lcr mitteilen. Noch vor Veröffentlichung der eigentlichen Projekt-Artikel (in diesem und dem nächsten Heft) gab es schon über 100 Interessenten. Das angestrebte Ziel von 150 „Unterstützern“ könnte also bereits mit dieser Ausgabe erreicht sein.

Für unser Redaktionsteam, das gerade über die Inhalte des nächsten Jahres nachdenkt, stellt sich somit immer wieder die Frage, wie wir die Entwicklung der Elektronik am besten begleiten. Denn wie schon gesagt, rein finanziell ist es in den letzten Jahren noch weniger lohnend geworden, Geräte und Module selbst zu bauen. Ein weiteres Beispiel ist auch unser zweites großes Projekt in diesem Heft, ein GPS-Tracker, der seine Position über LoRa-Netzwerke wie zum Beispiel TheThingsNetwork übermitteln kann. Solche Module sind inzwischen für eine Handvoll Euro zu kaufen - hier kann der Selbstbau nicht mithalten. Ich habe mich im Kollegenkreis trotzdem für das Projekt stark gemacht. Denn wenn Sie unser Projekt aufbauen, erhalten Sie ein komplett transparentes System - wir bieten auch die CAD-Files an, so dass Sie unser Projekt nach Belieben erweitern und „modden“ können. Und wir begleiten das Projekt mit Artikeln, bei denen Sie einiges über das Aufsetzen eines eigenen LoRa-Projektes lernen können. Natürlich können auch diejenigen etwas mitnehmen, die nicht die Zeit und Lust haben, den Lötkolben anzuhetzen oder das Steckbrett aus der Schublade zu holen. Wer sich aber die Zeit dafür nimmt, dem geht in Fleisch und Blut über, was Begriffe wie „Network Session Key“ und „Device Address“ bedeuten.

Genau diese Mischung aus Theorie und Praxis ist es, was Elektor auszeichnet - und dem bleiben wir auch in 2021 treu!

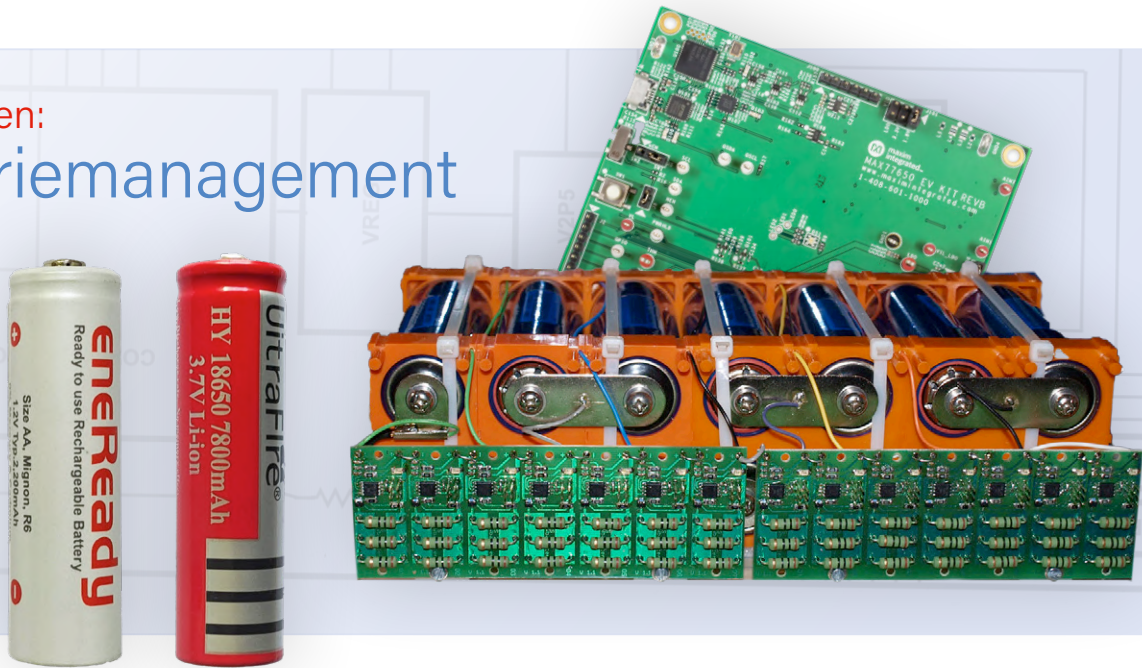
— Unser Team —



Chefredakteur: Jens Nickel (v.i.S.d.P.) (redaktion@elektor.de)
Redaktion: Eric Bogers, Jan Buiting, Stuart Cording, Rolf Gerstendorf, Denis Meyer, Dr. Thomas Scherer, Clemens Valens
Leserservice: Ralf Schmiedel
Elektor-Labor: Mathias Claußen, Ton Giesberts, Hedwig Hennekens, Luc Lemmens, Clemens Valens, Jan Visser
Grafik & Layout: Giel Dols, Harmen Heida
Herausgeber: Don Akkermans

Grundlagen: Batteriemanagement

25



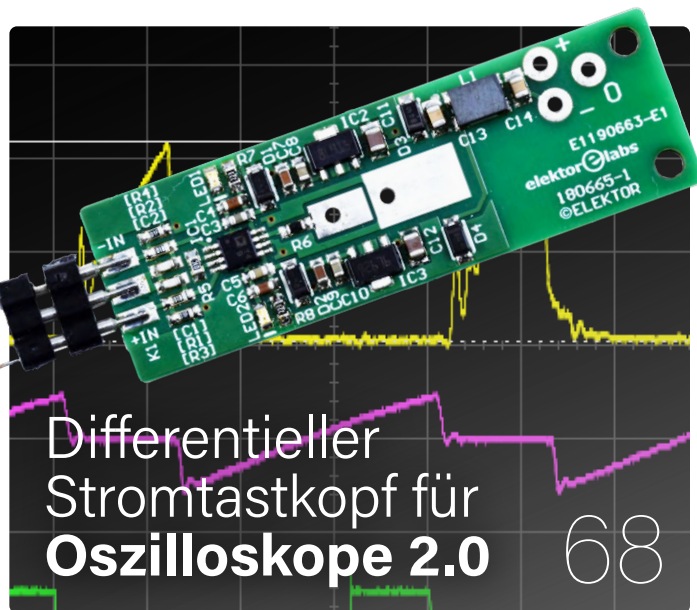
Rubriken

- 3 Impressum**
- 30 Aller Anfang...**
Basiskurs für Einsteiger
- 32 Kleine Schaltungen**
- 48 Von Entwicklern für Entwickler**
Tipps & Tricks, Best Practice und andere nützliche Infos
- 74 Retronik**
Die Hewlett-Packard Interface-Loop
- 90 Aus dem Leben gegriffen**
Der Zusammenprall von Alphas, Betas und Gammas
- 92 Elektor Ethics**
5G: Wie Infrastruktur unsere Gesellschaft formt
- 114 Hexadoku**
Sudoku für Elektroniker
- 51 Review: Vierkanal-Oszilloskop Rigol DS1054Z**
- 62 Review: Generator Rigol DG2072**
- 66 KiCad-Plugins und Add-Ons**
- 78 Die Zukunft des maschinellen Lernens**
Ein Interview mit Daniel Situnayake
- 82 Mobile Apps für Android und iOS**
Aus einem Guss programmiert
- 94 Praktisches ESP32-Multitasking (5)**
Task-Ereignisbenachrichtigung
- 107 Programmierung eines Endlichen Zustandsautomaten**
Mit 8-bit-PICs in Assembler und C

Projekte

Hintergrund

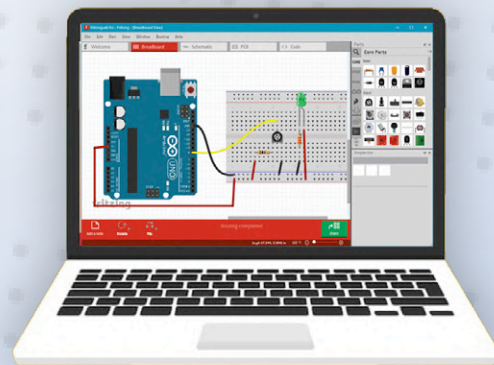
- 25 Batteriemanagement**
Was man beim Einsatz von (Lithium-)Akkus beachten muss
- 34 Breadboardansichten mit Fritzing**
Schritt für Schritt
- 38 Design analoger Filter (Teil 2)**
Aktive Filter
- 6 Neues LCR-Messgerät 50 Hz bis 2 MHz**
Automatische Impedanzmessbrücke misst Widerstand, Kapazität und Induktivität von Bauteilen mit einer Impedanz von 10 mΩ bis 100 MΩ
- 20 Vom Ding-Dong-Türgong zur IoT-Türglocke**
Verbinden Sie Ihre Türklingel im Home Assistant mit ESPHome!
- 46 Mein Projekt**
Rundenzähler für Rennbahnen



Differentieller Stromtastkopf für Oszilloskope 2.0

68

Breadboardansichten mit Fritzing



34



Neues LCR-Meter 50 Hz – 2 MHz

6

Vorschau

Elektor Januar/Februar 2021

Das nächste Heft ist wie immer randvoll gefüllt mit Schaltungsprojekten, Grundlagen sowie Tipps und Tricks für Elektroniker.

Aus dem Inhalt:

- > 32-Bit-I2S-Sinusgenerator
- > DIY-Thermo-Cam
- > Power-Analyzer
- > Bluetooth-Beacons in der Praxis
- > Homelab-Projekt: Feinstaubsensor
- > Hausautomatisierung mit Home Assistant (3)
- > FFT auf dem Maixduino
- > Objektorientierte Programmierung für Einsteiger

Und vieles mehr!

Änderungen vorbehalten.

Elektor Januar/Februar 2021 erscheint am 7. Januar 2021.

54 Weihnachtswunderkerze

Puste sie aus!

58 Durchstimmbare Röhren-Sinusgenerator

Retro ist in...

68 Differentieller Stromtastkopf für Oszilloskope 2.0

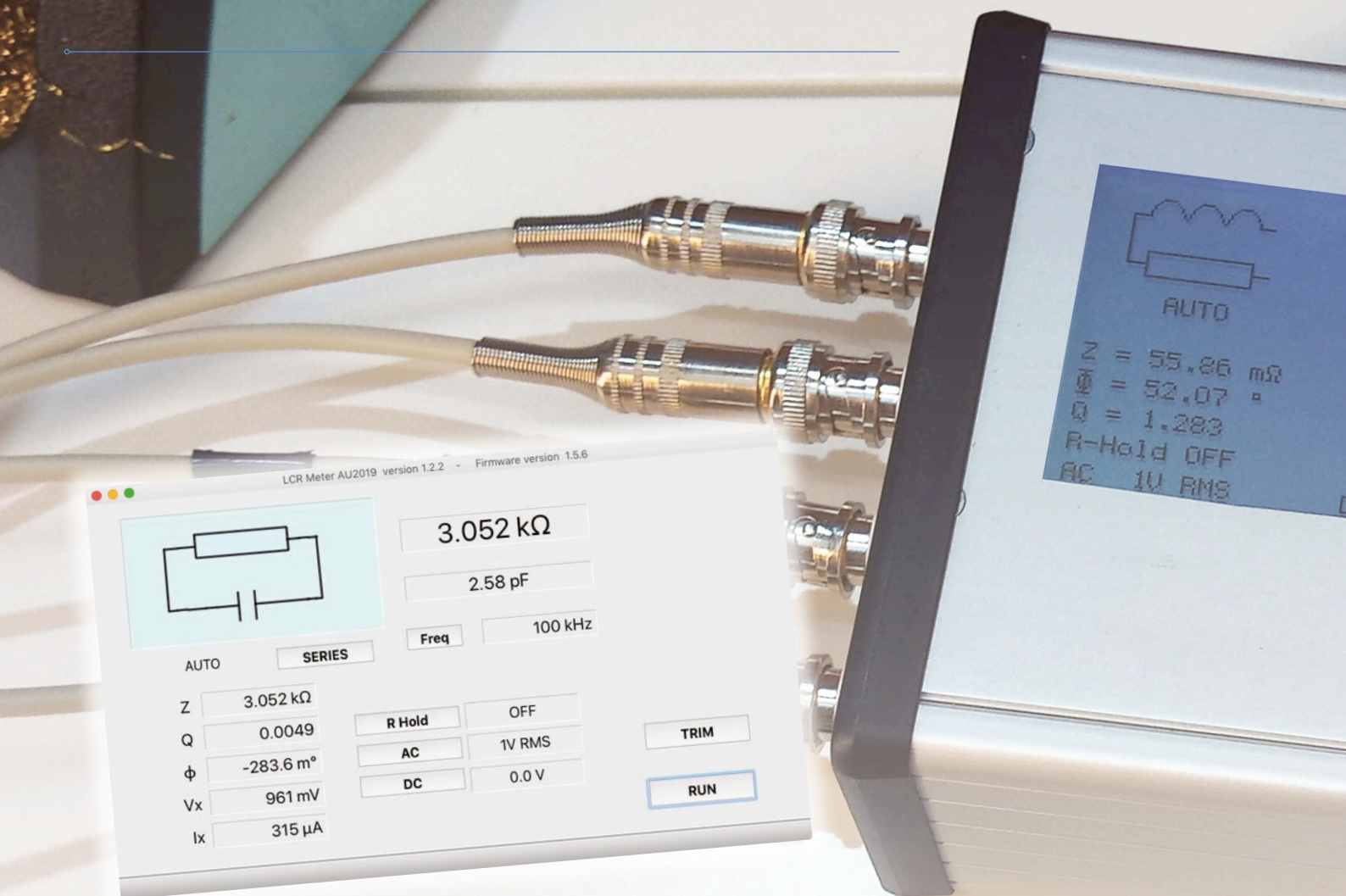
Ströme mit dem Oszilloskop messen

98 LoRa-GPS-Tracker

Mit offener Hard- und Software

Besuchen Sie die
neue Labs-Website



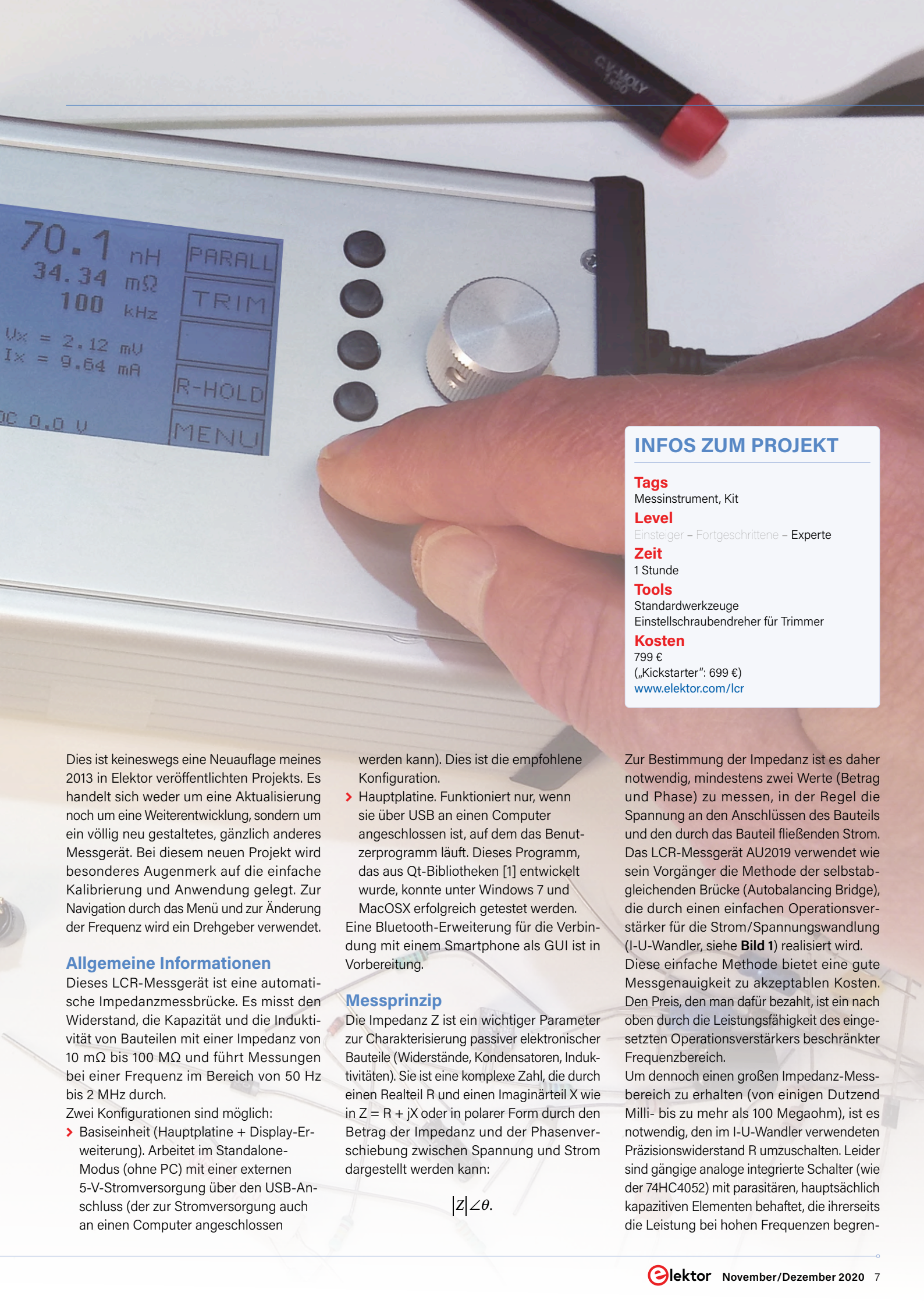


Neues LCR-Messgerät 50 Hz bis 2 MHz

Automatische Impedanzmessbrücke misst Widerstand, Kapazität und Induktivität von Bauteilen mit einer Impedanz von 10 mΩ bis 100 MΩ

Von **Jean-Jacques Aubry** (Frankreich)

Vor mehr als sieben Jahren stellte Elektor mein 500-ppm-LCR-Messgerät vor. Seitdem habe ich ein völlig neues LCR-Messgerät entwickelt, bei dem ich zugunsten einer erweiterten Funktionalität auf extreme Genauigkeit verzichtet habe: Prüffrequenz von 50 Hz bis 2 MHz, vier mögliche Prüfspannungen (100 mV, 200 mV, 500 mV und 1 Vrms) und einen zusätzlichen Offset bis zu 5 V für Kondensatoren und 50 mA für Induktivitäten.



INFOS ZUM PROJEKT

Tags

Messinstrument, Kit

Level

Einsteiger – Fortgeschrittene – Experte

Zeit

1 Stunde

Tools

Standardwerkzeuge

Einstellschraubendreher für Trimmer

Kosten

799 €

(„Kickstarter“: 699 €)

www.elektor.com/lcr

Dies ist keineswegs eine Neuauflage meines 2013 in Elektor veröffentlichten Projekts. Es handelt sich weder um eine Aktualisierung noch um eine Weiterentwicklung, sondern um ein völlig neu gestaltetes, gänzlich anderes Messgerät. Bei diesem neuen Projekt wird besonderes Augenmerk auf die einfache Kalibrierung und Anwendung gelegt. Zur Navigation durch das Menü und zur Änderung der Frequenz wird ein Drehgeber verwendet.

Allgemeine Informationen

Dieses LCR-Messgerät ist eine automatische Impedanzmessbrücke. Es misst den Widerstand, die Kapazität und die Induktivität von Bauteilen mit einer Impedanz von 10 mΩ bis 100 MΩ und führt Messungen bei einer Frequenz im Bereich von 50 Hz bis 2 MHz durch.

Zwei Konfigurationen sind möglich:

- Basiseinheit (Hauptplatine + Display-Erweiterung). Arbeitet im Standalone-Modus (ohne PC) mit einer externen 5-V-Stromversorgung über den USB-Anschluss (der zur Stromversorgung auch an einen Computer angeschlossen

werden kann). Dies ist die empfohlene Konfiguration.

- Hauptplatine. Funktioniert nur, wenn sie über USB an einen Computer angeschlossen ist, auf dem das Benutzerprogramm läuft. Dieses Programm, das aus Qt-Bibliotheken [1] entwickelt wurde, konnte unter Windows 7 und MacOSX erfolgreich getestet werden. Eine Bluetooth-Erweiterung für die Verbindung mit einem Smartphone als GUI ist in Vorbereitung.

Messprinzip

Die Impedanz Z ist ein wichtiger Parameter zur Charakterisierung passiver elektronischer Bauteile (Widerstände, Kondensatoren, Induktivitäten). Sie ist eine komplexe Zahl, die durch einen Realteil R und einen Imaginärteil X wie in $Z = R + jX$ oder in polarer Form durch den Betrag der Impedanz und der Phasenverschiebung zwischen Spannung und Strom dargestellt werden kann:

$$|Z| \angle \theta.$$

Zur Bestimmung der Impedanz ist es daher notwendig, mindestens zwei Werte (Betrag und Phase) zu messen, in der Regel die Spannung an den Anschlüssen des Bauteils und den durch das Bauteil fließenden Strom. Das LCR-Messgerät AU2019 verwendet wie sein Vorgänger die Methode der selbstabgleichenden Brücke (Autobalancing Bridge), die durch einen einfachen Operationsverstärker für die Strom/Spannungswandlung (I-U-Wandler, siehe **Bild 1**) realisiert wird. Diese einfache Methode bietet eine gute Messgenauigkeit zu akzeptablen Kosten. Den Preis, den man dafür bezahlt, ist ein nach oben durch die Leistungsfähigkeit des eingesetzten Operationsverstärkers beschränkter Frequenzbereich.

Um dennoch einen großen Impedanz-Messbereich zu erhalten (von einigen Dutzend Milli- bis zu mehr als 100 Megaohm), ist es notwendig, den im I-U-Wandler verwendeten Präzisionswiderstand R umzuschalten. Leider sind gängige analoge integrierte Schalter (wie der 74HC4052) mit parasitären, hauptsächlich kapazitiven Elementen behaftet, die ihrerseits die Leistung bei hohen Frequenzen begren-

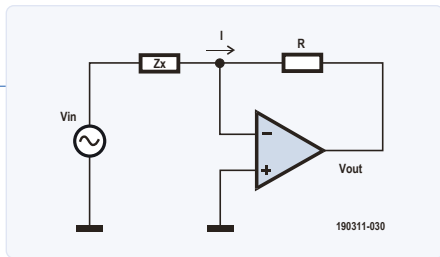


Bild 1. Die selbstabgleichende Brücke verwendet einen einfachen Operationsverstärker als Strom/Spannungswandler.

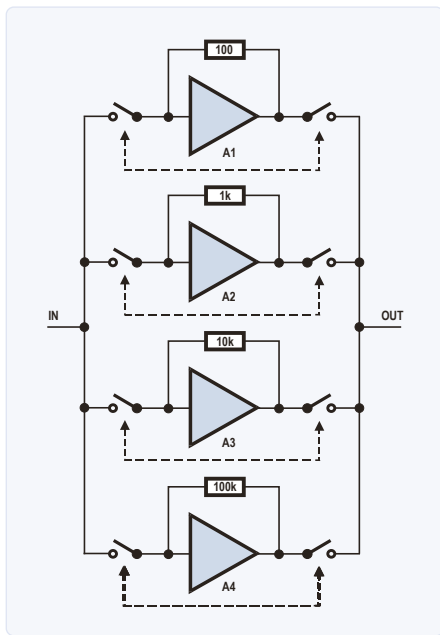


Bild 2. Vier Messbereiche, vier Brücken.

zen. Aus diesem Grund ist der Frequenzbereich der meisten ähnlich aufgebauten Messgeräte auf 100 kHz bis 200 kHz begrenzt. Durch ein originelles, wohlüberlegtes Design und die Wahl von hochqualitativen Bauteilen

ist es jedoch möglich, den Frequenzbereich ohne übermäßige Kosten auf 2 MHz aufzubo-
hren und gleichzeitig die Einfachheit des Designs zu erhalten.

Die gewählte Lösung, die in **Bild 2** dargestellt ist, besteht nicht darin, nur die vier Messwiderstände umzuschalten (und damit die parasitären Kapazitäten auf ein Minimum zu reduzieren), sondern gleich vier Paare von Verstärkern inklusive Widerstand. Dabei ist jedes Paar so bemessen, dass es mit der zu messenden Impedanz übereinstimmt.

Die verwendeten Operationsverstärker (AD8099 von Analog Devices) haben eine Grenzfrequenz von etwa 200 MHz bei einer Ausgangsspannung von $2 V_{SS}$ und verfügen über einen Steuereingang, über den der Ausgang gesperrt (stummgeschaltet) werden kann. Das ist genau das, was wir benötigen! Als Schalter werden Solid-state-Relais aus der PhotoMOS-Serie von Panasonic verwendet, die sich durch ein sehr niedriges Produkt von ON-Widerstand und Ausgangskapazität auszeichnen.

Ein weiterer wichtiger Punkt betrifft die Erzeugung der Testfrequenz. Heutzutage ist es einfach und kostengünstig, dafür DDS-ICs (Direkte Digitale Synthese) zu verwenden, mit dem Vorteil, dass man jede beliebige Frequenz im Bereich von 50 Hz bis 2 MHz erzeugen kann. Darüber lässt sich mit einer zweiten, zur ersten synchronen DDS-Schaltung ein Signal mit der gleichen Frequenz, aber mit variabler relativer Phase für den Synchrondetektor erzeugen.

Blockschaltbild

Die Blockschaltung in **Bild 3** zeigt die einzelnen Funktionsgruppen der Hardware des LCR-Messgeräts und wie diese Gruppen miteinander verbunden sind. Jede Funktions-

gruppe wird später in diesem Artikel besprochen. Die Benutzerschnittstelle (GUI Extension) ist auf einer separaten Platine untergebracht, wobei die Bedienelemente strikt auf ihr Minimum reduziert sind:

- LCD-Grafikdisplay mit 240 x 128 Punkten
- 5 Drucktasten
- Ein Drehgeber mit Hilfskontakt (Mitteltaster).

Obwohl ich davon abrate, können Sie diese GUI-Platine weglassen und das LCR-Messgerät über die USB-Schnittstelle eines PCs steuern. Zur Zeit bietet mein PC-Programm allerdings nicht den gleichen Funktionsumfang wie meine Standalone-Version. Es ist Geschmackssache, aber ich persönlich glaube, dass ein Standalone-Messgerät immer die beste Wahl ist. Physikalische Tasten sind einfacher und schneller zu bedienen und es ist nicht jedes Mal ein Update erforderlich, wenn das PC-Betriebssystem aktualisiert wird. Das LCR-Messgerät wird über einen Mini-USB-Anschluss mit Strom versorgt. Sie können ein Smartphone-Ladegerät oder eine externe Powerbank verwenden oder das Messgerät an einen Computer anschließen (letzteres ermöglicht auch ein Firmware-Update).

Die Schaltung auf der Hauptplatine

Analoge Eingangssektion (Bild 4)

Die Messung wird mit einer 5-Terminal-Konfiguration [2] durchgeführt, um den Einfluss der Testkabel zu minimieren. Die Steckverbinder J4 (High Drive) und J7 (Low Drive) versorgen das DUT (Device Under Test) mit Strom, während die Steckverbinder J5 (High Sense) und J6 (Low Sense) die Messung der Spannung so nahe wie möglich am Bauteil ermöglichen. Wenn eine Vorspannung angelegt wird, liegen die positiven Anschlüsse an J4/J5.

Die PhotoMOS-Solid-State-Relais U54...U57, die die Operationsverstärker U9...U12 verbinden, weisen einen ON-Widerstand von etwa 1Ω auf. Die Spannung an den DUT-Anschlüssen muss differentiell und nicht relativ zur Masse gemessen werden. Dies ist die Aufgabe des Differenzverstärkers, der mit U7A, U7B, U7C aufgebaut ist. Das Gleichaktunterdrückungsverhältnis (CMRR) muss über den gesamten Frequenzbereich des LCR-Messgeräts hoch sein. Daher ist eine Anpassung erforderlich, zunächst durch R31 für niedrige Frequenzen (10 kHz) und C44 für hohe Frequenzen (1 MHz). Der mit U27B aufgebaute Integrator ermöglicht einen automatischen Offset-Abgleich.

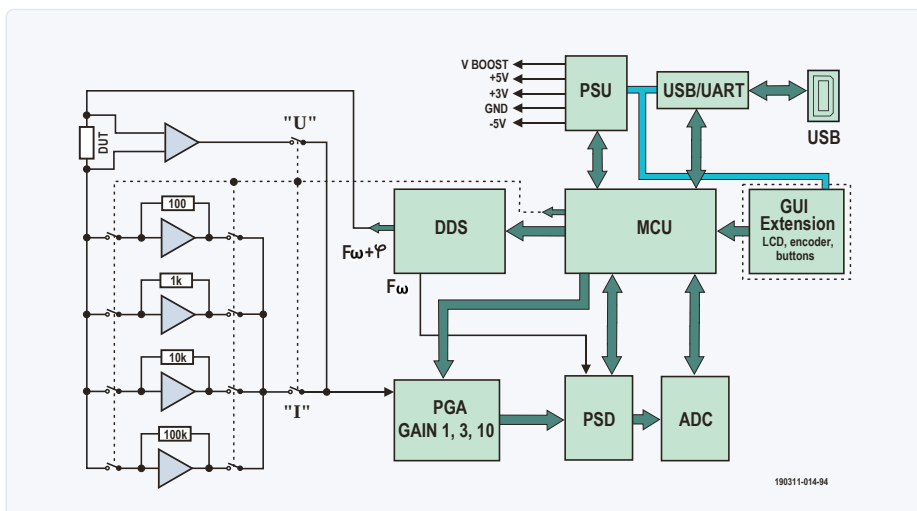


Bild 3. Die verschiedenen Teile des LCR-Messgeräts und wie sie miteinander verbunden sind.

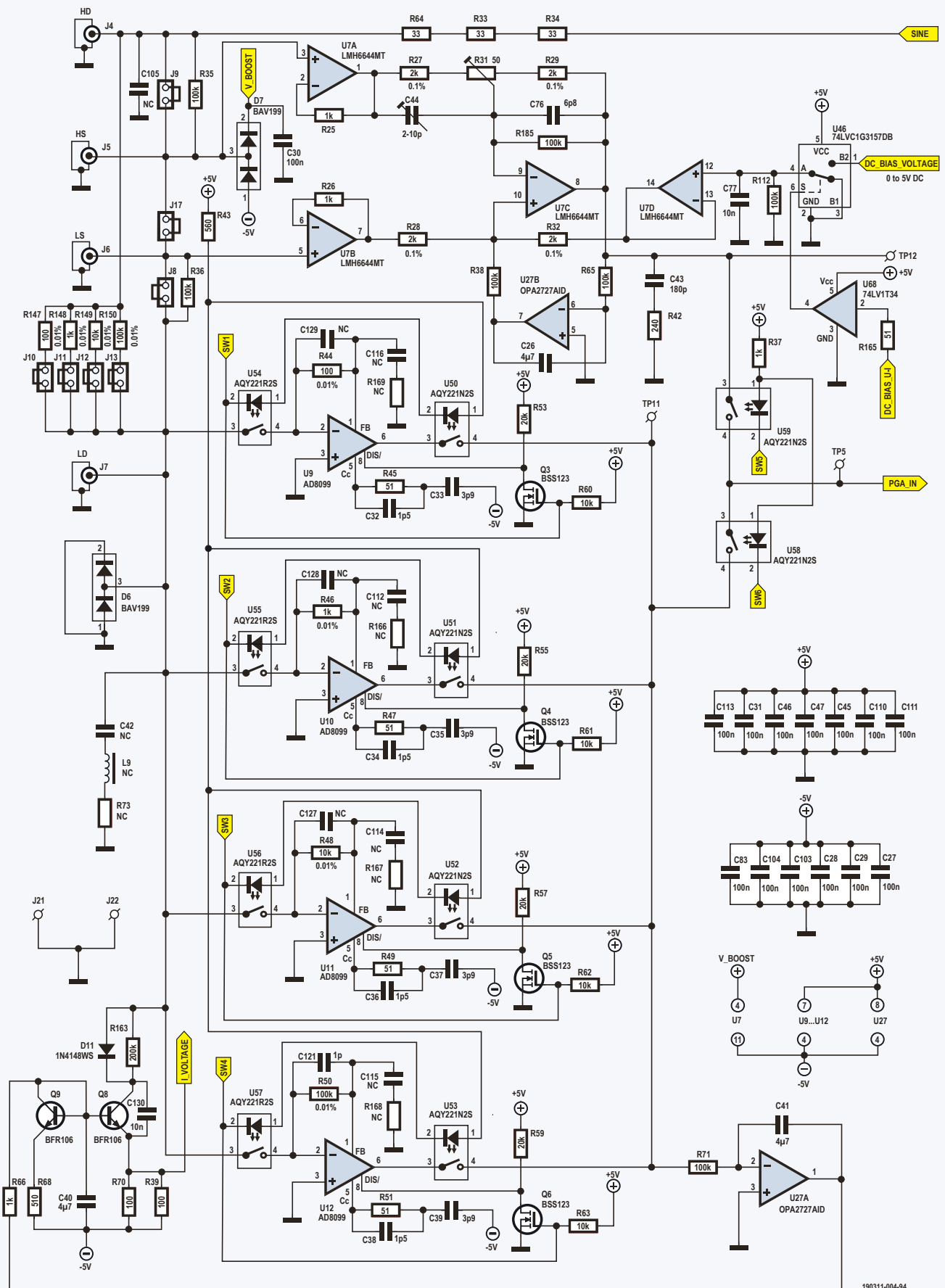


Bild 4. Die analogen Eingangsstufen. Der Mikrocontroller kann vier Impedanzbereiche wählen.

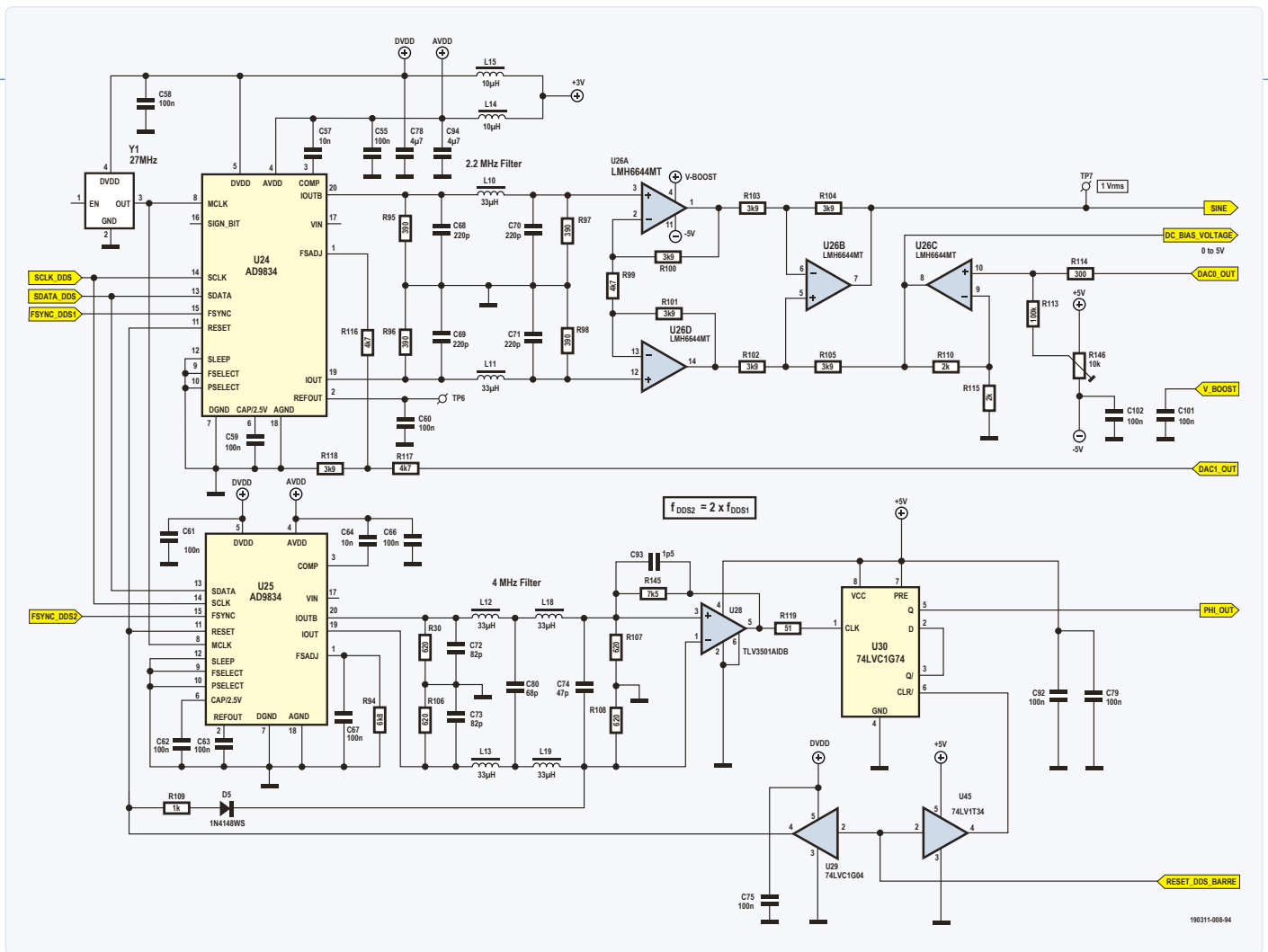


Bild 5. Zwei DDS-ICs erzeugen die Testsignale.

Je nach Messbereich wird einer der Operationsverstärker U9...U12 in die Messschaltung eingebunden. Dazu werden nicht nur die zugehörigen PhotoMOS-Relais am jeweiligen Eingang (U54...U57) und Ausgang (U50...U53) der Brückenschaltung durch den Controller (Signale SW1...SW4) eingeschaltet, sondern auch der entsprechende DIS/-Pin auf logisch high gelegt.

Die beiden Signale SW5 und SW6, die die Relais U58 respektive U59 aktivieren, schalten entweder die U7C-Ausgangsspannung (DUT-Spannung) oder die I-U-Wandler-Ausgangsspannung (DUT-Strom) zur Weiterverarbeitung auf den Ausgang dieser Funktionsgruppe (PGA_IN).

Das Anlegen der Messgleichspannung an das Messobjekt erfolgt am Sinusgenerator durch die Verschiebung seines Offsets im Bereich 0...5 V. Da ein Widerstand von 100 Ω (R33+R34+R64) in Reihe mit dem Ausgang dieses Generators geschaltet ist, entspricht dies einem Strom von 0...50 mA durch ein (induktives) DUT mit niedrigem Serienwiderstand. Für eine Spannung an einem Kondensator ermöglicht U7D eine Vorkompensation am Spannungsdifferenzver-

stärker. Für den Strom durch eine Induktivität sorgt die von Q8 und Q9 gebildete Gleichstromquelle. Der mit U27A aufgebaute Integrator gewährleistet dabei eine Gleichspannungslage des I-U-Wandlers nahe 0 V.

Um Kalibrierungen vorzunehmen, können Präzisionswiderstände, die mit denen in den I-U-Wandlern identisch sind, mit einem Jumper (J10 bis J13) verbunden werden.

Sinusgenerator (Bild 5)

Der Generator ist mit dem DDS-IC U24, einem AD9834 von Analog Devices, aufgebaut. Das 27-MHz-Taktsignal für das IC wird von Oszillator Y1 geliefert.

Das IC erzeugt zwei gegenphasige Ausgangssignale, die von U26A, U26B und U26D gefiltert und verstärkt werden. An Trimpoti R146 kann mit Opamp U26C ein Offset des Sinussignals (gemessen an TP7) eingestellt werden. Der verwendete Phasendetektor benötigt ein perfekt rechteckiges Signal mit genau der gleichen Frequenz wie der Generator, aber mit einer variablen relativen Phase. Dieses Signal wird durch U25 (ebenfalls ein DDS AD9834) erreicht. Der DDS-Generator erzeugt zunächst

ein Signal mit doppelter Frequenz, das einen schnellen Komparator (U28) passiert und auf ein Flipflop (U30) trifft, das die Signalfrequenz durch zwei teilt.

PGA (Programmable Gain Amplifier) (Bild 6)

Eine erste Verstärkerstufe, bestehend aus Opamp U19 und Multiplexer U23, besitzt eine vom Controller (PGA1_A und PGA1_B) wählbare Verstärkung von 1, 2, 5 oder 10, die die Verringerung des Nutzsignals durch eine abnehmende Amplitude des Testsignals kompensiert. Da die Verstärkung während eines Messzyklus nicht verändert wird, ist ihr Frequenzgang nicht sehr wichtig.

Die folgenden drei Opamps U20...U22, die von den Controllersignalen PGA_G1, PGA_G3 und PCA_G10 aktiviert werden, sorgen für eine zusätzliche Verstärkung von 1 (also keine), 3,17 und 10,1. U21 wird durch C108 einstellbar frequenzkompensiert, U22 durch C51. Das Kalibrierverfahren berücksichtigt das tatsächliche Ansprechverhalten dieses PGA für jede der verwendeten Frequenzen.

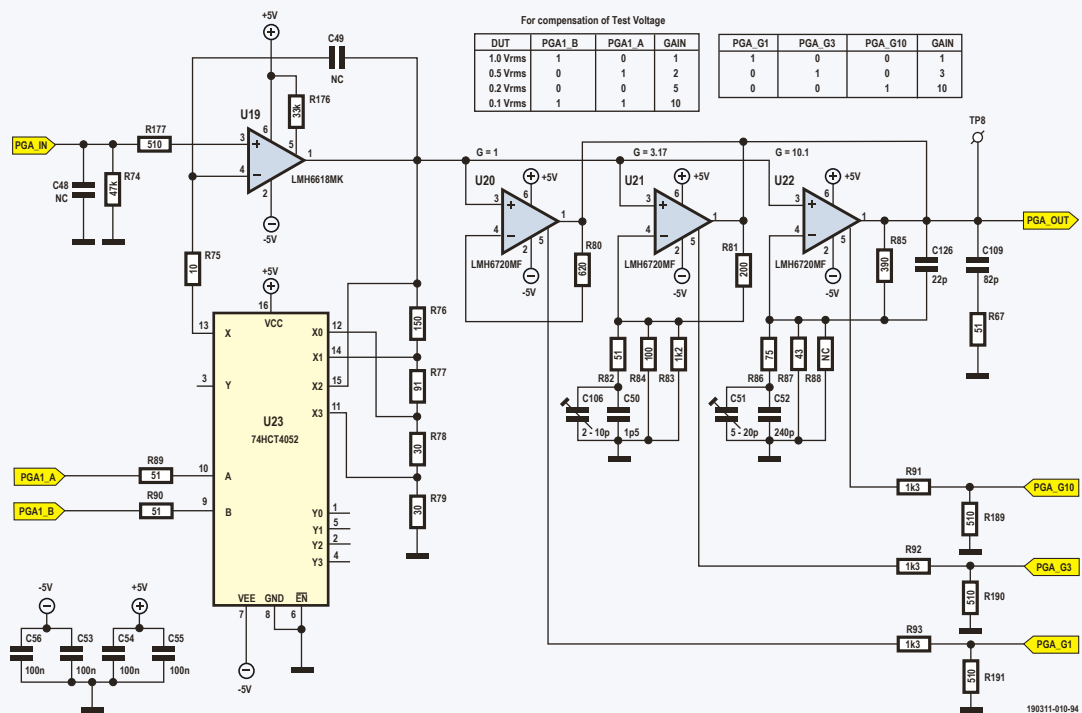


Bild 6. Programmable Gain Amplifier und Frequenzkompensation.

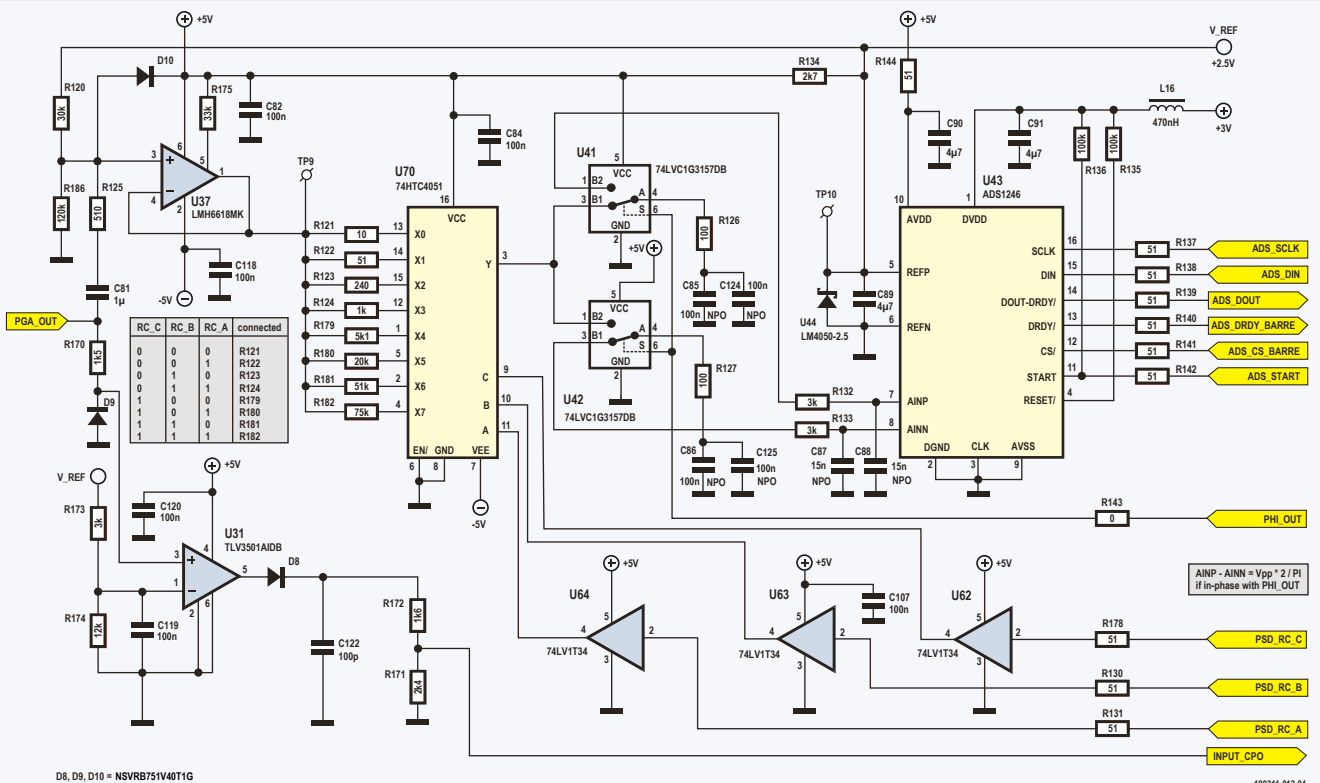


Bild 7. Der phasensensitive Detektor.

PSD (Phasensensitiver Detektor) (Bild 7)
Diese Schaltung ist mit den beiden alternierend arbeitenden Analogschaltern U41 und U42 aufgebaut. Sie verbinden für eine halbe Periode einen Speicherkondensator und

während der anderen halben Periode einen Integrationskondensator mit dem Eingangssignal. Damit wird eine doppelwertige Differenzspannung für den Analog-Digital-Wandler U43 erzeugt.

Die relative Phase des Schaltsignals gegenüber der Sinuswelle des Generators ermöglicht es, Messungen der Phasen- oder Quadraturkomponenten des Eingangssignals des PSDs durchzuführen.

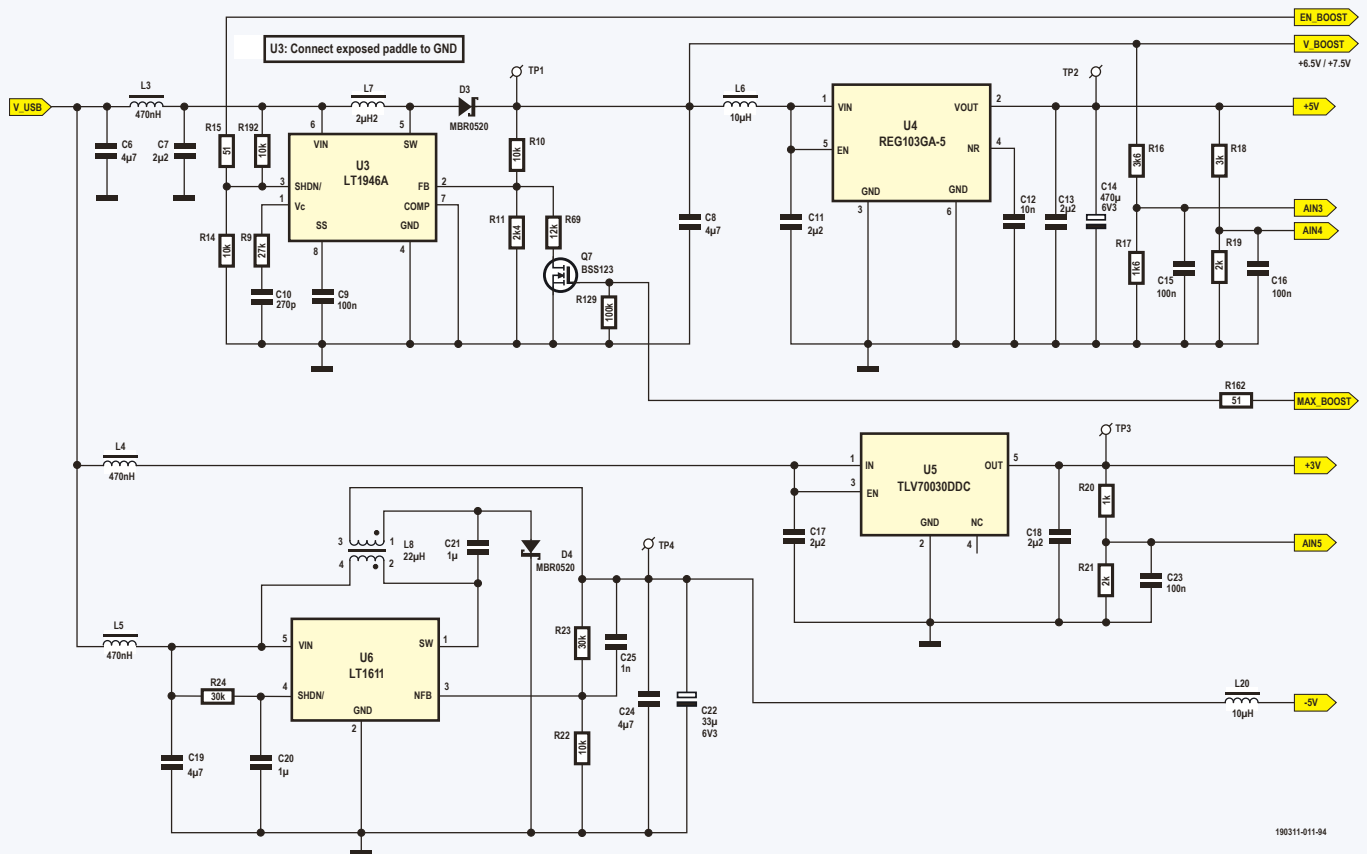


Bild 10. Auf der Hauptplatine befinden sich vier Spannungsregler.

Die LED D12 kommt beim Debugging während der Entwicklung der Firmware (zum Beispiel beim Drücken eines Knopfes oder beim Betrieb des Encoders) und auch beim normalen Betrieb zum Einsatz, um bestimmte Fehlerzustände zu signalisieren:

- Keine angeschlossene GUI-Platine an J14 und keine Kommunikation über USB: permanentes Blinken (je 0,5 s an und aus).
- Power-Test-Fehler: Zusätzlich zur Meldung inklusive der Fehlernummer auf dem Display permanentes Blinken (0,5 s an und 1 s aus).

Die ICs U66 und U67 ermöglichen die Synchronisation mit dem Sinussignal für die Spannungs- und Strommessungen am Prüfling. Ein Jumper an Header J16 informiert den Bootloader, dass ein Firmware-Update vorgenommen werden soll. Drucktaste K1 (RESET) ist optional und wird nur während der Entwicklungsphase verwendet.

USB-Controller und Spannungsversorgung(en) (Bild 9)

Ein FT232RL (U2) von FTDI dient als USB-/UART-Schnittstelle zwischen PC und Controller. Normalerweise erfolgt auch die Stromver-

sorgung über die USB-Buchse J1 (Jumper J3 auf 2-3), aber es ist auch möglich, die Schaltung über den Verbinder J2 zu versorgen. In diesem Fall muss Jumper J3 auf 1-2 gesetzt werden.

Für die Stromversorgung aller Schaltungsteile sind vier Spannungen erforderlich (**Bild 10**):

- Eine V_BOOST-Spannung von +6,5 V oder +7,5 V (abhängig von der Steuerungsspannung MAX_BOOST), die vom Aufwärtsregler U3 und L7/D3 bereitgestellt wird.
- Eine Spannung von +5 V, die vom Linearregler U4 bereitgestellt wird.
- Eine Spannung von +3 V, die vom Linearregler U5 bereitgestellt wird.
- Eine Spannung von -5 V, die von dem Inverter-Regler U6 und L8/D4 geliefert wird.

All diese Spannungen werden vom Programm beim Einschalten mit dem 12-Bit-ADC und dem im Controller integrierten Multiplexer verifiziert.

GUI/Display-Platine

Ein Flachbandkabel an J1 verbindet die GUI/Display-Platine in **Bild 11** mit der Hauptplatine. Fünf Drucktasten K1...K5 und der Drehko-

dierschalter SW1 in einer Matrix benötigen nur drei Portleitungen des Controllers zur Dekodierung. Für den Drehgeber sind zwei weitere Portleitungen erforderlich.

RC-Filter dämpfen das Tastenprellen, so dass die Debouncing-Maßnahmen auf Softwareseite relativ einfach gehalten werden konnten. Über die Dioden D12 und D13 wird bei einem Tastendruck eine Interrupt-Routine des Controllers ausgelöst.

Das Grafikdisplay U1 wird über den +3-V-Linearregler U2 aus der gefilterten USB-Spannung versorgt. Die Hintergrundbeleuchtung wird über den Transistor Q1 gesteuert, während der Transistor Q2 beim Abschalten des Geräts für eine schnelle Entladung der intern erzeugten VLCD-Spannung sorgt, wodurch ein unangenehmes „Nachleuchten“ vermieden wird. Diese Anzeigeplatine wird während des Hochfahrens des Messgeräts durch den Widerstand von R7+R9 nach Masse über Pin 19 (BKL) von J1 vom Controller identifiziert.

In Vorbereitung: Bluetooth Low Energy

Eine Erweiterungsplatine mit einem BLE-Modul (**Bild 12**) ist in Entwicklung. Es wird über Pin 6 (CONFIG2) von J1 durch den Widerstand R32 von 1 kΩ gegen Masse identifiziert.

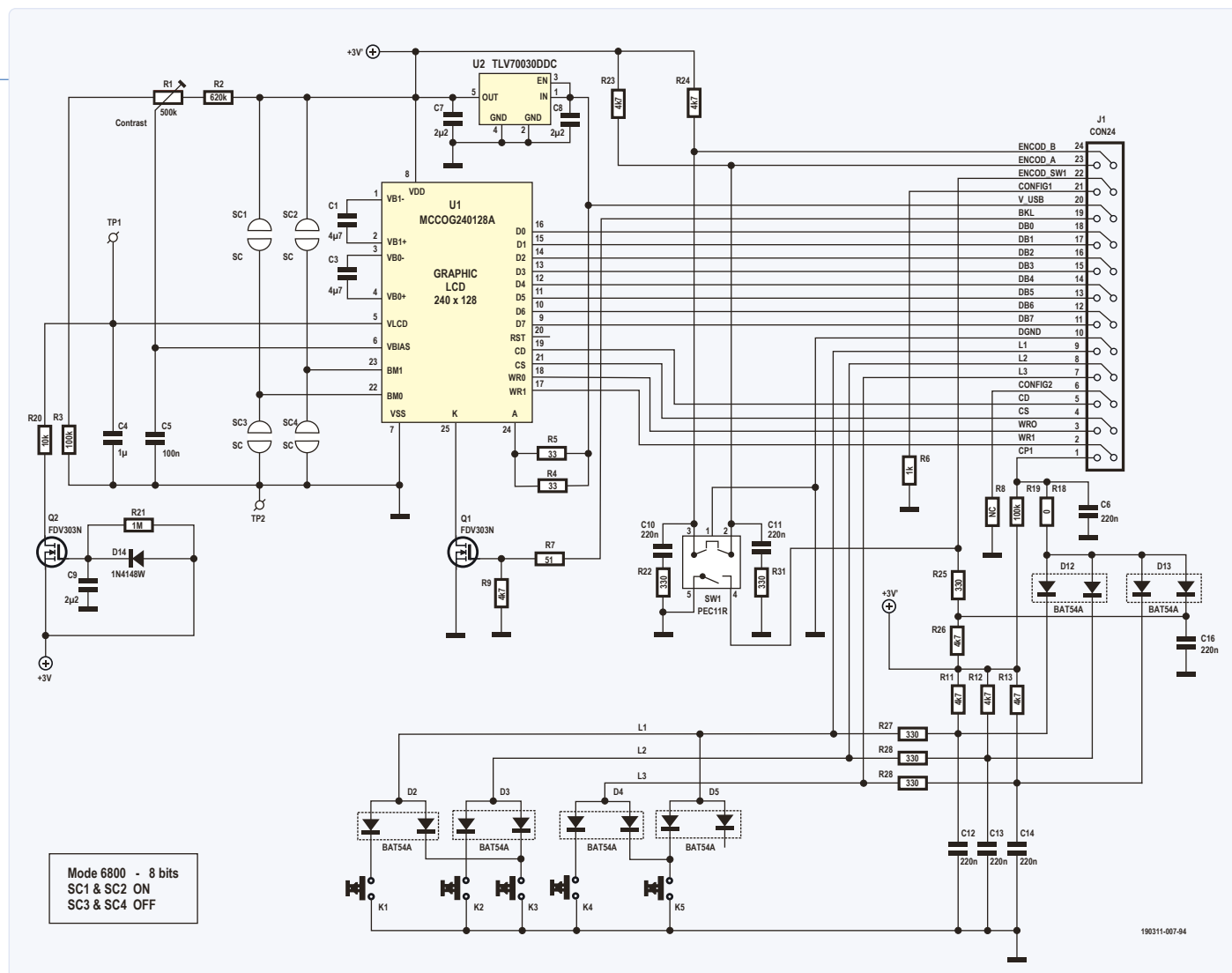


Bild 11. Die Platine mit Tasten, LCD und Drehgeber ist direkt hinter der Frontplatte angebracht.

Kalibrierung in der Software

Die selbstabgleichende Brücke hat, wenn sie mit einem einfachen Opamp (A2) aufgebaut ist, bei hohen Frequenzen einen erheblichen Nachteil: Die virtuelle Masse (Spannung V3 in **Bild 13**) kann aufgrund des nach oben beschränkten Frequenzbereichs des Verstärkers nicht auf Null gehalten werden.

Die meisten einfachen Berechnungen zur Kompensation der parasitären Elemente im Leerlauf/Kurzschluss-Abgleich (OPEN/SHORT), die davon ausgehen, dass die Spannung V3 eine virtuelle Masse ist, sind daher bei hohen Frequenzen nicht mehr anwendbar. Statt dessen müssen ausgefeiltere Kompensationsmethoden wie der Leerlauf/Kurzschluss/Last-Abgleich (OPEN/SHORT/LOAD) verwendet werden.

Für mehr Infos zu diesem Thema lesen Sie bitte den Artikel „Test Fixture Compensation Techniques in Impedance Analysis“ [4]. Die Kalibrierung des Geräts wird in mehreren Schritten durchgeführt:

- Für jede vordefinierte Frequenz und für

PGA-Verstärkungen von 3 und 10.

- LOAD, für jeden der vier Bereiche mit $Z_x = R_{ref}$ für jede vordefinierte Frequenz.
- OPEN TRIM (ohne angeschlossenes DUT), für jede vordefinierte Frequenz.
- SHORT TRIM (Eingang kurzgeschlossen), für jede vordefinierte Frequenz.

Die gespeicherten Werte werden zur Berechnung der Korrekturen der rohen Messwerte verwendet. Nach Anpassung der gemessenen DUT-Werte zur Einstellung der Verstärkung von A3 (komplexer Wert) wird sowohl für die Strom- als auch für die Spannungsmessung der Endwert nach folgender Formel berechnet:

$$Z_x = R_{ref} \frac{(Z_{xm} - Z_{short})(Z_{open} - Z_{stdm})}{(Z_{stdm} - Z_{short})(Z_{open} - Z_{xm})}$$

All diese Werte sind komplex:

- Z_{xm} ist der gemessene Wert.
- Z_{stdm} ist der gemessene Wert von R_{ref} während der LOAD-Phase.
- Z_{short} ist der während der SHORT-TRIM-Phase gemessene Wert.

- Zopen ist der während der OPEN-TRIM-Phase gemessene Wert.

Dabei ist es möglich, die beiden TRIM-Operationen (TRIM-Taste) mit der aktuellen Frequenz zu wiederholen, wenn sich beispielsweise der Messkopf geändert hat.

Bei der Wahl einer anderen Frequenz werden die Korrekturen durch Interpolation aus den zwei nahegelegenen vordefinierten Frequenzen vorgenommen.

Die Platinen

Um es vorweg zu sagen: Das Projekt ist nur sehr erfahrenen Lötprofis mit den richtigen Fähigkeiten und Werkzeugen zu empfehlen. Zwar lässt sich mit Geduld und einer sehr ruhigen Hand, einer Lupe oder besser einem (Stereo-)Mikroskop die Arbeit auch mit einem sehr feinen Lötcolben erledigen, ideal ist das wegen der Bauteile mit nicht mit dem Lötcolben zugänglichen „exposed pads“ sicherlich nicht. Besser, man verwendet Lötpaste mit einem geeigneten Dispen-

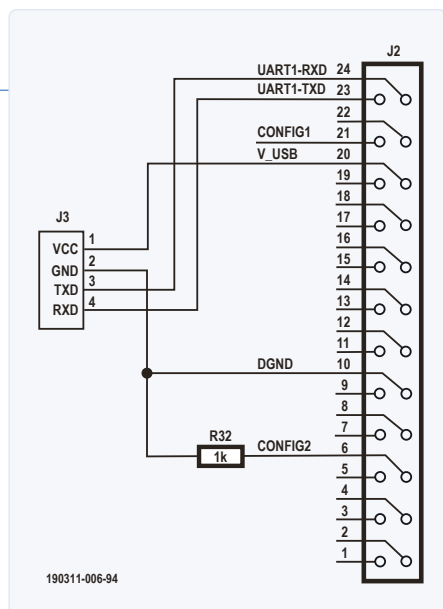


Bild 12. Zur Steuerung des LCR-Meters kann eine Bluetooth-Schnittstelle verwendet werden.

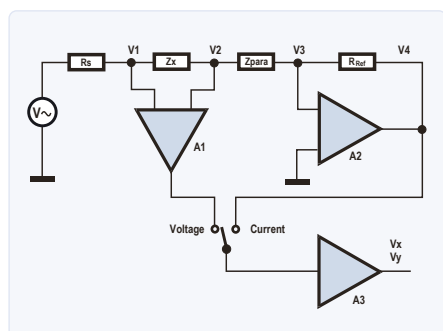
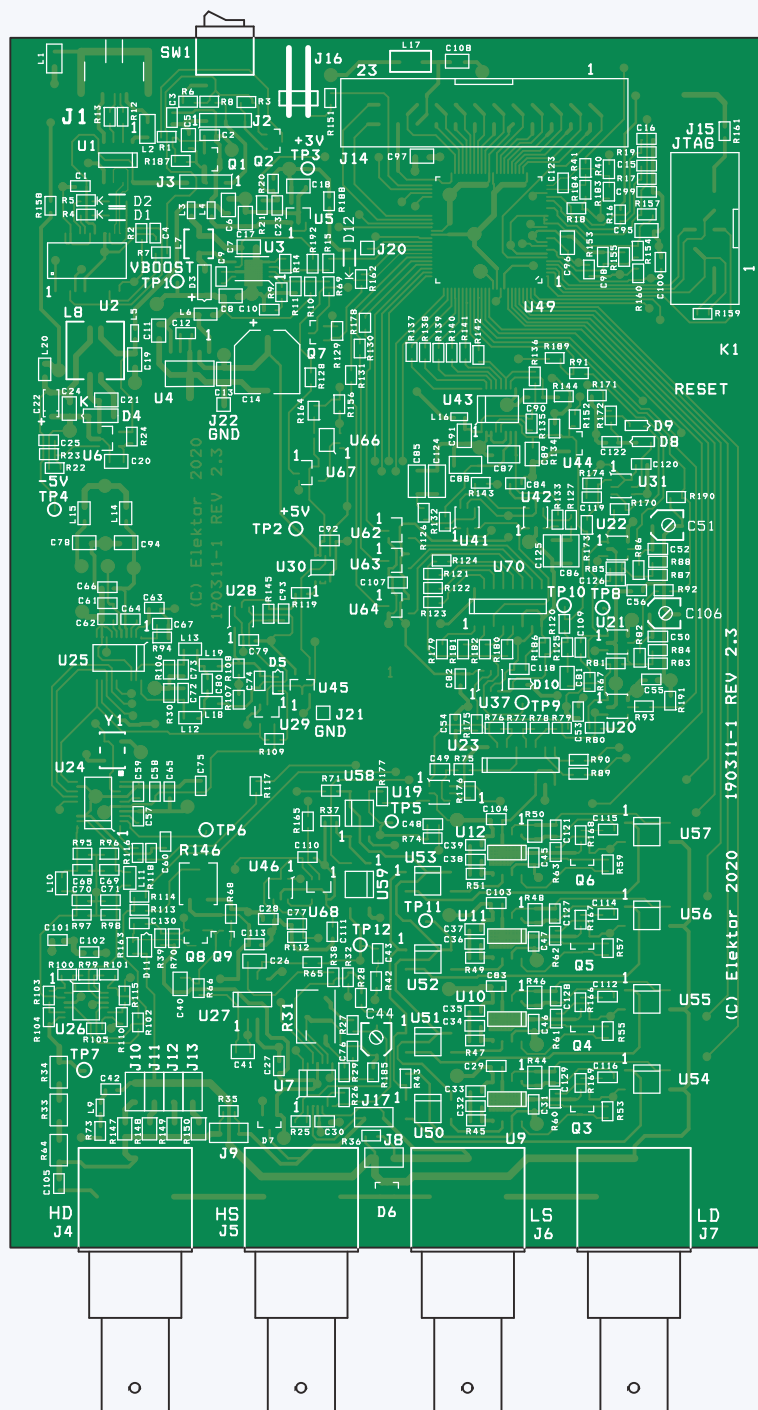


Bild 13. Vereinfachtes Diagramm der Messschaltung.

ser oder Schablonen und ein Heißluft-Lötwerkzeug oder einen Reflow-Ofen. Wenn Sie irgendwelche Zweifel haben, diese Arbeit erledigen zu können, lassen Sie es besser sein!

Die Platinenlayouts (**Bild 14**) inklusive Gerber-Dateien und auch Bohr- und Fräszeichnungen für das Hammond-Gehäuse, das wir für dieses Projekt verwendet haben, stehen auf der Elektor-Website [3] zum Download zur Verfügung. Angesichts der Komplexität der Platinen - die Hauptplatine hat vier Kupferlagen! - sind sie zum Selbstätzen wenig geeignet. Besser, man überlässt die Produktion einem professionellen Platinenhersteller!

In diesem Moment läuft auf der Elektor-Labs-Seite [7] eine Kickstarter-Kampagne, bei der bereits 40% der angestrebten 150 potentiellen Käufer daran interessiert sind, vorbestückte Platinen für dieses Projekt zu erwerben (Bild 14d). Verpassen Sie nicht Ihre Chance, mitzumachen!





STÜCKLISTE HAUPTPLATINE

Widerstände:

(alle SMD 0603, 1%, 100 ppm, wenn nicht anders angegeben)

R1,R4,R5,R8,R20,R25,R26,R37,R66,R109,R124,
R164,R188 = 1 k
R2,R7,R10,R14,R22,R60...R63,R151,R187,
R192 = 10 k
R3,R35,R36,R38,R65,R71,R112,R113,R129,R135,
R136,R160,R183...R185 = 100 k
R6,R15,R45,R47,R49,R51,R67,R82,R89,R90,R119,R1
22,R128,R130,R131,R137...142,R144,R162,
R165,R178 = 51 Ω
R9 = 27 k
R11,R171 = 2k4
R12,R13 = 27 Ω
R16 = 3k6
R17,R172 = 1k6
R18,R132,R133,R173 = 3 k
R19,R21,R110,R115 = 2 k
R23,R24,R120 = 30 k
R27...R29,R32 = 2 k, 0,1% 25 ppm
R30,R80,R106...R108 = 620 Ω
R31 = Trimmpoti 50 Ω (Bourns 3266W)
R33,R34,R64 = 33 Ω, 100 ppm 1206
R39,R70,R84,R126,R127 = 100 Ω
R40,R41 = 110 k
R42,R123 = 240 Ω
R43 = 560 Ω
R44,R147 = 100 Ω 0,01%, 50 ppm, 0805
R46,R148 = 1 k, 0,01%, 10 ppm, 0805
R48,R149 = 10 k, 0,01%, 10 ppm, 0805
R50,R150 = 100 k, 0,01%, 10 ppm, 0805
R53,R55,R57,R59,R180 = 20 k
R68,R125,R177,R189...R191 = 510 Ω
R69,R174 = 12 k
R73,R88,R166...R169 = n.c.
R74 = 47 k
R75,R121,R153,R161 = 10 Ω
R76 = 150 Ω
R77 = 91 Ω
R78,R79 = 30 Ω
R81 = 200 Ω
R143 = 0 Ω
R83 = 1k2
R85,R95...R98 = 390 Ω
R86 = 75 Ω
R87 = 43 Ω
R91...R93 = 1k3
R94 = 6k8
R99,R116,R117,R154,R155,R159 = 4k7
R100...R105,R118 = 3k9
R114 = 300 Ω
R134 = 2k7
R145 = 7k5
R146 = Trimmpoti 10 k (Vishay TS53YJ)
R152 = 47 Ω
R156,R170 = 1k5k
R157 = 56 k
R158 = 160 k
R163 = 200 k
R175,R176 = 33 k
R179 = 5k1k
R181 = 51 k

R182 = 75 k
R186 = 120 k

Induktivitäten:

L1,L2 = Ferritperle (Murata BLM21AG102SN1D)
L3,L4,L5,L16 = 470 n, TDK MLZ1608DR47DT
L6,L14,L15,L20 = 10 µ, TDK MLZ2012M100WT
L7 = 2µ2 (Coilcraft 1008PS-222KLB)
L8 = 22 µ (Cooper Bussmann DRQ73)
L9 = n.c.
L10,L11,L12,L13,L18,L19 = 33 µ (TDK
MLZ2012M330WT)
L17 = 22 µ (Taiyo Yuden CBC3225T220MR)

Kondensatoren:

(0603, falls nicht anders angegeben)
C1...C4,C9,C15,C16,C23,C27...C31,C45...C47,C53...
C56,C58...C63,C65...C67, C75,C79,C82...
C84,C92,C98...C104,C107,C110,C111,C113,C118...
C120,C123 = 100 n, 50 V, 10%, X7R
C5,C6,C8,C19,C24,C26,C40,C41,C78,C89...
C91,C94,C108 = 4µ7, 10 V 10%, X7R, 0805
C7,C11,C13,C17,C18 = 2µ2, 10 V, 10%, X7R, 0805
C10 = 270 p, 50 V, 5%, NP0
C12,C57,C64,C77, C130 = 10 n, 50 V, 10%, X7R
C14 = 470 µ, 6V3, Alu-Elko
(Panasonic EEEFP0J471AP)
C20,C21,C81,C95,C96,C97 = 1 µ, 25 V, 10%, X7R,
0805
C22 = 33 µ, 6V3, CASE-A (Tantal)
C25 = 1 n, 50 V, 5%, NP0, 0603
C32,C34,C36,C38,C50,C93 = 1p5, 50 V, 0p25,
NP0
C33,C35,C37,C39 = 3p9, 50 V, 0p25, NP0
C42,C48,C49,C105,C112,C114...C116,
C127...C129 = n.c.
C43 = 180 p, 50 V, 5%, NP0
C44,C106 = Trimmer 2-10 p
(Knowless Voltronics JZ100)
C51 = Trimmer 4,5-20 p
(Knowless Voltronics JZ200)
C52 = 240 p, 50 V, 5%, NP0
C68,C69,C70,C71 = 220 p, 50 V, 5%, NP0
C72,C73,C109 = 82 p, 50 V, 5%, NP0
C74 = 47 p, 50 V, 5%, NP0
C76 = 6,8 p, 50 V, 0p25, NP0
C80 = 68 p, 50 V, 5%, NP0
C85,C86,C124,C125 = 100 n, 50 V, 5%, NP0, 1206
C87,C88 = 15 n, 50 V, 5%, NP0, 1206
C121 = 1 p, 50 V, 0p25, NP0, 0603
C122 = 100 p, 50 V, 5%, NP0, 0603
C126 = 22 p, 50 V, 5%, NP0, 0603
C112, C114, C115, C116, C127, C128, C19 = n.c.

Halbleiter:

D1,D2,D12 = LED rot, low power, 0805
D3,D4 = MBR0520 (Schottky-Diode)
D5,D11 = 1N4148WS
D6,D7 = BAV199 LT1G (Doppeldiode)
D8,D9,D10 = NSVRB751V40T1G
(Schottky-Diode)
Q1 = IRLML6402 (P-Kanal-MOSFET)
Q2...Q7 = BSS123 (N-Kanal-MOSFET)

Q8,Q9 = BFR106 (HF-NPN)
U1 = USB6B1 (ESD-Schutz)
U2 = FT232RL (USB zu UART)
U3 = LT1946A (DC/DC-Schaltregler)
U4 = REG103GA-5 (Linearregler, 5 V, 500 mA)
U5 = TLV70030DDCR (Linearregler, 3 V,
200 mA)
U6 = LM2611AMF (Abwärtswandler)
U7,U26 = LMH6644MT (Vierfach-Opamp)
U9...U12 = AD8099ARDZ (Opamp)
U19,U37 = LMH6618MK (Opamp SOT23-6)
U20...U22 = LMH6720MF (Opamp SOT23-6)
U23 = 74HCT4052D (Analog-Multiplexer)
U24,U25 = AD9834BRUZ (DDS-IC)
U27 = OPA2727AID (Präzisions-Opamp)
U28,U31 = TLV3501AIDB (Analog-Komparator)
U29 = 74LVC1G04DBVT (Inverter)
U30,U66 = 74LVC1G74DCTR (Flipflop)
U41,U42,U46 = 74LVC1G3157DBVR
(Analogschalter)
U43 = ADS1246IPW (24-Bit-ADC)
U44 = LM4050CIM3-2.5
(Spannungsreferenz 2,5 V)
U45,U62,U63,U64,U67,U68 = 74LV1T34DBVR
(Puffer)
U49 = C8051F120GQ (8-Bit-MCU, Silicon Labs)
U50,U51,U52,U53,U58,U59 = AQY221N2S
(Halbleiterrelais)
U54,U55,U56,U57 = AQY221R2S
(Halbleiterrelais)
U70 = 74HCT4051D (Analog-Multiplexer)

Außerdem:

J1 = USB_B_Mini-Buchse
(TE Connectivity 1734035-2)
J2,J3 = 1x3-polige Stiftleiste (Raster 2,54 mm)
J4,J5,J6,J7 = BNC-Buchse für Platinenmontage
(TE Connectivity 1-1337543-0)
J8,J9,J17,J10,J11,J12,J13 = 1x2-polige Stiftleiste
(Raster 2,54 mm)
J16 = 1x2-polige Stiftleiste, gewinkelt
(Raster 2,54 mm)
J14 = 2x12-polige Stiftleiste mit Box
J15 = 2x5-polige Stiftleiste mit Box
J20,J21,J22 = Testpad
K1 = 6-mm-Drucktaster, optional
(Omron BF3-1020)
SW1 = Kippschalter
(Nidec Copal AS1D-5M-10-Z)
Y1 = 27-MHz-Oszillator
(Epson X1G0044510005 SG5032CAN
27 MHZ TJGA)
Platine 190311-1 V2.3

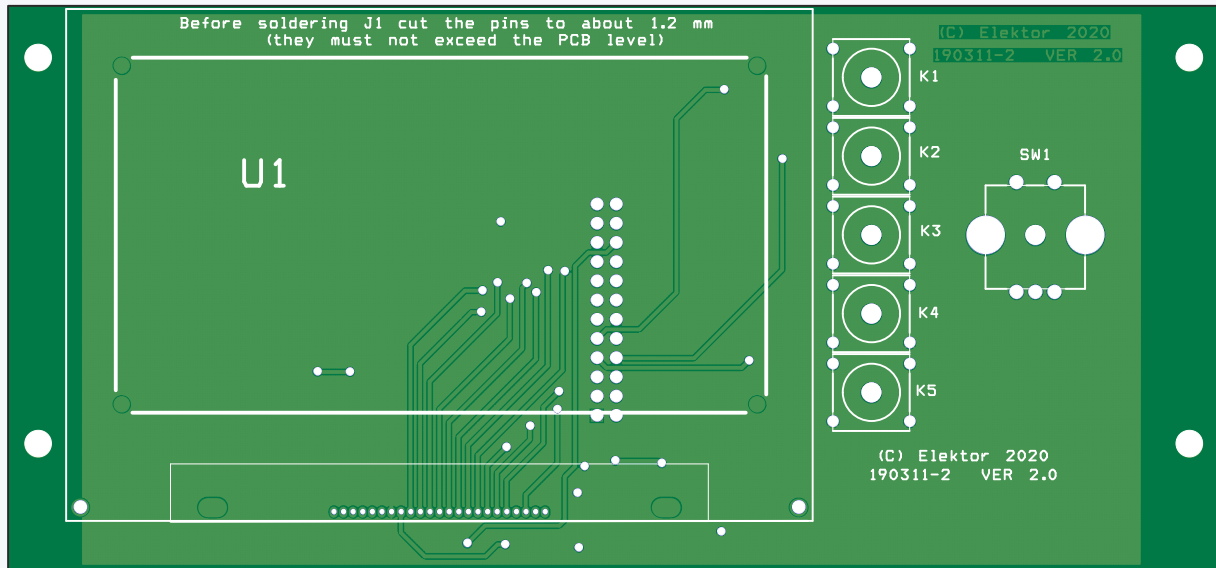
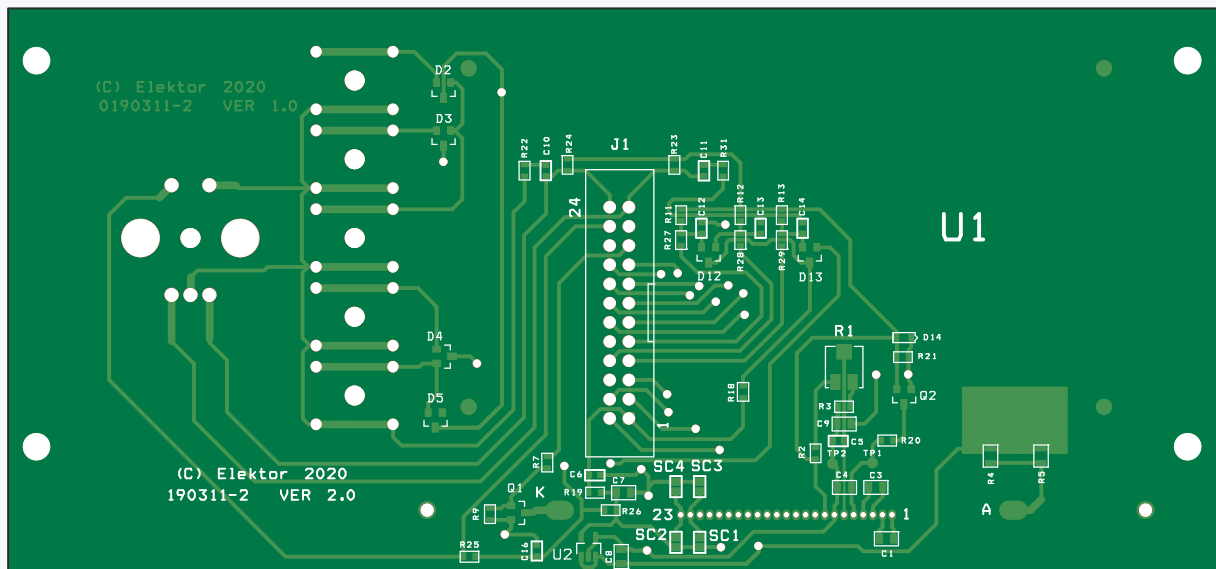


Bild 14b. Layout der Frontplattenplatine, Unterseite. . ▲

Bild 14c. Layout der Frontplattenplatine, Oberseite. ▼



gerät. Achten Sie darauf, dass die HIGH- und LOW-Markierungen am „Test Fixture TH26001A“ gegenüber dem LCR-Meter vertauscht sind!

Es ist die Impedanz Z des Bauteils, die den Messbereich (Wahl des I/U-Wandlers) und die (De-)Aktivierung des PGA bedingt. Der Fehler beträgt maximal 0,1% für $50 \Omega < Z < 330 \text{ k}\Omega$

und bei Frequenzen $< 100 \text{ kHz}$. Für extremere Werte beträgt der Fehler maximal etwa 1%. Bei Frequenzen $> 1 \text{ MHz}$ nimmt der Fehler etwas zu und kann bis 3% anwachsen.

Genauigkeit und Fehler

Die erzielbare Genauigkeit hängt hauptsächlich ab von

- dem Typ des zu prüfenden Bauteils (L, C oder R) und seinem Wert,
- der Prüffrequenz, die für dieses Bauteil angemessen sein muss,
- dem Messkopf (Kelvin-Clips, Test Fixture...), wiederum abhängig von der Testfrequenz und
- der Sorgfalt bei der Kalibrierung (neu vorzunehmen, wenn sich die Umgebungstemperatur signifikant geändert hat).

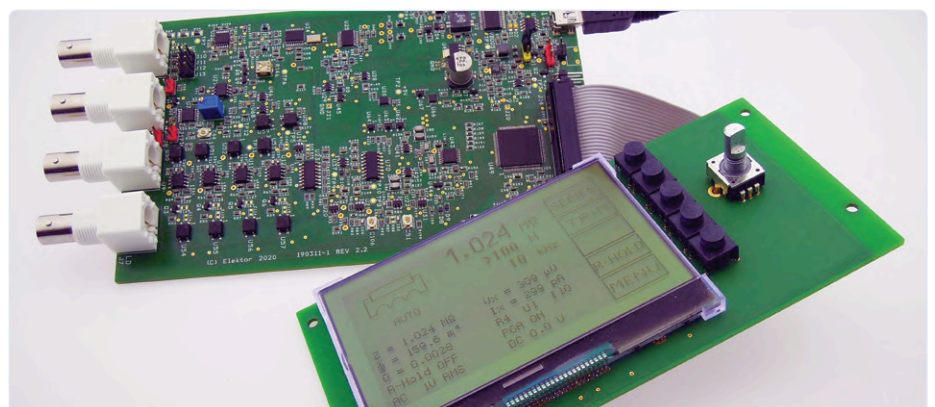


Bild 14d. Haupt- und Frontplattenplatine sind zum Testen miteinander verbunden.



STÜCKLISTE GUI/DISPLAYPLATINE UND GEHÄUSE

Widerstände:

(SMD 0603, wenn nicht anders angegeben)

R1 = 500 k
R2 = 620 k
R3, R19 = 100 k
R4, R5 = 33 Ω, 0805
R7 = 51 Ω
R6, R8 = n.c.
R9, R11...R13, R23, R24, R26 = 4k7
R18 = 0 Ω
R20 = 10 k
R21 = 1 M
R22, R25, R27, R28, R29, R31 = 330 Ω

Kondensatoren:

C1, C3 = 4,7 μ, 10 V, 10%, X7R, 0805
C4 = 1 μ, 25 V, 10%, X7R, 0805
C5 = 100 n, 50 V, 10%, X7R, 0603

C6, C10...C14, C16 = 220 n, 50 V, 10%, X7R, 0603
C7...C9 = 2μ2, 10 V, 10%, X7R, 0805

Halbleiter:

D2...D5, D12, D13 = BAT54A SOT23 (Doppeldiode)
D14 = 1N4148WS, SOD323
Q1, Q2 = FDV303N, SOT23
(digitaler N-Kanal-MOSFET)
U1 = Display MCOG240128A6W-FPTLW
(Midas)
U2 = TLV70030DDCR, SOT23-5
(Linearregler, 3 V, 200 mA)

Außerdem:

J1 = 2x12-polige Stiftleiste mit Box,
(Amphenol, Serie T821)
SW1 = Drehgeber PEC11R-4115F-S0018 (Bourns)
Platine 190311-2 V2.0

Gehäuse:

Hammond 1455N1601
4 x Abstandhalter, Sechskant, Buchse/Buchse,
M3x7 mm
1 x Abstandhalter, Sechskant, Buchse/Buchse,
M3x6 mm
1 x dünne Unterlegscheibe, Stahl, 3 mm
5 x Maschinenschraube, Senkkopf M3 16 mm
5 x Unterlegscheibe, 3 mm
5 x Mutter M3
2 x 2x12-poliger IDC-Verbinder, female
(Amphenol T81214A101CEU)
24-poliges Flachbandkabel, ca. 15 cm
Drehknopf aus Aluminium für 6-mm-Welle
(z.B. Mentor 50761)
4 x Jumper, Raster 2,54 mm, vergoldet



Bild 15a. Ein Kelvin-Kabel, das man zu einem vernünftigen Preis erhalten kann.



Bild 15b. Der vierpolige Messkopf TH26001A: schön, aber teuer!

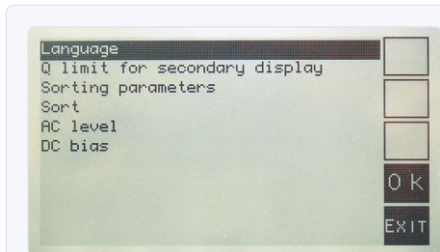
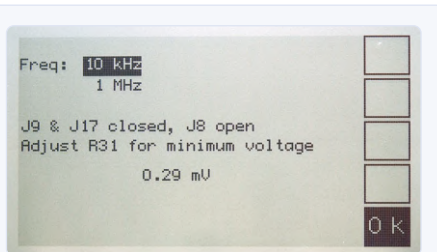


Bild 16. Zwei Screenshots des Kalibrierungsmenüs, dessen viele Optionen wir im zweiten Teil des Artikels kennenlernen werden. Die Einfachheit der Kalibrierung ist eines der Hauptvorteile dieses Geräts.



Fortsetzung folgt!

Im zweiten Teil dieses Artikels, der in der nächsten Elektor-Ausgabe veröffentlicht wird, werden die Hauptthemen sein:

- > Stand-alone-Modus und PC-Modus (bei dieser displaylosen Konfiguration kommunizieren Sie mit dem Gerät über den PC-Monitor).
- > Kalibrierung, erster Einsatz mit einem „blanken“ leeren Mikrocontroller und Software-Aktualisierungen.
- > Messungen von Widerstand, Kapazität und Induktivität.
- > Und natürlich die Beschreibung der Konfigurations- und Kalibrieremenüs (Bild 16). Die folgenden sechs Screenshots (Bild 17 und Bild 18) zeigen einige typische Messsituationen.
- > Wir wollen auch Fragen beantworten, die dieser erste Teil sicherlich aufwerfen wird.

Leser können bereits jetzt die Dokumentation (bestehend aus zwei Handbüchern) und alle Dateien im Zusammenhang mit diesem Projekt (einschließlich Gerber-Dateien) herunterladen.

In der nächsten Ausgabe nehmen wir auch Vergleichsmessungen des neuen AU2019-LCR mit Referenzgeräten wie HP 4263B-1, HP-4263B-2, Hameg HM8118 und Keysight (Agilent) U1732A vor. ◀

190311-03

PASSENDE PRODUKTE

- > **Elektor „Kickstarter Project“:**
www.elektor.com/lcr

Sie haben Fragen oder Kommentare?

Haben Sie Fragen oder Anmerkungen zu diesem Artikel? Schicken Sie dem Autor eine E-Mail an jjacques.aubry@free.fr.

Ein Beitrag von

Idee, Gestaltung, Text und Illustrationen:
Jean-Jacques Aubry

Schaltpläne: **Kurt Diedrich**

Messungen: **Alfred Rosenkränzer**

Redaktion: **Luc Lemmens, Denis Meyer**

Übersetzung: **Rolf Gerstendorf**

Gestaltung: **Giel Dols**

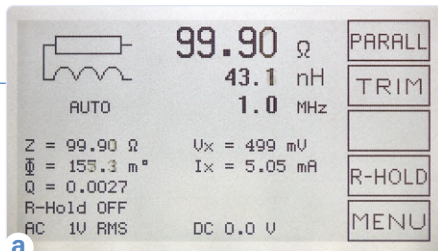


Bild 17a. Messung eines kleinen Widerstands (100 Ω), dessen parasitäre Induktivität (43,1 nH) bei hoher Messfrequenz offenbar wird!

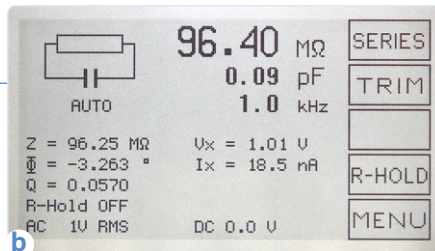


Bild 17b. Messung eines hochohmigen Widerstands (100 MΩ), dessen parasitäre Kapazität (0,09 pF) sich bei niedrigen Frequenzen zeigt!

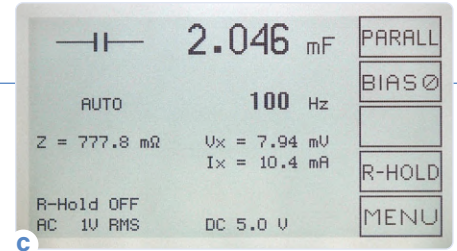


Bild 17c. Messung eines großen Kondensators bei niedriger Frequenz und 5 V Biasspannung.

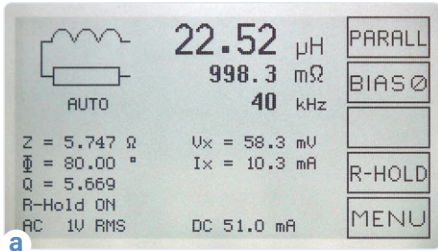


Bild 18a. Messung einer Induktivität mit Bias-Strom.

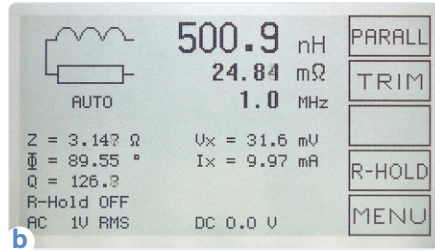


Bild 18b. Messung einer Induktivität bei höherer Frequenz.

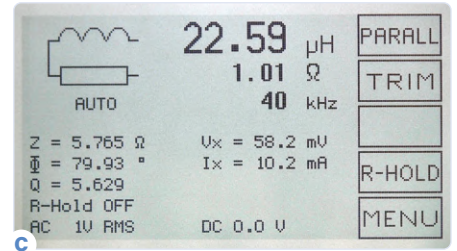


Bild 18c. Messung einer Induktivität ohne Bias-Strom.

WEBLINKS

- [1] Qt: <https://www.qt.io/>
- [2] The Impedance Measurement Handbook (2003 – Agilent Technologies Co. Ltd), Bild 3.1.4: https://wiki.epfl.ch/carplat/documents/rcl_agilent.pdf
- [3] Vollständiger Projektdownload (Gerber, drill&mill, BOM, ...): www.elektormagazine.de/190311-03
- [4] Test Fixture Compensation Techniques in Impedance Analysis: <http://www.eejournal.ktu.lt/index.php/elt/article/view/11206/5939>
- [5] Betriebshandbuch (aktuelle Version): www.elektormagazine.de/190311-03
- [6] Benutzerhandbuch (aktuelle Version): www.elektormagazine.de/190311-03
- [7] Neues LCR-Meter auf Elektor Labs: www.elektormagazine.com/labs/remake-lcr-meter
- [8] LCR-Meter 2013 (1) - März 2013, S.14: www.elektormagazine.de/magazine/elektor-201303/20363
- [9] LCR-Meter 2013 (2) - April 2013, S.34: www.elektormagazine.de/magazine/elektor-201304/20067
- [10] LCR-Meter 2013 (3) - Mai 2013, S.26: www.elektormagazine.de/magazine/elektor-201305/20575
- [11] LCR-Meter im Vergleich 2013 (4) - Juni 2013, S.48: www.elektormagazine.de/magazine/elektor-201306/20729
- [12] LCR-Meter 2013 (4) - Juli-August 2013, S. 110: www.elektormagazine.de/magazine/elektor-201307/20893
- [13] LCR-Meter 2013 (5) - November 2013, S.66: www.elektormagazine.de/magazine/elektor-201311/23923

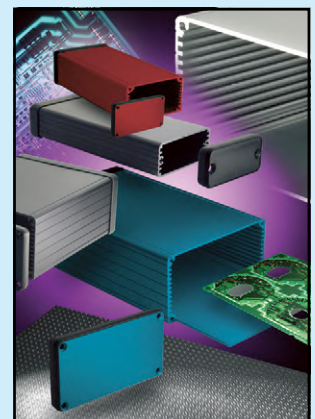
Anzeige



1455 extrudierte Gehäuse

Mehr erfahren: hammfg.com/1455

Mehr als 5000 verschiedene Gehäusedesigns.
hammfg.com/small-case
eusales@hammfg.com • + 44 1256 812812



Vom Ding-Dong-Türgong zur IoT-Türglocke

Verbinden Sie Ihre Türklingel im Home Assistant mit ESPHome!

Ein Beitrag von

Idee, Entwicklung, Text und Illustrationen: **Clemens Valens**
Schaltbilder: **Kurt Diedrich**

Redakteur: **CJ Abate**

Übersetzer: **Rolf Gerstendorf**

Layout: **Giel Dols**

In einem früheren Artikel [1] habe ich die Open-Source-Projekte zur Hausautomation ESPHome und Home Assistant vorgestellt. Dieser Artikel zeigt, wie ich die Türklingel unseres Hauses an das System angeschlossen habe. Hauptziel war die Erkennung von Türklingel-Ereignissen im Home Assistant und die Weiterleitung einer Meldung an Orte, an denen die Türklingel selbst nicht zu hören ist.



Bild 1. So wird aus einem einfachen Ding-Dong ein komplex vernetztes Türklingelsystem.

Der Drucktaster unserer elektromechanischen Ding-Dong-Türglocke wurde vor vielen Jahren durch ein kommerzielles drahtloses System mit mehreren Klingeltönen ersetzt. Es wurde mit zwei abgesetzten Stationen zur erheblichen Vergrößerung der Reichweite der Türklingel geliefert. Leider wird in dem System alles von Batterien versorgt. Wenn deren Kapazität zur Neige geht, wird das Glockenspiel unzuverlässig. Außerdem neigt es dazu, Besucher zu verwirren, da sie nichts hören, wenn sie den Knopf drücken. An unserer Tür klingelt der Briefträger deshalb immer zweimal.

Da ich vor kurzem begonnen hatte, mit der Hausautomation zu experimentieren, war dies eine gute Gelegenheit, die Türklingel zu erneuern und in das System zu integrieren. Um das Verlegen von Kabeln zu vermeiden, entschied ich mich für eine WLAN-Verbin-

dung zwischen der Türklingel und der Steuerung der Hausautomatisierung.

Die alte Klingel und ihre Anschlussdrähte sind einschließlich der Stromversorgung an Ort und Stelle geblieben. Dies erwies sich als recht praktisch, da im Gong genügend Platz war, um eine selbstgebaute drahtlose Türklingel-Schnittstelle einzubauen und diese auch mit Strom zu versorgen.

Da ich die funktionierende drahtlose Türglocke nicht wegwerfen wollte, beschloss ich, deren Drucktaste für meine neue vernetzte Türklingel zu verwenden. Auf diese Weise kam ich zu einem Türglockenspiel der dritten Generation mit WLAN-Verbindung, das einen von mehreren digital erzeugten Klingeltönen abspielt und gleichzeitig einen mechanischen Ding-Dong-Ton erzeugt, während es Benachrichtigungen an die Cloud sendet oder eine andere nützliche Aktion ausführt (siehe

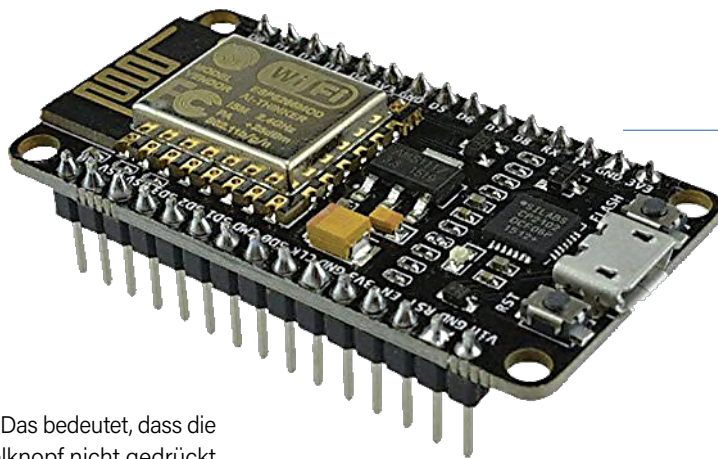
Bild 1). Das akustische Feedback für den Besucher wurde ebenfalls wiederhergestellt.

Aufbau einer Türklingel-Schnittstelle

Für die WLAN-Türklingel-Schnittstelle wählte ich ein ESP8266-basiertes NodeMCU-Board (**Bild 2**), weil es einen analogen Eingang besitzt, den ich verwenden wollte (mehr dazu später). Außerdem benötigt das Board keinen USB-Seriell-Wandler, so dass außer einem Lötkolben keine speziellen Werkzeuge für dieses Projekt erforderlich sind.

Wie bereits erwähnt, wird der Klingelknopf der drahtlosen Türglocke über die alten, vorhandenen Drähte mit dem Türklingel-Interface verbunden. Da sich der Taster außen an der Eingangstür befindet und den Unbilden der Elemente ausgesetzt ist, habe ich die Verbindung mit Vorwiderständen, einem Filterkon-

Bild 2. Das NodeMCU-Modul besitzt an seinem analogen Eingang intern einen Spannungsteiler, der die Eingangsspannung durch 3,2 teilt.



densator und Clamp-Dioden abgesichert. Die Magnetspule der Türklocke benötigt etwa 1 A, wenn sie mit 12 V (AC) betrieben wird. Das ist viel zu viel für den winzigen taktilen Drucktaster des Glockenspiels, denn solche Kontakte vertragen normalerweise nicht mehr als etwa 50 mA, aber mit einem Relais oder Leistungstransistor ist die Verbindung möglich. Ich habe ein kleines 5-V-Relais benutzt, das in meinem Heimlabor herumlag und das vom Mikrocontroller über einen Transistor gesteuert werden kann. Der vollständige Schaltplan ist in **Bild 3** zu sehen.

Überwachen einer Batteriespannung

Die Leitung zum Klingelknopf S1 ist über R4 mit der kalten Seite des Pull-up-Widerstandes

des Tasters verbunden. Das bedeutet, dass die Ader, wenn der Klingelknopf nicht gedrückt wird, die Spannung der Batterie im Inneren des Klingelknopf-Gehäuses führt. Daher habe ich den Draht nicht nur mit einem GPIO-Pin (D7) des Controllers verbunden, sondern auch mit einem Analogeingang (AD0). Auf diese Weise kann der Home Assistant nicht nur Klingeltöne erfassen und weiterleiten, sondern auch den Füllstand der Batterie im Auge behalten. Dies ist praktisch, da die Reichweite der Funkverbindung proportional zur Batteriespannung ist. Ist die Spannung zu niedrig, können die Gegenstelle(n) nicht mehr erreicht werden und das System arbeitet unzuverlässig. Es wäre natürlich auch möglich, auch das Drücken der Klingel statt über den GPIO über den Analogeingang zu erfassen, doch dies würde auch die Firmware verkomplizieren.

Stromversorgung

Die Stromversorgung erfordert einige Aufmerksamkeit, da eine Klingel üblicherweise nur mit einem Trafo versorgt wird (bis 24 VAC), während ich 5 V Gleichspannung für das Relais und das NodeMCU-Board benötigte. Die Magnetspule des Türklocke ist eine ziemlich hohe Belastung für den Trafo, so dass die Ausgangsspannung des Transformators ziemlich in die Knie geht, wenn die Klingel aktiviert wird. Ein großer Speicherkondensator (C1) filtert diese Spannungseinbrüche heraus. Ohne diesen Kondensator würde das NodeMCU-Modul bei jedem Drücken des Klingelknopfes neu gestartet. Der 5-V-Regler ist ein kleiner, 7805-kompatibler Schaltregler (mit einem Eingangs-

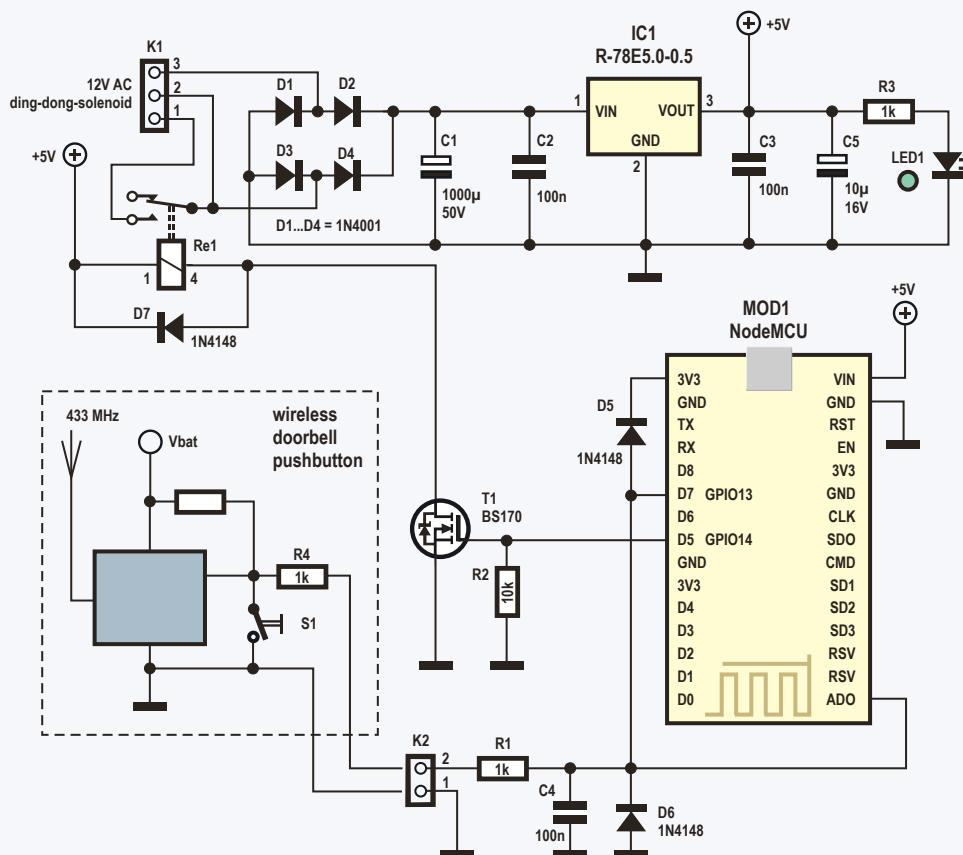


Bild 3. Die Türklingel-Schnittstelle benötigt eine saubere Stromversorgung und die mit der Klingel verbundenen Pins einen gewissen Schutz, da der Taster im Freien dem Wetter ausgesetzt ist.

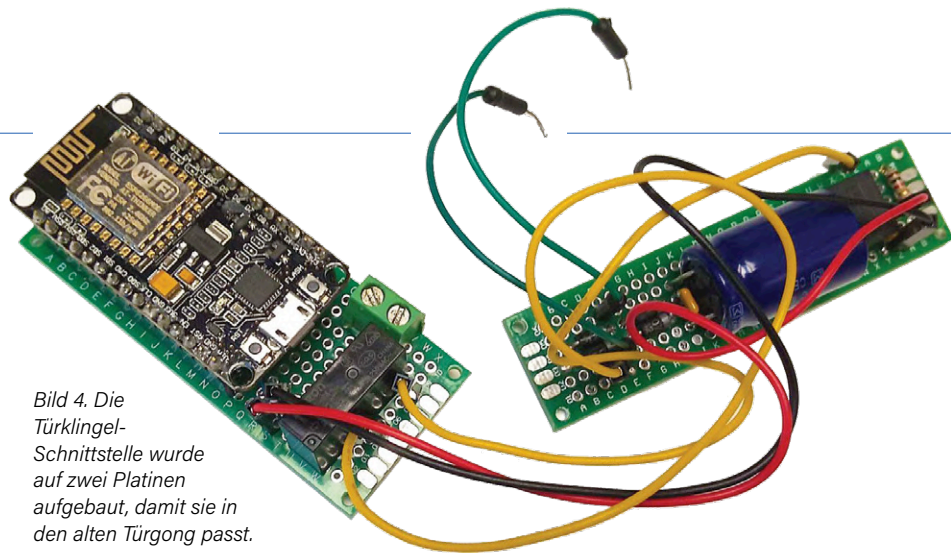


Bild 4. Die Türklingel-Schnittstelle wurde auf zwei Platinen aufgebaut, damit sie in den alten Türgong passt.

ZUR ERINNERUNG

ESPHome ist eine Open-Source-Firmware für ein ESP8266- oder ESP32-basiertes Remote-Gerät wie einen Sensor oder Schalter (und vieles mehr).

<https://esphome.io/>

Home Assistant ist ein Open-Source-Projekt zur Steuerung der Heimautomation, das entfernte Geräte wie Sensoren und Schalter (und vieles mehr) mit Aktoren und Diensten und ähnlichem verbindet, so dass sie durch komplexe Regeln gesteuert werden können.

<https://www.home-assistant.io/>

Siehe [1] für weitere Informationen.

spannungsbereich bis zu 28 V), der dazu beiträgt, Probleme mit der Wärmeableitung zu vermeiden.

Es wurde mit LED1 auch eine praktische Einschaltkontrolle hinzugefügt, über die das NodeMCU-Board nicht verfügt. Ich habe die Türklingel-Schnittstelle auf zwei Prototyping-Platinen verteilt - eine für die Stromversorgung, die andere für den Rest, so dass alles genau in das Gehäuse der alten Klingel passte (**Bild 4**).

ESPHome-Firmware-Konfiguration

Die Software für die Türklingel-Schnittstelle habe ich für ESPHome geschrieben. Der Vorteil liegt auf der Hand: Etwa 30 min „Programmierung“ reichen aus für ein voll funktionsfähiges Gerät mit Over-the-Air -Programmierungsmöglichkeit (OTA), eigener Web-Schnittstelle und perfekter Integration in den Home Assistant. Darüber hinaus können Sie das System nach Belieben mit Ihrem Smartphone (oder Tablet oder Computer) modifizieren, wann immer Sie Lust dazu haben. Die Programmierung erfolgt nicht in C, C++ oder Python, sondern in YAML. YAML ist keine Programmier-, sondern eine Auszeichnungssprache (Markup Language). Ein YAML-Skript oder eine Konfigurationsdatei

wird von ESPHome als Eingabe verwendet und in C/C++-Code umgewandelt, der dann dem Rest der Bibliothek hinzugefügt und mit diesem kompiliert wird. Das Skript gibt an, welche Teile aus ESPHome eingebunden werden sollen, etwa WLAN-Einstellungen, Webinterface oder nicht, wie und welche Arten von Sensoren und Aktoren an die MCU angeschlossen werden und was mit ihnen geschehen soll. Die meisten solcher üblichen Sachen sind vordefiniert, was die Einrichtung ziemlich einfach und schnell macht. YAML mag gewöhnungsbedürftig sein, aber es ist nicht kompliziert, solange man die Einrückungen respektiert.

Das Relais an die Drucktaste binden

Das Relais, das die Magnetspule des Gongs steuert, ist als interner **switch** definiert, der an **GPIO14** angeschlossen ist. Wenn Sie das Relais als **intern** definieren, wird es für den Home Assistant unsichtbar, wodurch eine Überlastung Ihres Systems vermieden wird.

```
switch:
  - platform: gpio
    pin: GPIO14
    id: ding_dong_relay
    internal: true
```

Der **switch** wird mit einer ID versehen, damit er in Automationen verwendet werden kann (siehe unten).

Der Klingelknopf ist ein **binary_sensor** an Pin **GPIO13**. Mit dem Schlüsselwort **inverted** informieren wir ESPHome darüber, dass der Knopf **aktive low** ist.

```
binary_sensor:
  - platform: gpio
    name: "my doorbell
    pushbutton"
    pin:
      number: GPIO13
      inverted: true
    on_press:
      then:
        - switch.turn_on:
            ding_dong_relay
        - delay: 200ms
        - switch.turn_off:
            ding_dong_relay
```

Wenn die Drucktaste gedrückt wird, erzeugt ESPHome ein **on_press**-Ereignis, das an den Home Assistant gesendet wird. Diese Ereignisse können aber auch als Trigger innerhalb der YAML-Datei verwendet werden, um eine oder mehrere Aktionen auszulösen. Dies wird als „automation“ bezeichnet und stellt eine leistungsstarke Funktion von ESPHome dar. Hier verwenden wir das **on_press**-Ereignis, um einen **200ms** langen Impuls am Relaisausgang zu erzeugen. Dazu wird einfach nur die ID des Relais **ding_dong_relay** angegeben. Nun ertönt der Gong, wenn jemand auf die Taste drückt, auch wenn das Netzwerk ausgefallen ist. Kurz gesagt, der ursprüngliche Gong ist wiederhergestellt, wie er vor mehr als 40 Jahren installiert wurde, nur haben wir der einfachen, direkten Druckknopfverbindung mehrere Tausend Transistoren hinzugefügt!

Ein Auge auf den Batterie-Füllstand

Die Batterie des Glockenspiels wird als **sensor** auf **A0** definiert. Sein Typ (**platform**) ist **adc**, da es sich um eine zu digitalisierende Analogspannung handelt. Das NodeMCU-Modul teilt die Eingangsspannung auf **A0** durch 3,2. In ESPHome ermöglichen Filter die Änderung des Ausgangswerts eines Sensors. Ein **multiply** wird als Filter betrachtet. Wenn wir also den Wert des Sensors mit **3.2** multiplizieren, erhalten wir den ursprünglichen Wert zurück.

```
sensor:
  - platform: adc
    name: "doorbell battery
    level"
```

```
pin: A0
filters:
- multiply: 3.2
```

Leuchten spezifizieren

Das NodeMCU-Modul besitzt eine Flash-Taste an GPIO0 und eine LED an GPIO16, die ebenfalls verwendet werden können. Die Flash-Taste könnte man als Test-Taste einsetzen, die LED könnte eine visuelle Anzeige dafür sein, dass jemand an der Tür geklingelt hat und der Assistent dieses Ereignis registriert hat. Daher habe ich GPIO0 als binären Sensor definiert und dem Binärsensor-Abchnitt hinzugefügt.

```
light:
- platform: binary
  name: "LED1"
  output: led1

output:
- platform: gpio
  id: led1
  pin:
    number: GPIO16
    inverted: true
```

PROGRAMMIEREN VON ESPHOME AUF EINEM FABRIKNEUEN ESP-MODUL

Da ein neues (jungfräuliches) ESP8266- oder ESP32-Modul noch nicht ESPHome-kompatibel ist, muss es beim ersten Mal über die serielle Schnittstelle programmiert werden. Je nach Gerät müssen Sie dazu möglicherweise einen USB-zu-seriell-Port-Treiber auf dem Computer installieren, auf dem Sie Ihre ESPHome-Entwicklung durchführen (das heißt, auf dem Computer, auf dem der Home Assistant läuft). Die NodeMCU-Boards mit einem CP2102-USB-Chip von Silabs (Silicon Laboratories), die ich verwendet habe, funktionierten direkt nach dem Auspacken an einem Raspberry Pi. Nachdem Sie das Gerät an den Computer angeschlossen haben, auf dem der Home Assistant läuft, öffnen Sie das ESPHome-Dashboard entweder über die Sidebar (wenn Sie diese Option aktiviert haben) oder indem Sie in der Supervisor-Dashboard-Ansicht auf Open Web UI klicken. Vergewissern Sie sich, dass der serielle Anschluss in der Dropdown-Liste in der oberen rechten Ecke verfügbar ist, die standardmäßig auf OTA (Over-The-Air) eingestellt ist. Sollte dies nicht der Fall sein, starten Sie das ESPHome-Add-on neu (indem Sie zum Supervisor-Dashboard zurückgehen). Das sollte reichen. Der nächste Schritt ist das Klicken auf die rosafarbene Plus-Taste auf dem ESPHome-Dashboard. Dadurch wird ein Assistent geöffnet, der Sie durch den ersten Teil führt. Beachten Sie, dass ich nie ein Zugangspasswort für irgendein Gerät angegeben habe, aber das könnte ziemlich leichtsinnig von mir gewesen sein. Wählen Sie für den Upload den seriellen Port, an den das Gerät angeschlossen ist.

Zur Veranschaulichung dieses Konzepts habe ich ein `light` definiert, das mit einem Ausgang namens `led1` verbunden ist. Dadurch wird im Home Assistant eine „Lichtinstanz“ erzeugt, die andere Möglichkeiten bietet als ein `switch` und außerdem ein anderes Icon besitzt. Die LED an GPIO16

ist als Ausgang mit der ID `led1` definiert, der aktiv low ist (`inverted: true`). Wenn nun der Home Assistant eine Nachricht an das `light` mit dem `name: "LED1"` sendet, um sie ein- oder auszuschalten, wird die Nachricht an den Ausgang mit der entsprechenden `id: led1` weitergeleitet.

Anzeige

almit
Technology for the future

Ready, steady, go!

Schneller in die Zukunft –
Jet-Printing mit LFM-48U SDK
und LFM-48U MJD

Entdecken Sie die Lötpasten, die Ihnen beim Jet-Printing neue Möglichkeiten eröffnen – mit einer überragenden Performance und einer nahezu 100% identischen Punktgröße. Welche Lötpaste die genau richtige für Ihre Anforderungen ist, sagt Ihnen gerne Ihr Almit-Fachberater.





PASSENDE PRODUKTE

- > **NodeMCU Mikrocontroller-Board mit ESP8266**
www.elektor.de/nodemcu-microcontroller-board-with-esp8266-and-lua
- > **Buch: IoT Home Hacks with ESP8266**
www.elektor.de/iot-home-hacks-with-esp8266

Die vollständige YAML-Konfigurationsdatei für ESPHome kann unter [2] heruntergeladen werden. Beachten Sie, dass Sie die SSID und das Passwort an Ihr WLAN-Netzwerk anpassen, möglicherweise möchten Sie auch den Namen des Geräts ändern.

ESPHome-Add-on für den Home Assistant

Da ich das ESPHome-Add-On im Home Assistant installiert habe, habe ich das YAML-Skript im Add-on geschrieben. Auf diese Weise ist die YAML-Datei des Geräts bereits im System vorhanden, was spätere OTA-Updates erleichtert. Beim ersten Mal jedoch, zumindest bei einem fabrikneuen ESP8266, müssen Sie die serielle Schnittstelle zur Programmierung der MCU verwenden, da die OTA-Programmierung noch nicht aktiviert wurde (siehe **Kasten**).

Automatisierung im Home Assistant

Die LED an GPIO16 wurde nur in der YAML-Datei von ESPHome als einsetzbar erklärt. Eine Automatisierungsregel dafür wurde nicht erstellt, so dass sie ohne Aufforderung des Home Assistant gar nichts tut. Die Idee war, sie aufleuchten zu lassen, wenn jemand klingelt. Dies kann mit einer einfachen Automatisierung im Home Assistant erreicht werden: auf einen Klingelknopfdruck zu warten und entsprechend die LED einzuschalten.

Im Home Assistant gibt es mehrere Möglichkeiten, den Automationseditor zu öffnen, zum

Beispiel links im Menü auf **Configuration** und danach rechts auf **Automations** zu klicken. Eine neue Automatisierung wird durch einen Klick auf die runde, orangefarbene „+“-Schaltfläche in der unteren rechten Ecke eingeleitet. Sie können nun eine Phrase eingeben, die der Home Assistant versucht, in die gewünschte Automation umzuwandeln. Es ist jedoch nicht so einfach, eine passende Phrase zu finden, und es ist Zeitverschwendung, eine vom Home Assistant erstellte fehlerhafte Automatisierungsregel zu korrigieren, so dass wir besser auf **Skip** klicken. Geben Sie einen Namen für die Automatisierung ein und optional eine Beschreibung. Verlassen Sie den Modus durch **Single (default)**.

Trigger und Aktionen

Es gibt mehrere Möglichkeiten, ein Trigger-Ereignis anzugeben, das die Automatisierung startet. Für mich ist es in diesem Fall der am meisten intuitive Weg, als **Action type** ein **Device** zu wählen, das in der **Device**-Liste mit dem Namen aufgeführt ist, den Sie in der YAML-Datei angegeben haben. Genau so suchen Sie in der **Trigger**-Liste nach dem Namen, den Sie der Flash-Taste gegeben haben.

Es gibt keine **conditions**, also überspringen Sie diesen Teil. **Actions** sind ähnlich wie **Trigger**, und wie bei Triggern gibt es mehr als einen Weg, um das gewünschte Ergebnis zu erhalten. Eine Möglichkeit besteht darin, **Device** als **Action type** zu wählen und Ihr Gerät aus der **Device**-Liste auszuwählen. Aus der **Action**-Liste geben Sie **Turn on LED1** an (oder den

Namen, den Sie der LED gegeben haben). Sichern Sie Ihre Arbeit, indem Sie auf das runde orangefarbene Disketten(!)symbol in der unteren rechten Ecke klicken.

Überprüfen Sie, ob die Automatisierung funktioniert, indem Sie an der Tür klingeln. Wenn Sie dies tun und die Automatisierung funktioniert, werden Sie feststellen, dass es keine Möglichkeit gibt, die LED wieder auszuschalten.

Ein light in der Benutzeroberfläche

Um dem abzuweichen, werden wir der Übersichtsseite des Home Assistant eine so genannte **card** hinzufügen, die (durch Klicken auf die **card**) nicht nur die LED ausschaltet, sondern auch eine Statusansicht der LED ermöglicht, so dass Sie nicht den ganzen Weg bis zur Türklingel gehen müssen, um zu sehen, ob die LED eingeschaltet ist oder nicht.

Klicken Sie im Menü auf der linken Seite auf **Overview**, dann auf die drei Punkte in der oberen rechten Ecke (**Open Lovelace UI menu**) und wählen Sie **Configure UI**. Klicken Sie nun auf die runde orangefarbene „+“-Schaltfläche in der unteren rechten Ecke (im Home Assistant wird viel herumgeklickt). Suchen Sie die **light**-Karte und klicken Sie darauf. Suchen Sie die **Entity**-Liste nach der LED und wählen Sie sie aus. Geben Sie auch einen einprägsamen Namen ein, mit dem Sie die LED in der Übersicht leichter identifizieren können, wenn es in Ihrem weiterentwickelten Hausautomationssystem einmal eine Menge solcher **lights** gibt.

Speichern Sie die Karte und verlassen Sie den UI-Konfigurationsmodus, indem Sie auf das Kreuz in der linken oberen Ecke klicken (**Close**). Durch Klicken auf die Karte sollten Sie nun den Status der LED umschalten können.

Zum Abschluss

Wenn Sie die in diesem Artikel vorgestellten Schritte befolgt haben (und alle meine Fehler korrigiert haben, die ich in den neuesten Versionen der sich ständig weiterentwickelnden Projekte ESPHome und Home Assistant natürlich eliminiere), haben Sie jetzt eine Türklingel mit WLAN-Verbindung, die Sie im Home Assistant automatisieren können. Sie können viel mehr (nützliche) Dinge tun, als eine LED leuchten zu lassen, aber das überlasse ich Ihnen. Dieser Artikel zeigte für einen Einstieg in die Materie die Grundprinzipien, und jetzt liegt es an Ihnen, die Dinge zu verbessern und sie an Ihre Bedürfnisse und Wünsche anzupassen. ◀

200089-03

Sie haben Fragen oder Kommentare?

Haben Sie Fragen oder Anmerkungen zu dem Artikel? Schicken Sie eine E-Mail an den Autor unter clemens.valens@elektor.com oder kontaktieren Sie Elektor unter editor@elektor.com.

WEBLINKS

- [1] **Hausautomation leicht gemacht:**
www.elektormagazine.de/magazine/elektor-154/58936
- [2] **How-To: Integrate Your Doorbell in Home Assistant Using ESPHome:**
www.elektormagazine.com/labs/esphome-doorbell

Batteriemanagement

Was man beim Einsatz von (Lithium-)Akkus beachten muss

Ein Beitrag von

Text und Bilder: **Dr. Thomas Scherer**

Redaktion: **Jens Nickel**

Layout: **Giel Dols**

Mit modernen Akkus auf der Basis des Alkalimetalls Lithium wurde das Speichern von elektrischer Energie einfacher und komplizierter zugleich. Man bekommt mehr Energie pro Rauminhalt unter und hat kaum Selbstentladung. Aber beim Laden mehrzelliger Akkus braucht es „Management“!



Was ist bei Lithium-Akkus so besonders, dass sie anders als die anderen gängigen Akku-Technologien zwingend ein elektronisches Akkumanagement benötigen? Um diese Frage und den Umgang damit geht es im nachfolgenden Text.

Nicht-Lithium-Technik

Blei

Wie war das weiland mit Blei-Akkus doch simpel: Man drückte so viel Strom wie zulässig oder verfügbar in den Akku und hörte damit auf, wenn die Ladeschlussspannung (ab jetzt: LSS) erreicht wurde. Und Akkus aus mehr (in Serie geschalteten) Zellen? Auch kein Problem. Einfach die LSS mit der Zellenzahl multiplizieren und die Sache war geritzt. Das gilt noch heute.

Warum klappt das? Einfach deswegen, weil Blei-Säure-Akkus beim Überladen nicht gleich kaputt gehen (sondern höchstens etwas ausgasen). Wenn eine der Zellen eines Akkus daher etwas weniger

Kapazität als der Rest aufweisen sollte, kann man die restlichen Zellen eben trotzdem auf die passende LSS aufladen. Diese Robustheit sorgt dafür, dass fast alle Autos auch heute noch eine 12-V-Bord-Batterie in eigentlich anachronistischer Blei-Technologie besitzen, obwohl andere Akkutypen ordentlich Gewicht sparen würden. Ein mit Blei-Akkus bestückter Laptop würde eindeutig zum Schlepptop. Trotzdem: Diese Akkus sind so robust, dass man eine Standard-44-Ah-Autobatterie für einen Kleinwagen locker jahrelang mit einer 55-A-Lichtmaschine mit Primitivregler quälen kann, was schon sehr eindrucksvoll ist. Bekanntlich beträgt die typische Zellenspannung 2 V. Die LSS liegt bei etwa 2,35...2,4 V/Zelle, was sich für die sechszellige Autobatterie auf 14,1...14,4 V summiert. Eine Blei-Akkuzelle sollte nicht unter 1,8 V entladen werden. Ergo nimmt eine Autobatterie Schaden, wenn sie unter 10,8 V gerät, so wie das durch die Selbstentladung von etwa 5%/Monat z.B. bei Motorrädern passieren kann, wenn man sie im Winter nicht zwischendurch einmal auflädt.



Bild 1. Typische NiMH-Zelle im Format AA mit geringer Selbstentladung.

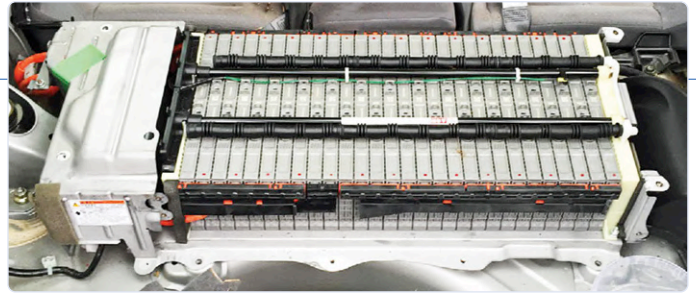


Bild 2. Akku des ehemaligen Prius II des Autors. Zu sehen sind 28 Pakete zu je 6 Zellen = 201,6 V Nennspannung.

Nickel-Cadmium

Sogenannte NiCd-Akkus in Rundzellen wurden vor allem als Ersatz für Zink-Kohle-Batterien in Geräten der Unterhaltungselektronik populär. Mittlerweile ist dieser Akkutyp aufgrund der Umweltschädlichkeit des enthaltenen Cadmiums in der EU schon über 10 Jahre verboten. NiCd-Akkus sind mittlerweile aus dem Alltag verschwunden, weshalb sich viele Worte hier nicht lohnen.

Kurz zu den Eigenschaften dieser Akku-Technologie: NiCd-Akkus vertragen recht hohe Ströme und viele Ladezyklen, weshalb sie früher gerne für Modell-Autos etc. eingesetzt wurden. Auch hier gibt es eine Nominalspannung, nämlich 1,2 V pro Zelle. Geladen wird ähnlich wie beim Blei-Akku bis zur LSS von 1,5...1,57 V/Zelle (je nach Strom). Tiefentladung unter 0,85 V/Zelle schadet dem Akku, auch wenn dieser noch wesentlich robuster als ein NiMH- oder Lithium-Akku ist. Die Selbstentladung reicht bis zu heftigen 20%/Monat. Eine Serienschaltung beim Laden ist kein Problem – ergo ist kaum Management erforderlich.

Nickel-Metallhydrid

Kleine NiMH-Rundzellen wie z.B. der AA-Typ von **Bild 1** haben als ungiftige Alternative das Erbe der NiCd-Akkus angetreten. Aber es gibt sie nicht nur in klein: Jahrelang waren sie z.B. in akkubetriebenen Werkzeugen aufgrund des günstigen Preises und des einfachen Handlings Standard. Es gibt sie sogar nicht so selten noch als größeres Modell: Der „Prototyp“ aller Hybrid-Autos, der Toyota Prius, war bei den Modellen II und III mit einem fast einen Zentner schweren NiMH-Akku mit rund 200 V und etwa 6,5 Ah bestückt. Dieser erreicht in solchen Autos enorme Lebensdauern und Zyklenzahlen, die auf ein Management des Lade- und Entladeverhaltens und der Ausnutzung lediglich einer Teilkapazität beruhen. **Bild 2** zeigt die geöffnete Batterie meines ehemaligen Prius II. Bei Plug-in-Hybriden und neueren Hybrid-Modellen ist Toyota und auch alle anderen Hersteller aufgrund der Gewichtsersparnis dann doch auf Lithium umgestiegen, nachdem die mit dieser Technologie verbundenen Probleme anscheinend gut genug gelöst werden konnten.

Die Selbstentladung von NiMH-Akkus ist (normalerweise) ähnlich hoch wie bei NiCd-Typen. Allerdings gibt es Varianten wie der Akku in Bild 1, die auf Kosten der Leistungsdichte eine sehr viel geringere Selbstentladung von etwa 1,5%/Monat aufweisen. Für seltener genutzte Gadgets sind diese Akkus immer vorzuziehen, da tiefentladene NiMH-Akkus unbrauchbar werden.

Lithium

Akkus auf Lithium-Basis verdrängen zunehmend alle anderen Akkutypen in den meisten Anwendungsgebieten, da sie bekanntlich das bessere Leistungs- und Volumengewicht und eine extrem niedrige Selbstentladung von etwa 5%/Jahr aufweisen. Doch kein Licht ohne

Schatten: Die sogenannten LiIon-Akkus (Lithium-Ionen) sind sehr empfindlich gegen Überladung und Tiefentladung. Sie verlieren dabei nicht nur Kapazität, manche Typen können durchaus platzen oder gar in Flammen aufgehen. Nur um es erwähnt zu haben: Bei geringen Temperaturen sinkt die verfügbare Kapazität. Auch wenn Letzteres reversibel ist, so ist das doch ein Problem für damit betriebene Elektroautos. Hohe Temperaturen bekommen LiIon-Akkus ebenfalls nicht. Aufgrund der Empfindlichkeit gegen Überladung ist bei einer Serienschaltung für höhere Spannungen Vorsicht geboten. Es braucht hier auf jeden Fall zusätzliche Elektronik in Form eines sogenannten Balancers, der beim Laden verhindert, dass einzelne Zellen mit zufällig etwas geringerer Kapazität schon überladen werden, wo die anderen Zellen noch nicht ganz voll sind. Und es gibt noch mehr Faktoren zu berücksichtigen. Auf jeden Fall dürfte klar sein, dass LiIon-Akkus immer zwingend ein Batteriemanagementsystem (das Akronym BMS passt auch im Englischen) mit zum Teil komplexen Eigenschaften benötigen.

Was die Sache noch komplizierter macht ist, dass es nicht nur einen Typ von Lithium-Akkus gibt. Je nach verwendeter Technologie und interner Chemie variieren die möglichen Zyklenzahlen, die TZS (typische Zellenspannung), die LSS, die ELSS (Entlade-Schluss-Spannung), der maximale Entladestrom und weitere Kenngrößen. Die BMS müssen auf diese Parameter und natürlich die im konkreten Fall zu beherrschenden Ströme und Temperaturbedingungen abgestimmt sein.

Wenn man von Elektroautos absieht, für die besondere Bedingungen gelten und bei denen auch enormer Aufwand zum Schutz der teuren Akkus betrieben wird, ist die Sache bei kleineren Batteriesystemen etwa für tragbare Geräte wie Laptops oder Bluetooth-Lautsprecher, Werkzeuge wie Akkuschrauber oder Haushaltsgeräte wie Akkusauger doch zum Glück etwas einfacher, da Kühlung und Heizung entfallen. Am einfachsten ist die Sache übrigens bei Smartphones, denn die enthalten keine in Serie geschalteten Akkuzellen, weshalb kein Balancer nötig ist. Die integrierte Ladeelektronik muss lediglich die LSS und die ELSS sowie den maximalen Ladestrom berücksichtigen.

Bei den kleineren LiIon-Akkus dominieren zwei Kategorien: Auf der einen Seite Typen, deren Elektroden mit den Werkstoffen Cobalt, Cobalt-Dioxid, Mangan und/oder Nickel und Aluminium gebaut sind. Der Löwenanteil entfällt hier auf die bewährte LiCoO₂-Technik. Glücklicherweise beträgt bei all diesen Typen die TZS etwa 3,6...3,7 V, die LSS 4,2 V und die ELSS 2,5 V (Ausnahme: bei LiNi_xCo_yAl_zO₂ ist die ELSS 3,0 V). Auch wenn sich die erlaubten Ströme und die Lebensdauer unterscheiden, lassen sich all diese Typen im Prinzip mit den gleichen Verfahren = den gleichen ICs managen.

Abweichend hiervon ist der Akku aus Lithium-Eisenphosphat (LiFePO₄), bei dem gilt: TZS = 3,2 V, LSS = 3,65 V und ELSS = 2,5 V. Dieser Akkutyp bietet hohe Zyklenzahlen und eine etwas gerin-

gere Leistungsdichte. Man braucht hierfür also zumindest für diese Spannungen gedachte Balancer. Die Halbleiter-Industrie bietet selbstverständlich Lösungen für beide Kategorien und für verschiedene in Serie geschaltete Zellenzahlen und Ströme an – manche ICs sind sogar programmierbar, für bestimmte Parameter umschaltbar oder haben ein Interface für Mikrocontroller.

LiIon-Akkus gibt es in verschiedenen Bauformen: Die LiPo-Typen (Lithium-Polymer) sind aufgrund der möglichen Formen und ihrer Flachheit vor allem in Mobilgeräte und Laptops eingebaut. Der Begriff Polymer bezieht sich vor allem darauf, dass die LiPos statt in ein festes Gehäuse einfach in eine Kunststofffolie eingeschweißt sind – Knicke sind da unerwünscht. Dann existieren diverse Rundzellen, darunter besonders verbreitet der Typ 18650 ($\varnothing = 18$ mm, Länge = 65 mm), der erstaunlicherweise zu Tausenden als Paket in Teslas Elektroautos verbaut wird – die Lücken zwischen den runden Zellen werden zum Temperaturmanagement genutzt. **Bild 3** zeigt eine 18650-Zelle aus chinesischer Fertigung mit Fantasieangaben: Wie man sieht, ist nicht nur Papier, sondern auch Kunststoff und HTML sehr geduldig, denn 7.800 mAh passen noch lange nicht in das 18650-Format. Die besten aktuellen Zellen kommen real etwa auf die Hälfte. Schließlich gibt es noch die sogenannten prismatischen Zellen, deren Bezeichnung lediglich darauf hinweist, dass ihr Gehäuse nicht rund, sondern quaderförmig ist. Solche Zellen sind z.B. im Mini-Auto „Hotzenblitz“ aus deutscher Produktion verbaut. In **Bild 4** sieht man schön, dass jede Zelle mit extra Elektronik versehen ist, welche ihren Zustand überwacht.

Balancer

Der Begriff „Batteriemanagment“ ist ja eigentlich recht schwammig. Er umfasst alles, was sich irgendwie mit der Überwachung und Steuerung des Ladens, des Entladens und des Betriebszustands von Akkus befasst. Während sich dies bei Blei-, NiCd-, und NiMH-Akkus lediglich auf die Einhaltung von LSS, ELSS und gelegentlich der Begrenzung von Strömen bezieht, ist die Sache aufgrund der Probleme mit der Serienschaltung von Zellen bei Lithium-Akkus komplexer. Hier braucht das BMS bei mehr als einer Zelle zwingend einen Balancer!

Der zentrale Grund hierfür ist, dass Lithium-Akkus Überladung gar nicht mögen. Und weil die Kapazität der Zellen auch bei Massenfertigung in einer Charge leicht variiert und Unterschiede auch durch Alterung auftreten können, gibt es bei einer Serienschaltung immer eine Zelle, die beim Laden zuerst voll ist. An dieser Stelle ist aber die LSS für den Gesamt-Akku noch nicht erreicht, und wenn dann weiter geladen würde, würde die schwache Zelle zumindest geschädigt – sie wäre dann beim nächsten Laden noch früher „fertig“ und ein circulus vitiosus wäre im Gang, der nach ein Paar Zyklen zumindest zum Ausfall der schwachen Zelle und damit des Akkus – wenn nicht zu Schlimmerem – führt. Ein Balancer verhindert das.

In der einfachsten Version bei einem Zwei-Zellen-Akku wie in **Bild 5** überwacht der Balancer die Spannungen der beiden Zellen. Erreicht eine der beiden Zellen die LSS, dann wird eine angemessenen kleine Last parallel zur Zelle gelegt, damit der weitere Ladestrom quasi an dieser Zelle vorbei fließt und so immer noch die andere, bessere Zelle vollgeladen werden kann. Bei mehr Zellen skaliert man den Balancer entsprechend. **Bild 6** zeigt eine kleine BMS-Platine für vier (kleine) Zellen, wie sie in eBay für unterschiedliche Zellenzahlen massenhaft für Preise zwischen 1 € und 10 € angeboten werden.

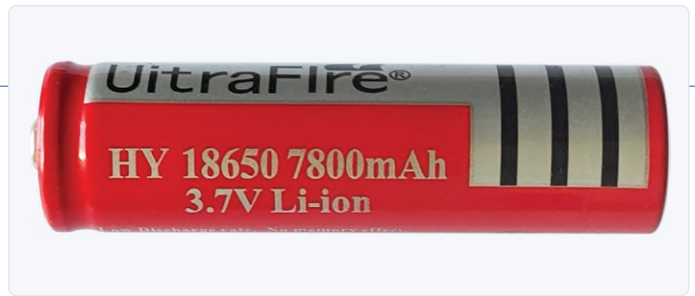


Bild 3. Typische Lithium-Zelle im Format 18650 mit extrem übertriebener Kapazitätsangabe. Diese Zelle hat eine Elektronik integriert, die Tiefentladung und Überladung verhindert.

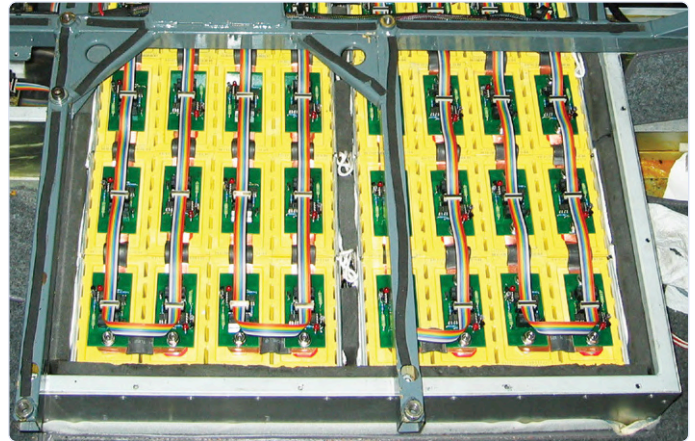


Bild 4. Teil einer LiFePo-Batterie des deutschen Kleinstwagens „Hotzenblitz“ mit Batteriemangement-Elektronik (Bild: Joes-Wiki, CCASA 3.0 DE [2]).

In **Bild 7** kann man sehen, welchen Aufwand ich für den Balancer des 48-V-LiFePo-Akkus meines Segway-Clones getrieben habe: Jede Zelle wird von einem eigenen Mikrocontroller überwacht. Man könnte nun einwenden: „Eine kleine Last?“ und hätte damit nur prinzipiell, aber nicht praktisch recht. Rein formal sollte die im Notfall parallel geschaltete Last zwar den vollen Ladestrom an der zu schützenden Zelle vorbeischießen können, aber der ist praktisch

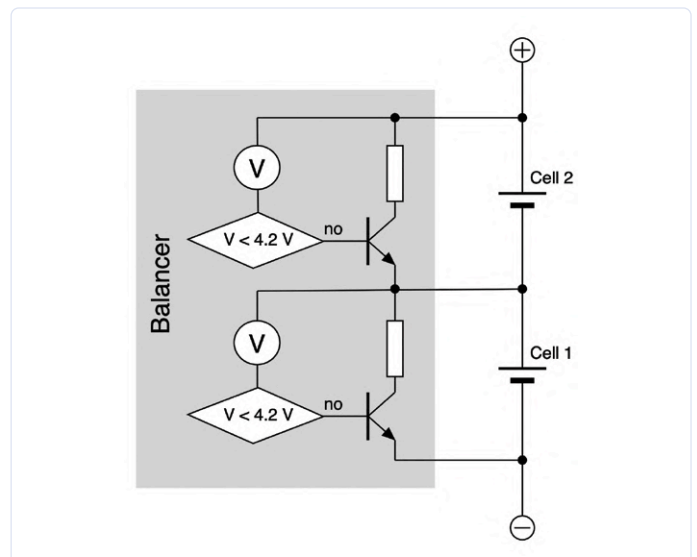


Bild 5. Prinzipschaltung eines einfachen Balancers mit zwei Zellen.

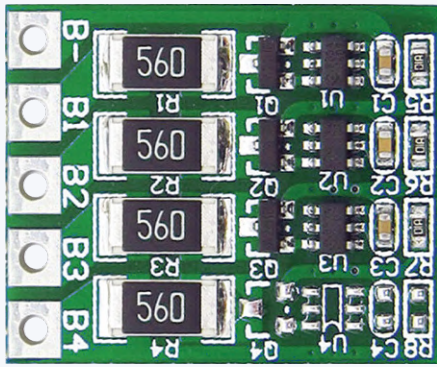


Bild 6. Solche Balancer-Platinen findet man bei eBay in jeder Variante und Ausführung zu geringem Preis.



Bild 7. LiFePo-Akku mit Balancer aus dem Segway-Clone des Autors.

deutlich kleiner als der maximal vorkommende Ladestrom. Hintergrund ist, dass LiIon-Akkus zwar am Anfang bis zu einem Ladezustand von 50...80% mit dem nominalen Strom geladen werden, ab dann aber nur noch mit einem kleinen, in der Regel stetig sinkenden Strom. Bei LiCoO₂-Zellen ist der Strom jenseits von 3,95 V beim Laden ziemlich niedrig. Da die Unterschiede gesunder Zellen nicht sooo groß sind, reicht es, wenn man diesen deutlich geringeren Strom abführt. In der Praxis reichen dafür Zusatzlasten für etwa 1...5% des Maximalstroms fast immer aus. Reicht es mal nicht, ist eine Zelle „echt schlecht“ und ein höherer Ballast würde auf die Dauer auch nichts bringen.

Je nach den vorkommenden Strömen wird durch einen Balancer Energie vernichtet, die vielleicht besser doch genutzt statt verheizt werden kann. Für solche Zwecke gibt es Spezial-Balancer, welche die überschüssige Energie an der zu schützenden Zelle nicht einfach in Wärme umwandeln, sondern via Step-up-Schaltregler (pro Zelle!) direkt wieder dem Gesamtsystem zuführen. Eine solcher Aufwand

wird aber verständlicherweise selten getrieben.

Noch ein anderes Kriterium ist wichtig: Balancer basieren auf elektronischen Bauelementen und integrierte Schaltungen lassen sich nicht ganz so einfach und vor allen Dingen nicht preiswert für hohe Spannungen bauen. Von daher stößt die mögliche Skalierung an praktische Grenzen. Moderne BMS-ICs gibt es typischerweise für 2...12 Zellen. Bei 12 Zellen hat man schon eine Betriebsspannung von gut 50 V. Will man mehr Zellen in einem Akku, skaliert man einfach die BMS-Elektronik. Bei den 800-V-Systemen moderner Elektro-Autos kommt man dann schon auf gut 20 Module mit 12s-BMS-ICs in Serie (das „s“ bei 12s steht für „Zelle“). Die einzelnen Subsysteme sind dann per (potentialgetrenntem) Bus miteinander und meistens mit einer zentralen Steuerung verbunden.

ICs

Wie schon erwähnt, bietet die Industrie eine Vielzahl spezieller ICs für alle denkbaren Anwendungsfälle – zu viele, um zu versuchen, hier eine brauchbare Übersicht zu geben. Stattdessen werden ein konkretes IC und seine Schaltung exemplarisch beschrieben. Doch vorab gibt es noch etwas nachzutragen.

Neben dem Balancing hat ein BMS bei den meisten Anwendungen mit Lithium-Akkus auch noch die Aufgabe, den Akku vor Unterspannung und Überstrom zu schützen. Manchmal ist auch die Ladeschaltung mit in das System einbezogen. Bei diesen komplexen Aufgaben macht der Einsatz eines Mikrocontrollers durchaus Sinn. Moderne BMS haben deshalb manchmal einen kleinen SoC integriert oder ein entsprechendes Interface. Man findet BMS-ICs deshalb nicht nur bei den üblichen Verdächtigen für analoge ICs oder Schaltregler, sondern auch bei Firmen wie etwa Microchip, Maxim (siehe **Bild 8**) oder Renesas. Und von letzterem vor allem für seine Mikrocontroller bekannten Hersteller stammt das Beispiel-IC.

Bild 9 zeigt das interne Blockschaltbild des BMS-ICs ISL94212. Auf der linken Seite sieht man neben den Anschlüssen für Masse (unten) und positiver Betriebs- bzw. Akkuspannung (oben) die vielen Anschlüsse für insgesamt zwölf Lithium-Zellen. Die Spannungen gelangen vom „Input Buffer / Level Shifter“ über einen Umschalter (VC MUX) und einen weiteren Selektor (MUX) auf einen A/D-Konverter. Daneben gibt es Referenzspannungen, Einheiten zur Messung von Temperatur usw., Überwachung der ELSS und natürlich das Wichtigste: der voll digitale Block „Control Logic and Communications“).

Bild 10 zeigt die prinzipielle Anwendungsschaltung. Hier sieht man die IC-Beschaltung genauer. Links sind die Zellen angeschlossen

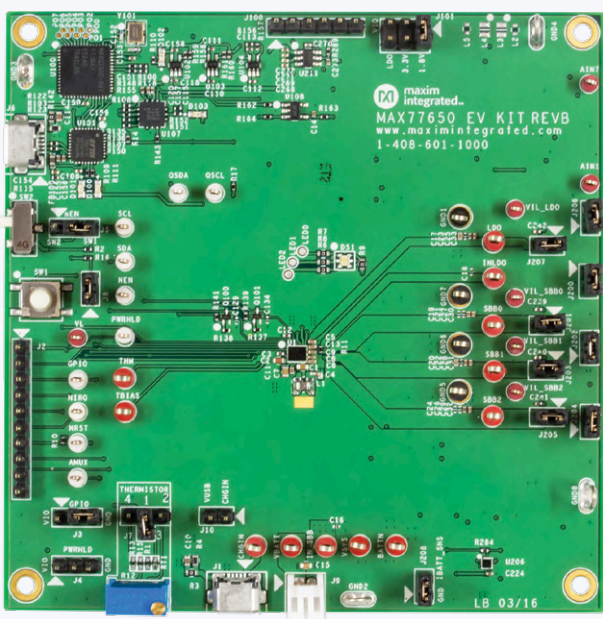


Bild 8. Evaluation Kit für die BMS-ICs MAX77650/MAX77651 (Bild: Maxim Integrated).

und rechts vom linken IC ISL94212 steuert ein Mikrocontroller das ganze BMS via digitaler Schnittstelle. Will man mehr als 12 Zellen überwachen, werden weitere Module mit je einem ISL94212 erforderlich. Die einzelnen Module werden seriell gekoppelt: Daher gibt es die beiden Pins DHi und DLo zweimal pro IC (mit dem Index 1 und 2). Aufgrund der hohen Spannungen solcher Modulketten werden die seriellen Schnittstellen nicht direkt, sondern über Kondensatoren mit kleinen Kapazitäten gekoppelt. Das IC bietet natürlich nicht nur eine Überwachung vieler Zellen, sondern auch das notwendige Balancing. **Bild 11** enthält die prinzipielle Anschaltung von je einem Lastwiderstand (ganz links) über einen kleinen Power-MOSFET, der über den zugehörigen IC-Ausgang CBX gesteuert wird. Auf diese Weise lässt sich das Balancing auch für recht hohe Ströme realisieren. Die Details der vielen IC-Funktionen entnehme man dem Datenblatt [1].

Praxis

Wenn Sie es mit einem Lithium-Akku zu tun haben, dann haben Sie im Prinzip die Wahl zwischen drei Strategien: Für einfache Anwendungen kaufen Sie schlicht so eine kleine, preiswerte und fertig bestückte Platine aus chinesischer Produktion wie die von Bild 5. Bei einem größeren und/oder teureren Akku können Sie auch von Markenherstellern aus einem unüberschaubar großen Fundus an fertigen BMS-Lösungen auswählen. Last not least bietet es sich an, für das in Frage kommende Akkupaket eine Lösung maßzuschneidern. Auswahl an ICs gibt es genug. In einigen Fällen kann und muss man dafür dann aber auch eigenen Code für die Überwachung und Steuerung des Akkus schreiben, wenn man es wirklich genau machen will. In all diesen Fällen ist Google ein guter Freund. ◀

180350-01

Sie haben Fragen oder Kommentare?

Gerne können Sie sich an die Redaktion wenden unter der E-Mail-Adresse: redaktion@elektor.de.



PASSENDE PRODUKTE

- **Akku-Checker**
www.elektor.de/battery-checker-071131-71
- **OWON XDM2041 Multimeter (55000 Counts)**
www.elektor.de/owon-xdm2041-multimeter-55000-counts
- **RS232-Datenlogger (140126-41)**
www.elektor.de/rs232-data-logger-and-spy-tool-140126-41

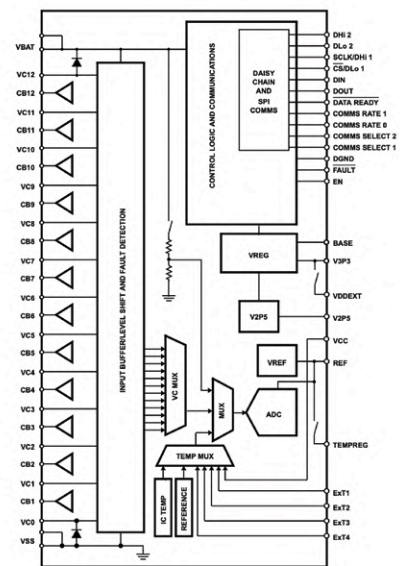


Bild 9. Blockschaltung des Innenlebens des BMS-ICs ISL94212 (Bild: Renesas).

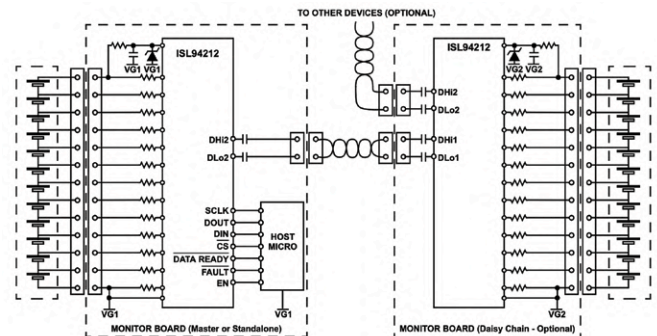


Bild 10. Prinzipielle Konfiguration, wenn man BMS mit dem ISL94212 auf mehr als 12 Zellen hochskaliert (Bild: Renesas).

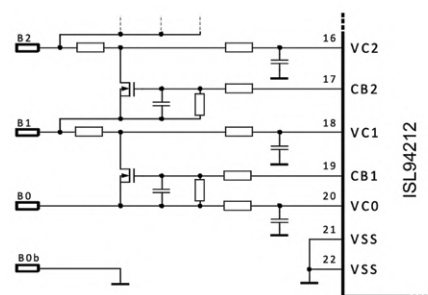
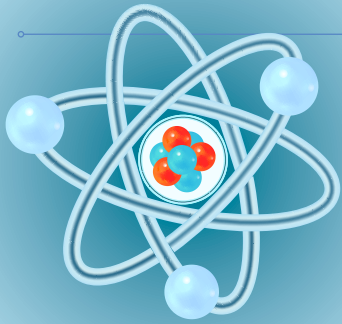


Bild 11. So werden beim ISL94212 zwecks Balancing externe MOSFETS angesteuert, die zu jeder Zelle einen Lastwiderstand parallelschalten können.

WEBLINKS

- [1] **Datenblatt Renesas ISL94212:** www.renesas.com/en-us/www/doc/datasheet/isl94212.pdf
- [2] **CCSA 3.0 DE:** <https://creativecommons.org/licenses/by-sa/3.0/de/deed.en>



Aller Anfang...

muss nicht schwer sein!

In der letzten Elektor-Ausgabe haben wir uns die H-Schaltung angesehen und dabei bemerkt, dass es besser ist, den Leitungswiderstand in der realen Welt um uns herum nicht (oder nicht immer) zu vernachlässigen. Jetzt ist es jedoch an der Zeit, dass wir uns mit Wechselstrom befassen.

In den früheren Folgen dieser Artikelserie haben sich alle Beispiele und Berechnungen auf **Gleichspannung** bezogen. Interessant wird es aber erst bei **Wechselspannung** - ein Audioverstärker beispielsweise verstärkt nur Wechselspannungen.

Bekanntlich liefern Batterien eine Spannung, deren Polarität sich nicht ändert - das ist eine Gleichspannung. Die **Höhe** dieser Spannung ändert sich (nimmt ab), wenn die Batterie leer wird, aber die **Polarität** ändert sich nicht. In **Bild 1** ist das Lehrbuchbeispiel einer Gleichspannung skizziert.

Bei einer Wechselspannung hingegen wechselt die Polarität (mehr oder weniger) periodisch. Normalerweise ändert sich auch die Höhe der Spannung (siehe unten), aber bei der Wechselspannung in **Bild 2** (eine Rechteckspannung, und ja, das ist wirklich eine Wechselspannung!) ändert sich nicht die Höhe, sondern nur die Polarität. Es wird deutlich, dass der Mittelwert der Wechselspannung (berechnet über einen längeren

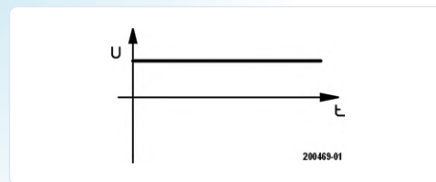


Bild 1. Gleichspannung.

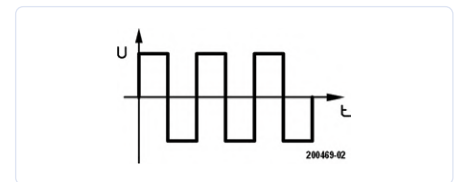


Bild 2. Wechselspannung (hier in Form eines Rechtecks).

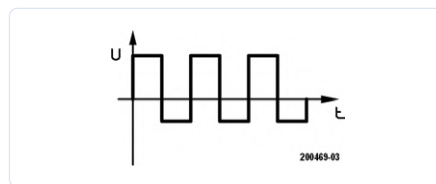


Bild 3. Eine Wechselspannung, die einer Gleichspannung überlagert ist.

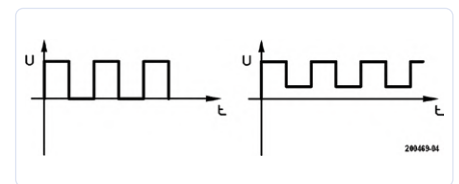


Bild 4. Pulsierende Gleichspannung.

Zeitraum) in Bild 2 gleich Null ist. Es handelt sich dann um eine reine Wechselspannung. Ebenso ist es denkbar, dass der Mittelwert einer Wechselspannung nicht gleich Null ist, obwohl die Polarität ständig wechselt. In diesem Fall haben wir es mit einer Wechselspannung zu tun, die von einer Gleichspannung überlagert ist, oder anders ausgedrückt, die betreffende Spannung ist die Summe aus einer reinen Wechselspannung und einer Gleichspannung. Dies ist in **Bild 3** zu sehen.

Bild 4 zeigt eine ähnliche Situation, jedoch mit einem wichtigen Unterschied. Wir sehen eine Spannung, deren Höhe sich periodisch ändert, deren Polarität aber immer gleich bleibt. In einem solchen Fall spricht man von einer pulsierenden Gleichspannung.

Periodendauer und Frequenz

Die Zeitspanne einer **vollständigen** Periode einer Wechselspannung wird als Periodendauer T dieser Wechselspannung bezeich-

net und wird in Sekunden angegeben. Der Kehrwert dieser Periodendauer ist die Frequenz f , ausgedrückt in Hertz (Hz):

$$f = \frac{1}{T}$$

In Europa beträgt die Netzfrequenz 50 Hz (auf der anderen Seite des großen Teichs 60 Hz). Die Periodendauer der Netzfrequenz von 50 Hz ist:

$$T = \frac{1}{f} = \frac{1}{50\text{Hz}} = 0,02\text{s} = 20\text{ms}$$

Sinusförmige Wechselspannung

Die Wechselspannung, mit der wir es am meisten zu tun haben, ist die in **Bild 5** skizzierte sinusförmige Wechselspannung. Der zeitliche Verlauf dieser Spannung wird durch diese Formel ausgedrückt:

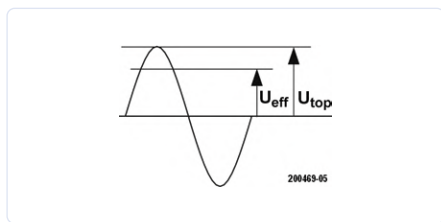


Bild 5. Eine vollständige Periode einer sinusförmigen Wechselspannung.

$$U(t) = U_{\text{top}} \cdot \sin\left(\frac{2 \cdot \pi \cdot t}{T}\right) = U_{\text{top}} \cdot \sin(2 \cdot \pi \cdot f \cdot t) = U_{\text{top}} \cdot \sin(\omega \cdot t)$$

Der höchste Wert, den eine Wechselspannung erreicht, wird wenig überraschend als Spitzenwert bezeichnet. Es wird Sie auch nicht überraschen, dass eine Wechselspannung, die nur periodisch einen Spitzenwert erreicht, weniger „effektiv“ ist (das heißt, weniger Leistung entwickeln kann) als eine Gleichspannung mit diesem Spitzenwert. Um Gleich- und Wechselstrom trotzdem miteinander vergleichen zu können, wurde der Begriff „Effektivwert“ eingeführt:

Der Effektivwert einer Wechselspannung entspricht dem Effektivwert einer Gleichspannung, der in einem ohmschen Widerstand die gleiche Wärmemenge freisetzt wie die entsprechende Wechselspannung.

Zum Beispiel leuchtet eine Glühbirne bei einer Gleichspannung von 12 V genau so hell wie bei einer Wechselspannung von 12 V_{eff}. Achtung: Wenn der Wert einer Wechselspannung angegeben wird, ist dies immer der Effektivwert, sofern nicht ausdrücklich anders angegeben.

An dieser Stelle könnten wir Sie mit einer Menge „höherer“ Mathematik necken (und/oder langweilen), aber es fällt uns weder ein noch in den Rahmen dieses Artikels, den mathematischen Beweis anzutreten. Sie werden uns beim Wort nehmen müssen, dass zwischen dem Spitzenwert und dem Effektivwert einer sinusförmigen Wechselspannung folgende Beziehung besteht:

$$U_{\text{top}} = U_{\text{eff}} \cdot \sqrt{2}$$

Und damit können wir etwas berechnen, was Sie wahrscheinlich schon lange wissen

wollten: An einer Steckdose liegt (in der EU) eine Wechselspannung von 230 V. Wir wissen jetzt, dass dies der Effektivwert ist, aber was ist der Spitzenwert dieser Wechselspannung?

$$U_{\text{top}} = U_{\text{eff}} \cdot \sqrt{2} = 230 \text{ V} \cdot 1,41 = 324,3 \text{ V}$$

Scheitelfaktor

Eine letzte Erläuterung, bevor wir die Theorie (vorerst) hinter uns lassen und uns mehr auf die Elektronik-Praxis konzentrieren. Das Verhältnis des Spitzenwertes zum Effektivwert einer Wechselspannung wird als *Scheitelfaktor* (englisch *crest factor*) bezeichnet:

$$CF = \frac{U_{\text{top}}}{U_{\text{eff}}}$$

Für eine Rechteckspannung wie in Bild 2 ist der Scheitelfaktor gleich 1, für eine Sinusspannung beträgt er $\sqrt{2}$ und für ein Dreieckssignal $\sqrt{3}$. Erwähnenswert ist, dass der Scheitelfaktor für ein Rauschsignal bis unendlich geht (je „zufälliger“ das Rauschen, desto höher der Scheitelfaktor). Es ist nicht wirklich wahrscheinlich, dass Sie dieses Wissen jemals brauchen werden, aber man weiß ja nie...

Zwischen Theorie und Praxis...

Zwischen Theorie und Praxis stehen Schaltpläne und Stücklisten! Man baut ja eine elektronische Schaltung nicht, indem man eine Kiste voller Teile auf die Werkbank kippt und sie auf gut Glück miteinander verbindet. Im Gegenteil: Sie denken darüber nach, was Sie wollen, und skizzieren dann auf Papier ein Schema dessen, was Ihrer Meinung nach die endgültige Schaltung sein wird. Man probiert es zunächst auf einer Experimentierplatine oder so etwas aus, und nach den notwendigen Verbesserungen ergibt sich daraus der endgültige Schaltplan.

Auf der Grundlage dieses Schaltplans entwerfen Sie das Layout der Leiterbahnen auf der Platine plus einen Plan, der zeigt, wo welche Bauteile montiert werden (das Bauteile-Layout). Oder Sie bauen die Schaltung „endgültig“ auf einer Lochrasterplatine auf, dann reicht in der Regel eine einfache Handskizze zur Verdrahtung aus.

Eine Stückliste ist in der Regel ebenfalls nützlich – auch wenn sie nur als Einkaufsliste für den Kauf aller notwendigen Teile im Elektronikladen dient.

Wir werden hier nicht weiter auf all diese Aspekte der Elektronikentwicklung eingehen. Wenn Sie durch diese (oder jede andere) Elektor-Ausgabe blättern, werden Sie viele Beispiele von Schaltplänen, Platinenlayouts, Stücklisten und so weiter finden. Wenn Sie sich einen guten Eindruck davon verschaffen wollen, was die Realisierung eines (kommerziellen) Elektronikprodukts mit sich bringt, verweisen wir Sie auf die ausgezeichnete Artikelserie „Von der Idee zum Produkt“ von Clemens Valens, deren erste Folge in der September/Okttober-Ausgabe 2019 von Elektor erschienen ist.

In der nächsten Folge werden wir über die Bauteile sprechen, mit denen Sie in der Praxis zu tun haben werden – sozusagen die „greifbaren“ Komponenten. ◀

200469-02

Die Artikelserie „Aller Anfang...“ basiert auf dem Elektor-Buch „Basiskurs Elektronik“ von Michael Ebner.

Ein Beitrag von

Idee und Illustrationen: **Michel Ebner**

Text und Redaktion: **Eric Bogers**

Übersetzung: **Rolf Gerstendorf**

Entwurf: **Giel Dols**

Sie haben Fragen oder Anmerkungen?

Haben Sie Fragen oder Anmerkungen zu diesem Artikel? Senden Sie eine E-Mail an den Autor über redactie@elektor.com.



PASSENDE PRODUKTE

➤ **Basiskurs Elektronik (PDF):**
www.elektor.de/basiskurs-elektronik-pdf

Von der Pike auf gelernt

Neues aus der Elektor-Ideenkiste

Zusammengestellt von **Eric Bogers**

Hier sind ein paar kleine Schaltungen, mit denen Sie ihre Löt- und Experimentierfähigkeiten auf Vordermann bringen, wenn Sie mal keine Lust haben, etwas sehr Kompliziertes zu beginnen...



Idee: Michael A. Shustov (Russland)

Pseudo-Thyristor

Ein herkömmlicher Thyristor funktioniert als Schalter recht einfach: Er schaltet sich ein, sobald (kurzzeitig) eine Steuerspannung angelegt wird, aber um ihn wieder auszuschalten, muss die Stromversorgung unterbrochen werden. Der Pseudo-Thyristor in **Bild 1** funktioniert völlig anders. Er wird durch kurzzeitiges Verbinden des Eingangs mit der positiven Versorgungsspannung ein- und dann durch kurzes Verbinden desselben Eingangs mit Masse abgeschaltet. Die Funktionsweise ist nicht schwer zu ergründen. Beim Einschalten der Versorgungsspannung sperrt der Transistor Q4, während Q1 und Q3 leiten und der eigentliche Schalter Q2 sperrt: Die Last (hier in Form des Lämpchens EL1) wird abgeschaltet. LED D1 zeigt an, dass die Schaltung „scharf“ gestellt ist. Wenn der Eingang der Schaltung kurzzeitig durch einen Druck auf den Taster SB1 auf high gelegt wird, geht Q4 in den leitenden Zustand über. Auf diese Weise wird die Basis für Q3 auf Masse gelegt, so dass dieser Transistor sperrt. Infolgedessen geht Q2 in den leitenden Zustand über und schaltet die Last ein. Diese Situation bleibt bestehen, auch wenn SB1

wieder losgelassen wird. Durch einen Druck auf SB2 kehrt die Schaltung in den Ausgangszustand zurück und die Last wird wieder abgeschaltet. Wenn SB1 und SB2 gleichzeitig gedrückt werden, hat SB2 Vorrang: Auch dann wird die Last abgeschaltet.



Idee: Roel Arits (Niederlande)

Funkelnde Jingle Bells

Weihnachten ist eine spannende Zeit - vor allem, wenn die Pakete unter dem Weihnachtsbaum liegen. Mit dieser Schaltung erhöhen wir die Spannung ein wenig - auf bis zu 30 kV, um genau zu sein. Wir benutzen diese 30 kV, um Funken zu erzeugen. Schon das wäre im Dunkeln spektakulär, aber in diesem Fall spielen die Funken auch noch eine Melodie, entweder „Jingle Bells“ oder für die Weihnachtsmuffel „Mission Impossible“. Die überraschend einfache Schaltung ist in **Bild 2** dargestellt. Das Herzstück der Schaltung ist ein Arduino Pro Mini, der ein PWM-Signal (ein gewöhnliches Rechtecksignal) mit einem festen Tastverhältnis von 10% erzeugt. Dieses Rechtecksignal wird auf 12 V angehoben und gepuffert. Und dann wandelt ein Hochspannungstransformator sie in eine Spannung von etwa 30 kV um - hoch genug, um schöne Funken zu erzeugen. Ein gewöhnliches Rechtecksignal macht aber noch keine Musik. Deshalb variiert die Software im Arduino [6] die Frequenz nach Maßgabe der beiden fest programmierten Melodien. Das Ergebnis ist in diesem Video des Autors zu sehen und zu hören [1]. Der Aufbau der Schaltung ist nicht kritisch, nur am Transformator muss etwas gebastelt werden. Die Primärwicklung besteht aus 20 Windungen aus 0,7 mm Kupferlackdraht am Ende von zwei nebeneinander liegenden Ferritstäben mit einem Durchmesser von 10 mm und einer Länge von 80 mm. Fest zusammengedrückt passen diese beiden Stäbe

genau in die Öffnung einer fertig gewickelten 16-kV-Spule, die als Sekundärwicklung dient (siehe **Bild 3** und [2]). Hinweis: Nach der Online-Veröffentlichung dieser Schaltung stellte sich heraus, dass die vom Autor verwendete Spule nicht mehr verfügbar ist. Eine Alternative ist [3], die sogar bereits mit einer Primärwicklung versehen ist, so dass Sie diese nicht mehr selbst herstellen müssen, eine andere Möglichkeit ist [4], aber dann müssen Sie die Primärwicklung selbst wickeln. Schließlich ist [5] eine weitere Möglichkeit, aber in diesem Fall muss die Spannung für die Primärwicklung möglicherweise gesenkt werden. Und in allen Fällen sollte das optimale Tastverhältnis für maximal starke Funken, ohne den Kern in die Sättigung zu treiben, ermittelt werden. Und ja: Diese Schaltung ist ein primitiver Funkensender und mag als solcher illegal sein, aber solange Sie sie nicht ständig benutzen, werden Sie wahrscheinlich keine Probleme bekommen. Mit einem kleinen AM-Empfänger können Sie prüfen, bis zu welcher Entfernung die Störungen noch wahrnehmbar sind. Aber Sie sind gewarnt! Bevor wir es vergessen: Die Software für dieses Projekt und eine größere Version von Bild 2 können kostenlos von der Projektseite des Artikels [6] heruntergeladen werden.



Idee: Elektor-Labor

Lügendetektor

Echte Lügendetektoren verwenden eine Vielzahl von Sensoren, die am Körper der befragten Person angebracht werden und Parameter wie Blutdruck, Herzfrequenz, Atmung, Hautwiderstand und Temperatur messen. Der Detektor schreibt diese Werte auf einen langen Papierstreifen und aus diesen Daten lässt sich angeblich ableiten, ob die betreffende Person die Fragen wahrheitsgetreu beantwortet hat oder nicht. Der Einsatz von Lügendetektoren ist in vielen Ländern zu

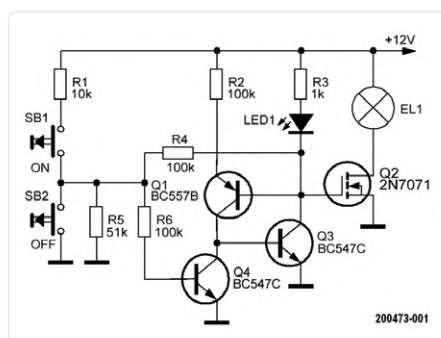


Bild 1. Mit wenigen Transistoren kann ein Pseudo-Thyristor gebaut werden, der bessere Eigenschaften als ein „echter“ Thyristor aufweist.

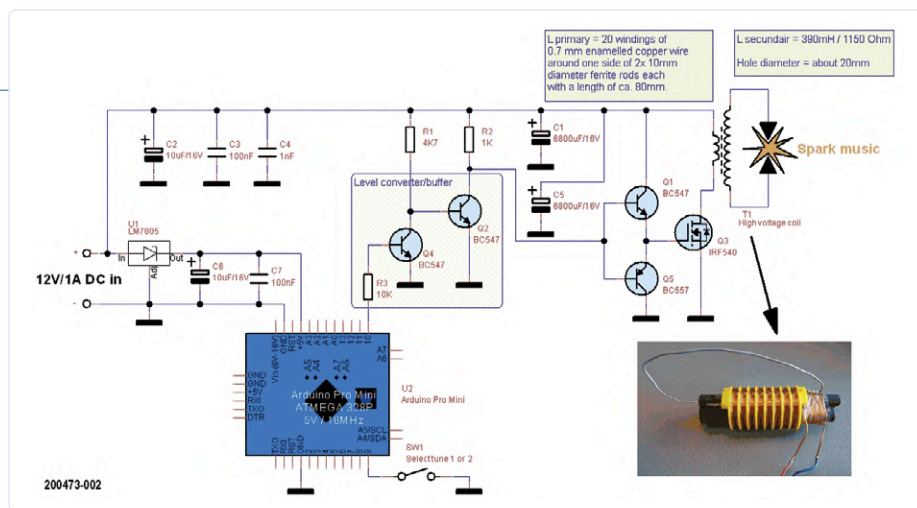


Bild 2. Das Herzstück des musikalischen Funkenerzeugers bildet ein Arduino Mini Pro.

Bild 3. Die Primärwicklung ist fest auf zwei Ferritstäben aufgebracht, die wiederum in die fertig gewickelte Hochspannungsspule passen.

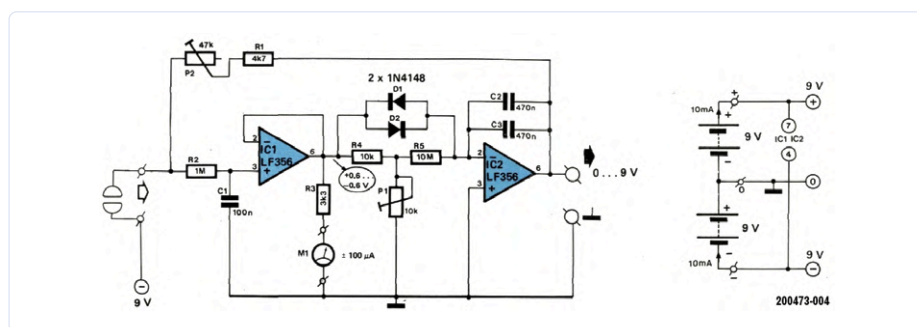


Bild 4. Ein einfacher Lügendetektor, der Veränderungen des Hautwiderstands anzeigt.

Recht umstritten. Hier ist eine einfache Schaltung, die Veränderungen des Hautwiderstands misst und die auf spielerische Weise als Lügendetektor verwendet werden kann. Ein gewöhnliches Ohmmeter ist für unseren Zweck nicht geeignet, denn wir wollen nicht den absoluten Wert des Hautwiderstands messen, sondern die Veränderung desselben. Eine Schaltung, mit der das möglich ist, ist in **Bild 4** dargestellt. Auf der linken Seite des Schaltbildes sehen wir zwei Elektroden, die auf dem Handrücken und auf dem entsprechenden Unterarm angebracht sind. An der „unteren“ Elektrode liegt eine negative Spannung von 9 V an. Über die Haut (Widerstand) und R2 wird diese Spannung an den extrem hochohmigen,

nicht-invertierenden Eingang des Opamps IC1 gelegt. Durch diesen Eingang fließt nahezu kein Strom, so dass der über oder durch die Haut fließende Strom durch P2 und R1 zu IC2 fließen muss. Dieser Opamp stellt seine Ausgangsspannung so ein, dass die „obere“ Elektrode immer auf einem Potential von 0 V liegt. Der hochohmige R5 und die Parallelschaltung von C2 und C3 sorgen für eine gewisse Verzögerung. Große Spannungsdifferenzen werden von beiden Dioden schnell beseitigt, bei kleinen Spannungsdifferenzen kann es bis zu 30 s dauern, bis die Spannung an der oberen Elektrode wieder auf 0 V zurückkehrt. Auf diese Weise kann man im Drehspulinstrument M1 (mit Nullpunkt in Skalenmitte) eine Änderung

Sie haben Fragen oder Kommentare?

Haben Sie Fragen oder Anmerkungen zu diesem Artikel? Bitte senden Sie eine E-Mail an den Autor über redactie@elektor.com.

des Hautwiderstandes bequem ablesen. R1 und C2/C3 bilden zudem ein Tiefpassfilter, um aufgefangene (Brumm-)Störungen zu beseitigen. Beim Abgleich können wir uns kurz fassen: Befestigen Sie beide Elektroden (wie diese ausgebildet werden, überlassen wir Ihrer Fantasie) auf der Haut der Testperson und schließen Sie ein Multimeter an den Ausgang der Schaltung an (Bereich: 10 V_{DC}). Drehen Sie dann P2, bis das Messgerät eine Spannung von 2...6 V anzeigt (warten Sie, bis der Messwert stabil ist). Dadurch wird der Detektor auf den Hauttyp der Testperson eingestellt. Wenn wir jetzt leicht auf die Elektroden drücken, sollte M1 einen Ausschlag zeigen. Wenn dies passiert, ist der Detektor einsatzbereit.

Warnung: Diese Schaltung darf nur mit zwei 9-V-Blockbatterien betrieben werden. Verwenden Sie unter keinen Umständen eine Netzspannungsversorgung! ⚡

200473-03

WEBLINKS

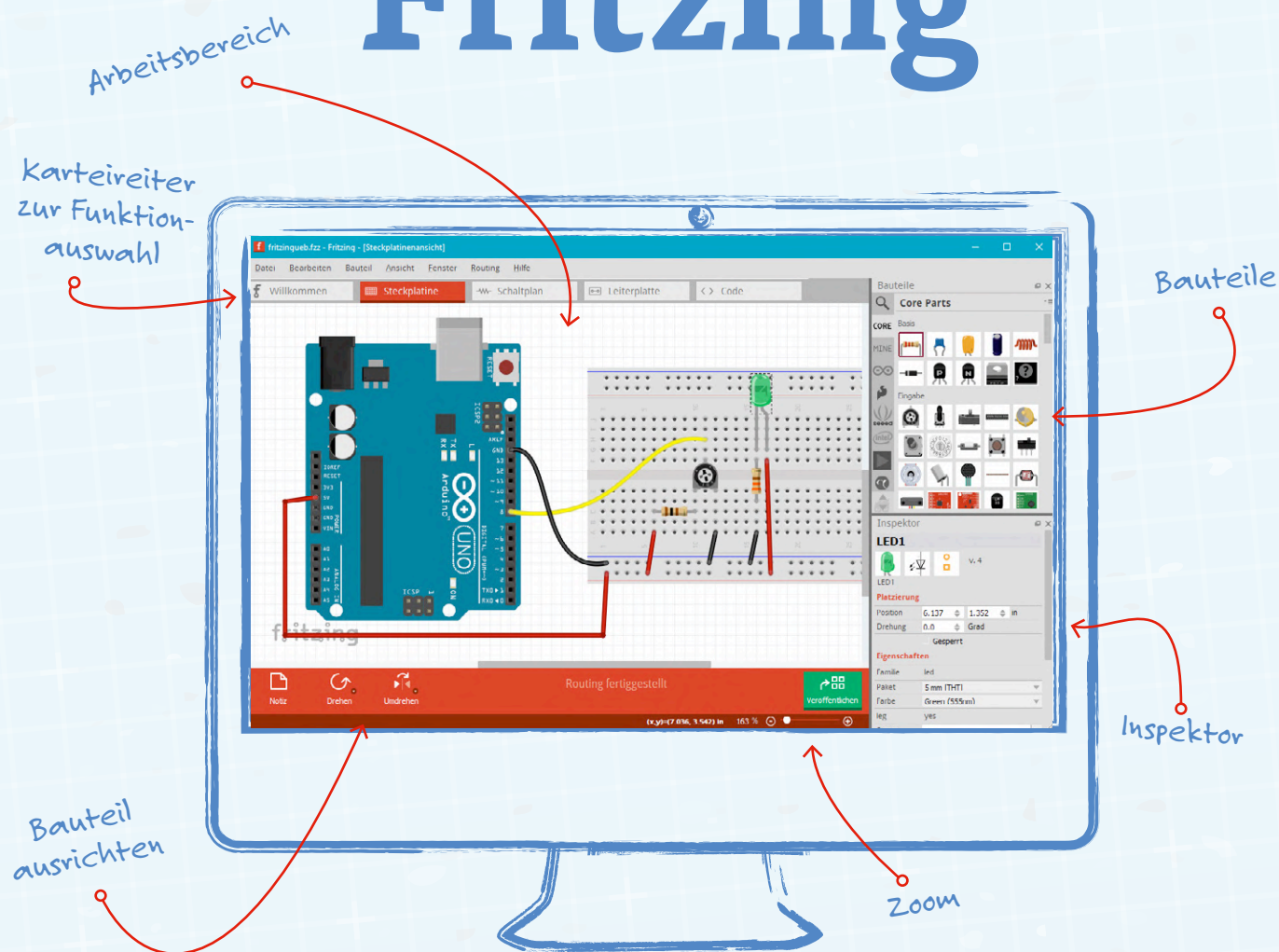
- [1] Video Jingle Bells: www.youtube.com/watch?v=NGSjcOffbul&feature=youtu.be
- [2] Originale HV-Spule: <https://bit.ly/3lRr8R7>
- [3] Alternative HV-Spule #1: <https://bit.ly/35crVGo>
- [4] Alternative HV-Spule #2: <https://bit.ly/2FfsCU4>
- [5] Alternative HV-Spule #3: <https://amzn.to/3jPcAiX>
- [6] Projektseite zu diesem Artikel: www.elektormagazine.de/200473-03



PASSENDE PRODUKTE

➤ **Englisches Buch**
„Electronic Circuits For All“
www.elektor.de/electronic-circuits-for-all

Breadboard-ansichten mit Fritzing



Von **Florian Schäffer**

Von einem Schaltplan (siehe letzte Ausgabe) zu einem ersten eigenen Aufbau auf einem Steckbrett ist es kein einfacher Weg. Denn die abstrakten Formen müssen den realen Bauteilen und Anschlüssen zugeordnet werden. Mit Fritzing erstellen Sie realistisch anmutende Aufbauvorschlge und erleichtern den Nachbau.



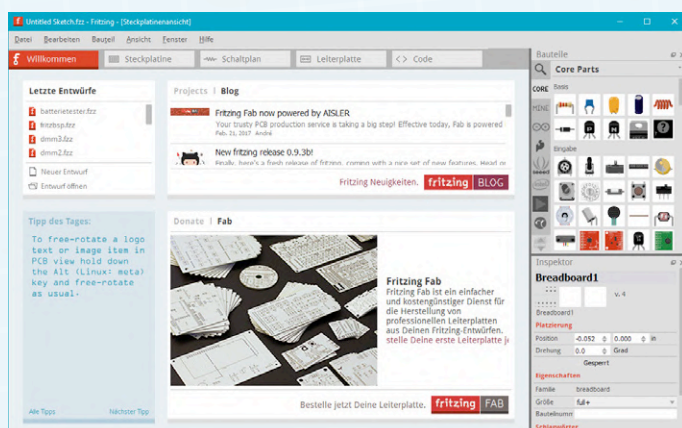
Ein Foto vom realen Aufbau anzufertigen wäre im Prinzip auch eine gute Idee. Beleuchtung und Positionierung erfordern aber einiges Geschick und das Ergebnis ist trotzdem oft nicht gut genug als Nachbauanleitung. Mit Fritzing können Sie Breadboardansichten erstellen, die übersichtlich und ästhetisch ansprechend sind. Die grafisch auf das wesentliche reduzierten Bauteile verdeutlichen die entscheidenden Anschlüsse. Auch im Vorfeld erweist es sich als praktisch, die Schaltung zuerst auf dem virtuellen Reißbrett aufzubauen, um so die optimale Platzierung auszuprobieren.

Alternative Software

Fritzing ist seit seinem Erscheinen sehr beliebt, weil es kostenlos und einfach zu bedienen ist und gefällige Ergebnisse produziert. Das Programm bietet noch weitere Funktionen, um Schaltpläne, Leiterplatten und Quellcodes zu erstellen - diese Programmteile sind allerdings bei weitem nicht so praktisch. In der nächsten Ausgabe lernen Sie Tinkercad Circuits kennen, um Schaltungen zu simulieren, mit dem sich ebenso sehr schöne Grafiken von Schaltungsaufbauten erstellen lassen. Der wesentliche Unterschied ist allerdings, dass für Fritzing jeder Anwender neue Bauteilgrafiken erzeugen kann, um bisher fehlende Objekte zu ergänzen, was bei Tinkercad Circuits nicht möglich ist.

Das Programm steht kostenlos zur Verfügung, allerdings verbirgt sich der Download seit Ende 2019 hinter der Aufforderung, Geld zu spenden. Um dies zu umgehen (was völlig legal ist), können Sie eine alte Version im Internet suchen und installieren (Sie werden beim ersten Start dann zum Update geleitet) oder direkt das Download für die derzeit aktuelle Version 0.9.4 über den Link <https://fritzing.org/download/api/1.0/update/344> starten.

Nach der Installation starten Sie Fritzing, woraufhin stets nach Updates für das Programm und die Bauteilbibliotheken gesucht wird. Als Sprache für die Oberfläche wird automatisch die Systemsprache benutzt – also in den meisten Fällen Deutsch. Anschließend sehen Sie den Startbereich. Am oberen Rand gibt es Karteireiter für die verschiedenen Funktionsbereiche. Wir nutzen ausschließlich die Ansicht STECKPLATINE, auf die Sie dazu wechseln.

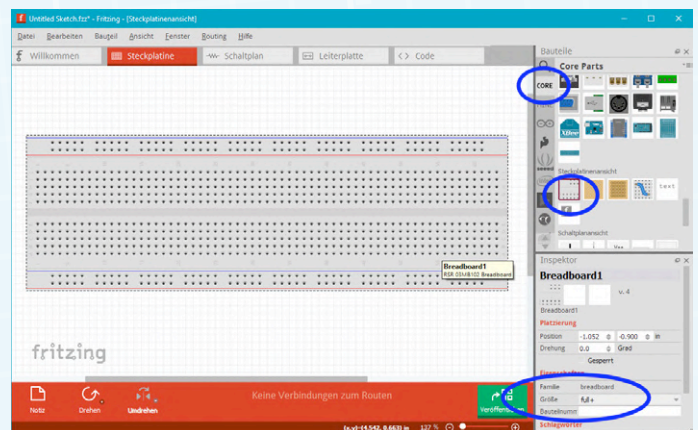


So präsentiert sich Fritzing nach dem Start. Wechseln Sie zur Ansicht STECKPLATINE.

Exportformate

Fritzing bietet den Export in verschiedene Dateiformate. Die eigenen Dateien werden über DATEI/SPEICHERN mit dem Dateifuffix .fzz gesichert. Diese Dateien können später weiter bearbeitet werden. Über DATEI/EXPORTIEREN stehen weitere Formate für unterschiedliche Zwecke zur Verfügung. Interessant sind vor allem die Formate bei ALS BILD. Mit PNG erzeugen Sie eine Bitmap-Grafik, die sich gut eignet, um auf Webseiten genutzt zu werden. Im Gegensatz zu JPG treten dabei keine Qualitätsverluste auf. Das Vektorformat eignet sich gut, wenn die Grafik später skaliert oder mit einem Grafikprogramm weiter bearbeitet werden soll. Allerdings interpretiert jedes Grafikprogramm SVG-Dateien etwas anders und so kann es zu Unterschieden bei der Darstellung kommen. Oft fehlen sogar ganze Bauteile, weil die Bibliotheken in Fritzing nicht korrekt angelegt wurden. Von daher ist es besser, ein PDF zu erzeugen und dann dieses zu bearbeiten, wozu ebenfalls viele Vektorgrafikprogramme in der Lage sind.

Breadboard auswählen

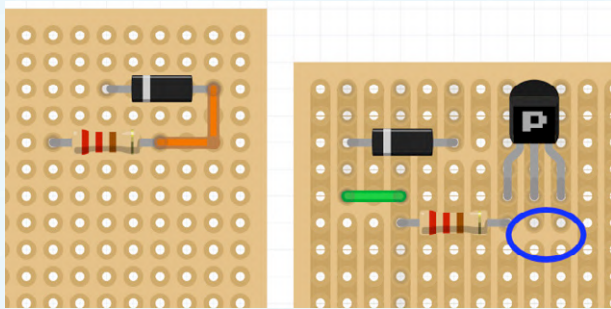


Am Anfang wird stets eine große Steckplatine bereitgestellt, deren Größe Sie aber ändern können. Mit dem Mausekranz können Sie die Ansicht zoomen und mit der mittleren Maustaste verschieben.

Das große Breadboard, das anfangs immer zu sehen ist, können Sie löschen oder ändern. Immer wenn ein Bauteil durch Anklicken ausgewählt wurde, werden dessen Attribute rechts im INSPEKTOR gezeigt. Bei GRÖÖE können Sie alternative Typen auswählen, wenn Sie andere Boards brauchen. Möchten Sie ein (weiteres) Board einfügen, dann finden Sie diese in der Liste BAUTEILE, in der Rubrik CORE im Bereich STECKPLATINENANSICHT: Ziehen Sie neue Bauteile direkt auf den Arbeitsbereich.

Platinenansichten entwerfen

Relativ unbekannt ist, dass sich mit Fritzing auch sehr gut Loch- und Streifenrasterplatinen gestalten lassen, um so zum Beispiel vor dem Löten auszuprobieren, wie die Bauteile optimal angeordnet werden und wo Leiterbahnunterbrechungen und Drahtbrücken notwendig sind. Neben dem Symbol für Breadboards in der Bauteilliste gibt es zwei braune Platinensymbole. Mit dem ersten erzeugen Sie eine Platine, die nur aus Lötläugen besteht. Das zweite Symbol ist für eine Streifenrasterplatine. Im Inspektor können Sie den Typ aber jederzeit wechseln und auch die Platinengröße anpassen.

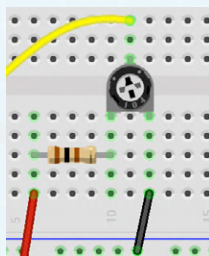


Eine Lochrasterplatte (Prefboard) und ein Streifenraster (Stripboard). Bei dieser können die Leiterbahnen aufgetrennt werden.

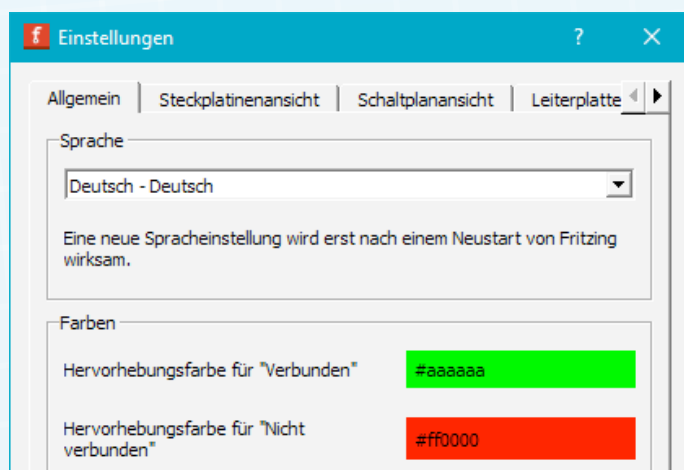
Der Clou ist, dass bei der Streifenrasterplatte die längsverlaufenden Leiterbahnen aufgetrennt (und wieder geschlossen) werden können, in dem Sie zwischen zwei Lötäugen klicken – eventuell stark heranzoomen, um besser treffen zu können. In der Praxis werden solche Auftrennungen mit einem manuellen Leiterbahnunterbrecher oder einem kleinen Bohrer (der etwas breiter als die Leiterbahn ist) vorgenommen.

Praktische Einstellungen

Um Fehler leichter zu erkennen, werden Verbindungen, die durch die Federkontakte des Breadboards erzeugt werden, grün hervorgehoben. Bei der Veröffentlichung von Grafiken kann das optisch stören.

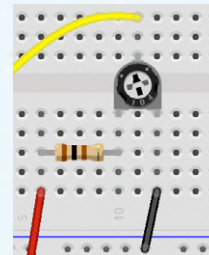


Grün gekennzeichnet sind die Verbindungen der Federkontakte im Steckbrett.



Fenster mit den möglichen Einstellungen, um Fritzing anzupassen.

Über BEARBEITEN/EINSTELLUNGEN können Sie das Programmverhalten ein wenig anpassen. Auf der Registerkarte ALLGEMEIN können Sie die Farben anpassen: Klicken Sie auf die grüne Fläche bei HERVORHEBUNGSFARBE FÜR „VERBUNDEN“ und anschließend auf EIGENE FARBE WÄHLEN, um beispielsweise ein mittleres Grau einzustellen, welches nicht so stark auffällt.



Mit Grau als Hervorhebungsfarbe ist die Kennzeichnung fast nicht zu sehen.

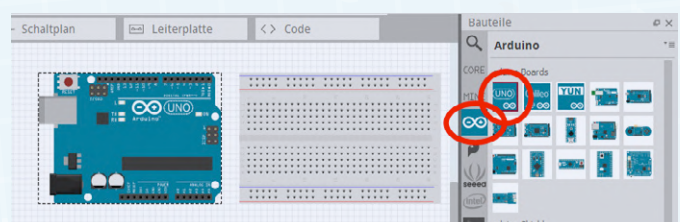
Normalerweise werden Drahtverbindungen immer als gerade Linie und mit Knicken erzeugt. Für viele Verbindungen sind gebogene Leitungen aber ansprechender – vor allem weil die geraden Linien oft nicht rechtwinklig angelegt werden oder dies viel Aufwand erfordert. Sie können in den Einstellungen auf der Registerkarte STECKPLATINENANSICHT die Option GEBOGENE DRÄHTE UND BEINCHEN aktivieren und dann sowohl gerade Linien, als auch gebogene erzeugen.

Beispielübung

Zum ersten Ausprobieren, soll die Darstellung nachgezeichnet werden, die Sie am Anfang dieses Artikels finden – die Funktion ist dabei hier nebensächlich.

1. Wählen Sie das Breadboard aus und ändern Sie im INSPEKTOR den Typ, so dass die abgebildete Version gezeigt wird.

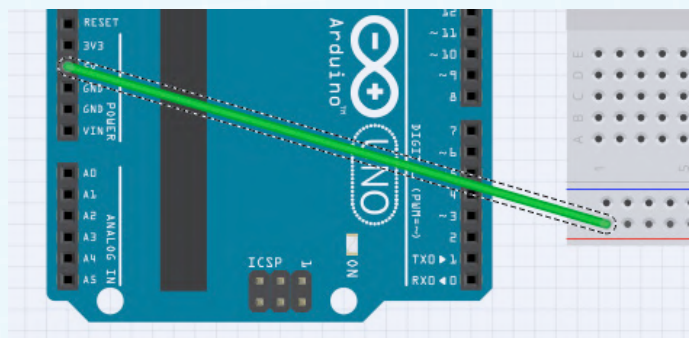
2. Bei BAUTEILE wechseln Sie in die Rubrik, die das Arduino-Logo zeigt. Nachdem alle Bauteile geladen wurden, ziehen Sie das erste Symbol („Uno“) auf den Arbeitsbereich, so dass dort ein Arduino Uno links neben dem Steckbrett liegt.



3. Solange das Arduino-Board ausgewählt ist, können Sie es verschieben und mit dem Symbol am unteren Fensterrand drehen. Drehen Sie es, bis die USB-Buchse nach oben zeigt.



4. Verbindungsleitungen werden immer dann gezeichnet, wenn Sie mit dem Mauszeiger an einem Kontaktpunkt ansetzen. Dies kann ein Ende eines Bauteilbeinchens sein, eine Buchse auf dem Breadboard oder ein Pin oder eine Buchse auf einer Platine wie die des Arduino Uno. Ziehen Sie vom Anschluss 5 V des Arduino eine Leitung zur roten Verteilerleiste auf dem Steckbrett.

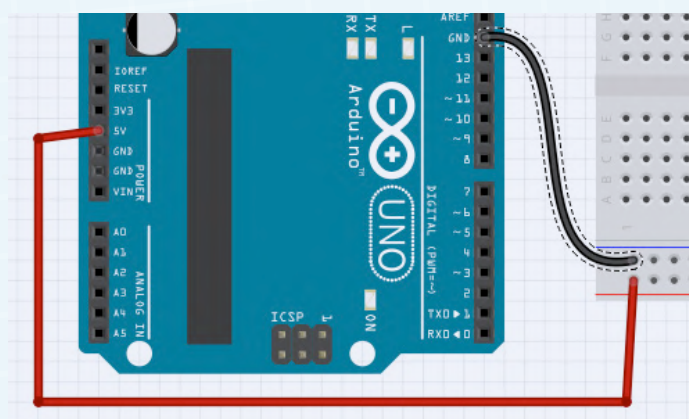


5. Die Farbe der Linie ist beim Zeichnen beliebig (meistens Blau). Klicken Sie die Linie einmal an und Sie können im INSPEKTOR die FARBE ändern.

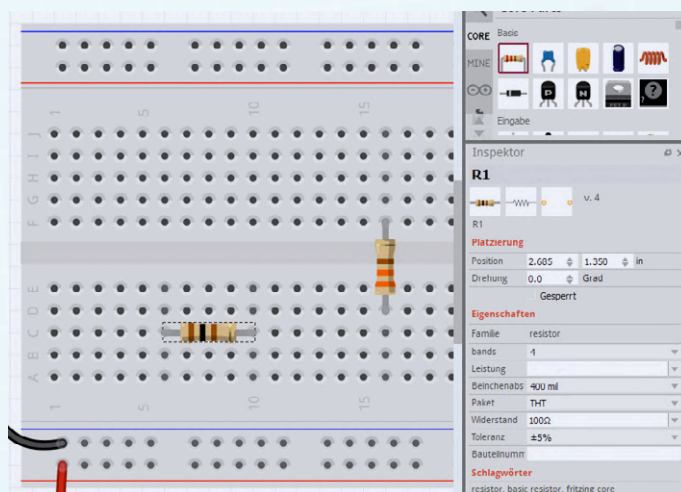
6. Wollen Sie einen Knickpunkt einfügen, um den Verlegepfad der Verbindung zu ändern, klicken Sie doppelt auf eine Stelle der Linie. Es wird ein Knickpunkt eingefügt, den Sie verschieben können. So können Sie beliebig viele Knicke einfügen. Per Rechtsklick auf einen Biegepunkt öffnet sich das Kontextmenü, in dem Sie verschiedene Auswahlmöglichkeiten finden, um die Biegepunkte etc. zu modifizieren.

7. Ziehen Sie eine gerade Verbindung vom rechten GND-Anschluss des Arduino zur blauen Verbindungsschiene auf dem Steckboard und ändern Sie die Farbe in Schwarz.

8. Möchten Sie eine gerade Linie in eine geschwungene Verbindung ändern, dann greifen Sie an einer beliebigen Stelle die Leitung und ziehen sie in eine Richtung. Dadurch erzeugen Sie an dem Punkt eine Bézierkurve. Sie können bis zu zwei Biegepunkte auf diese Weise an der Verbindung erzeugen.



9. Grundsätzlich lassen sich die meisten BAUTEILE auf die gleiche Weise platzieren. Je nach Bauteil gibt es aber verschiedene Einstellungen im INSPEKTOR, um beispielsweise die Farbe, Größe oder Dimension anzupassen. Die gängigsten Bauteile finden Sie in der Rubrik CORE. Die anderen Rubriken bieten oft Module der gängigen Hersteller für Arduino-Zubehör. Wenn Sie ein Bauteil nicht auf Anhieb finden, können Sie auf die Lupe klicken und nach einem (englischen) Begriff suchen. Fügen Sie einen Widerstand ein und schieben Sie ihn an die gewünschte Position. Im Inspektor wählen Sie, ob es ein Kohleschicht-Widerstand mit drei Ringen (BANDS) oder einer aus Metallfilm mit vier Ringen sein soll. Stellen Sie weiterhin den Widerstandswert (Sie können auch Werte eingeben, die nicht aufgelistet sind) und die Toleranz ein, damit die farbigen Ringe passend gezeigt werden.



10. Anschließend fügen Sie das Trimpotentiometer und die LED ein. Bei der LED achten Sie darauf, dass die Seite mit dem geknickten Bein (Anode/Plus) rechts liegt. Sie können die Bauteile teilweise auch horizontal spiegeln, indem Sie unten am Rand UMDREHEN anklicken. Für welche Wellenlänge Sie sich bei der LED entscheiden, hat hier lediglich Auswirkungen auf die Farbe in der Grafik.

11. Nun fehlen nur noch einige Verbindungen zwischen den Buchsen und dem Arduino.

Der Artikel wurde dem Elektor-Sonderheft „Einstieg in die Elektronik mit Arduino“ entnommen, das im Elektor-Shop bestellt werden kann: www.elektor.de/elektor-special-einstieg-in-die-elektronik-mit-arduino. Ab Oktober ist das Special auch in englischer Sprache erhältlich.

200225-B-01



Design analoger Filter (Teil 2)

Von **Alfred Rosenkränzer**

Im diesem zweiten Teil der Artikelserie zur Entwicklung analoger Filter geht es um aktive Varianten. Die in passiven Filtern verwendeten Spulen werden hier konsequent durch aktive Bauteile - in der Regel Operationsverstärker - ersetzt. Dies hat einige Vorteile, wie nachfolgend klar wird.

Ein Beitrag von

Idee, Schaltungen und Text: **Alfred Rosenkränzer**
Schaltpläne: **Patrick Wielders**

Redaktion: **Dr. Thomas Scherer**
Layout: **Giel Dols**

Spulen erfreuen sich bei den meisten Elektronikern nicht unbedingt großer Beliebtheit. Kein Wunder, denn die Eigenschaften von Spulen sind alles andere als ideal: Da sie aus einem aufgewickelten Draht bestehen, haben sie einen nicht ignorierbaren ohmschen Widerstand, der vom Querschnitt und der Länge des verwendeten Drahts abhängt. Dazu kommen noch Widrigkeiten wie interne Koppelungskapazitäten, die zu unerwünschten Resonanzen führen können. Außerdem erfordern tiefe Frequenzen große Induktivitäten und

damit auch physikalisch große Spulen, was sie teuer und sensibel für externe Magnetfelder macht. Sie können unter Umständen sogar selbst unerwünschte Streufelder generieren. Üblich sind große Spulen heute vor allem in Frequenzweichen passiver Lautsprecherboxen (**Bild 1**).

Aktive Filter

In aktiven Filtern kommen Spulen nur selten vor und wenn, dann in kleinen Bauformen. Was ein Filter „aktiv“ macht, sind aktive Bauelemente: Halbleiter in Form

von Transistoren (selten) oder Operationsverstärker (meistens). Ohne Spulen und dank der Möglichkeit mehrstufiger Filter, bei denen sich die einzelnen Zweige nicht durch Belastung gegenseitig beeinflussen, verhalten sich aktive Filter deutlich idealer.

Als aktives Element genügt im einfachsten Fall ein Transistor in Kollektorschaltung als Puffer. Mit den Opamps ist aber nicht nur das Design einfacher, sondern es lassen sich für bestimmte Zwecke leicht Verstärkungen > 1 in Filtern realisieren.

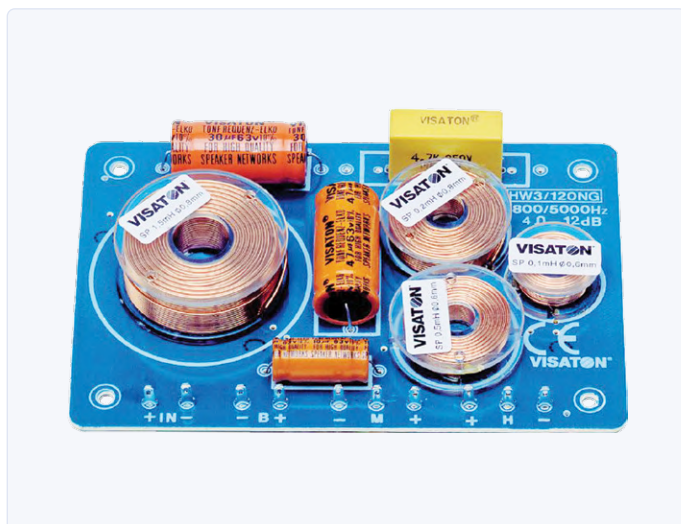


Bild 1. Lautsprecher-Frequenzweichen sind passive Filter mit dicken Spulen. Hier das Modell HW3/120NG von Visaton. Bild: Visaton.

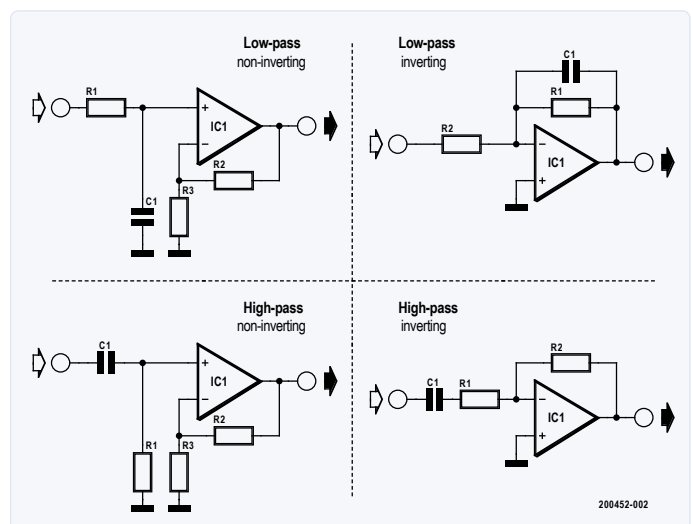


Bild 2. Prinzipschaltungen von Hoch- und Tiefpässen 1. Ordnung in invertierender und nichtinvertierender Variante.

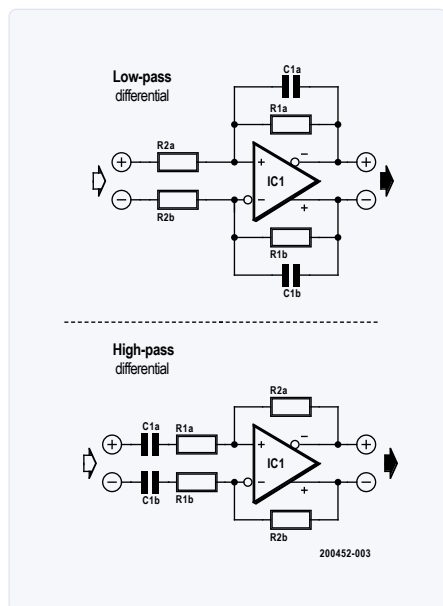


Bild 3. Prinzipschaltungen von differentiellem Hoch- und Tiefpass 1. Ordnung.

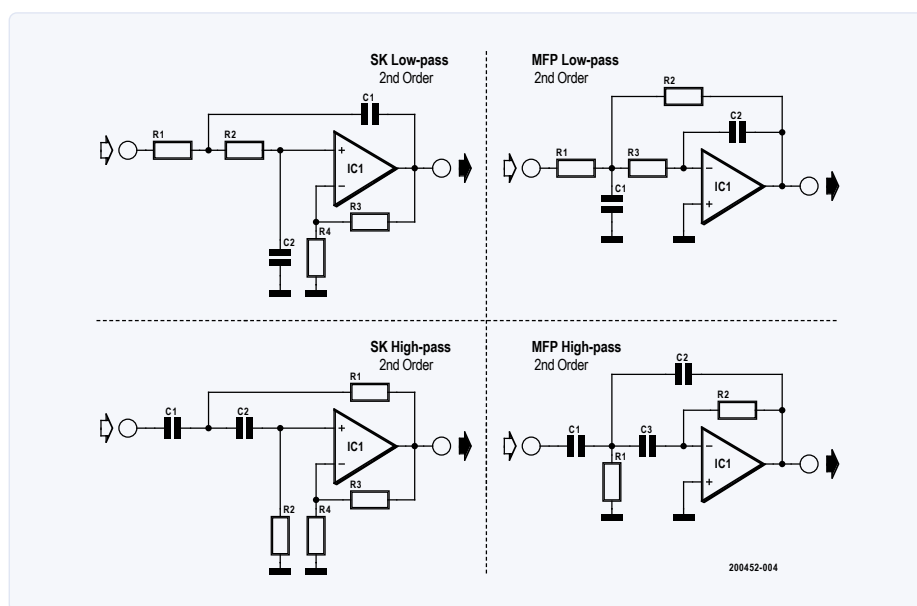


Bild 4. Prinzipschaltungen von Hoch- und Tiefpässen 2. Ordnung in Sallen-Key- (SK) und Multiple Feedback-Ausführung (MFB).

Filterart

Wie bei den passiven Varianten kann man aktive Filter ebenfalls als Tief-, Hoch-, All- und Bandpässe sowie Bandsperrn bauen. Da die Filterarten schon in Teil 1 [1] beschrieben wurden, wird zur Redundanzvermeidung darauf verwiesen.

Approximationen

Genau wie bei passiven Filtern können aktive Filter ebenfalls mit Bessel-, Butterworth-, Tschebyscheff-Charakteristik (und weitere) konzipiert werden. Die Realisierung von Cauer-Filtern und Invers-Tschebyscheff-Filtern mit ihren Notches im Sperrbereich ist zwar nicht prinzipiell unmöglich, doch wäre das aufgrund des hohen Aufwands und des begrenzten Nutzens nicht praktikabel.

Stattdessen wird zum Beispiel für Frequenzweichen im Audiobereich gerne ein Filter verwendet, das unter dem Namen seines Erfinders Siegfried Linkwitz bekannt ist. Es ermöglicht einen geraden Summenfrequenzgang der Tief, Mittel und Hochpasszweige einer Frequenzweiche. Ein Linkwitz-Filter 4. Ordnung besteht aus zwei hintereinander geschalteten Butterworth-Filtern 2. Ordnung – das ist nicht das Gleiche wie ein Butterworth-Filter 4. Ordnung! Bei einem Linkwitz-Filter ist zudem die Grenzfrequenz beim -6-dB-Punkt definiert, und nicht wie

sonst üblich bei -3 dB. Obwohl sich ein flacher Amplitudenfrequenzgang erreichen lässt, ist die Gruppenlaufzeit alles andere als eben. Dadurch gibt es besonders bei Rechtecksignalen deutliche Impulsverformungen.

Struktur und Ordnung

Bild 2 zeigt die Prinzipschaltungen von Tief- und Hochpassfiltern 1. Ordnung mit Opamps. Die beiden linken Filter bestehen aus einem RC-Tief- bzw. Hochpass, gefolgt von einem Opamp als Puffer in nicht invertierender Beschaltung. Die beiden Widerstände R2 und R3 definieren in althergebrachter Weise die Verstärkung. Die Filterfunktion wird durch die Verstärkung nicht beeinflusst. Die Eingangsimpedanz wird in beiden Fällen durch die Kombination von R und C gebildet. Der positive Eingang eines Opamps ist ja sehr hochohmig und kann daher vernachlässigt werden. Der Opamp-Ausgang ist gleichzeitig der Ausgang des Filters und damit sehr niederohmig. Eine eventuell folgende Filterstufe kann daher rückwirkungsarm direkt an diesen Ausgang angeschlossen werden.

Die beiden Filter auf der rechten Seite basieren auf einem invertierenden Opamp. Hier sind Verstärkung und Grenzfrequenz nicht unabhängig voneinander einstellbar. Die Grenzfrequenz ist von R1 abhängig und die Verstärkung vom Verhältnis der beiden Widerstände. Das von mir benutzte Filter-

Tool kann die invertierende Variante nicht berechnen. Dies wäre auch redundant, da sich beide Hoch- bzw. Tiefpässe gleich verhalten und sich die Grenzfrequenz immer zu $f_g = 1 / (2 * \pi * R1 * C1)$ ergibt.

Wie bereits erwähnt, kann man nicht so ohne Weiteres einen Bandpassfilter oder eine Bandsperr erster Ordnung bauen. Eine Art Bandpass erster Ordnung würde sich aber mit einer Reihenschaltung aus Tief- und Hochpass (oder umgekehrt) realisieren lassen. Mit wenig steilflankigen Filtern und nah beieinander liegenden Grenzfrequenzen lassen sich aber keine schmalen Bänder ausfiltern. Der Vorteil bei diesen Verfahren ist, dass man untere und obere Grenzfrequenz getrennt einstellen kann. Aber Achtung: Ist $f_{g_{HP}} > f_{g_{TP}}$, dann ergibt sich leider keine Bandsperr, sondern ein Breitbandabschwächer mit „schrägem“ Frequenzgang.

Differentielle Filter

Sowohl in der Audiotechnik als auch bei höheren Frequenzen mit mehreren hundert MHz werden zunehmend voll differentielle Verstärker eingesetzt (Fully Differential Amps). Auch damit kann man aktive Filter bauen, wie **Bild 3** beweist. Allerdings klappt das so nur in einer Topologie. Wenn es interessiert: Für IC1 könnte man z.B. den differentiellen Opamp LME48724 einsetzen. Für aktive Filter zweiter Ordnung gibt es zwei unterschiedliche Strukturen: Den

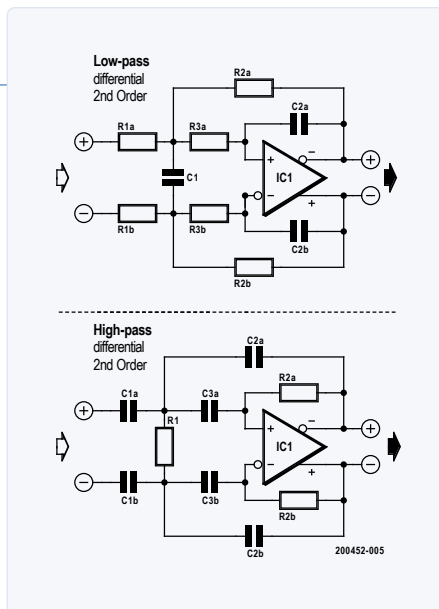


Bild 5. Prinzipschaltungen von differentiellm Hoch- und Tiefpass 2. Ordnung. Sie sind nur als MFB-Typ möglich.

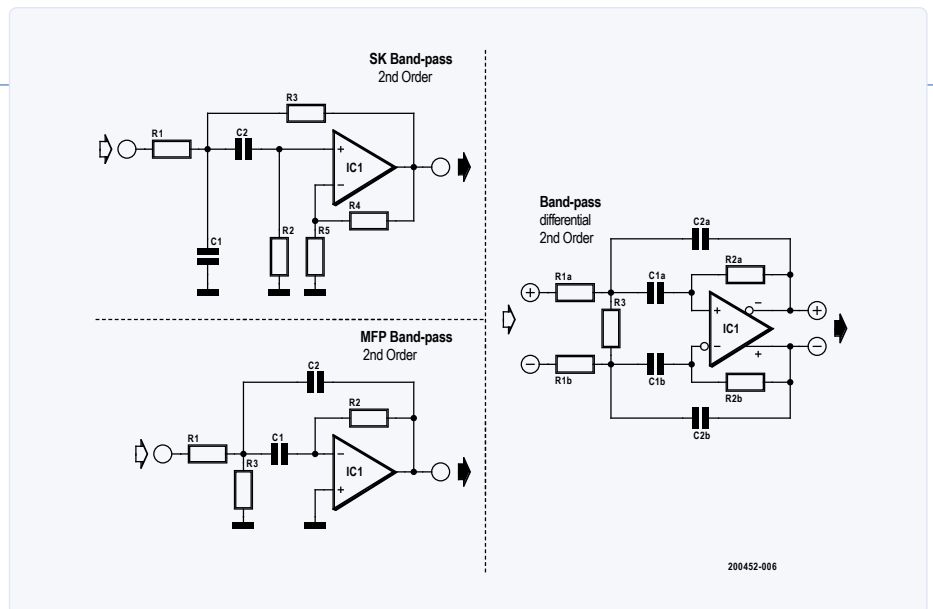


Bild 6. Prinzipschaltungen von Bandpässen 2. Ordnung.

nach seinen Erfindern benannte Sallen-Key-Filter (SK) und eine als „Multiple Feedback“ (MFB) bekannte Anordnung mit mehrfacher Rückkoppelung. **Bild 4** zeigt beide Strukturen für einen Tief- und Hochpass. Im Gegensatz zum SK-Filter invertiert ein MFB-Filter das Signal. Die Verstärkung im Durchlassbereich wird beim SK-Hoch- und Tiefpass durch das Verhältnis von R_3 und R_4 bestimmt. Bei MFB-Tiefpass ergibt sich die Verstärkung im Durchlassbereich durch das Verhältnis von R_1 und R_2 – Beim MFB-Hochpass allerdings

ist die Durchlassverstärkung vom Verhältnis von C_1 und C_2 abhängig.

Differentielle aktive Filter 2. Ordnung können nur als MFB-Struktur realisiert werden (siehe **Bild 5**). Die Prinzipschaltungen von Bandpässen sind in **Bild 6** zu sehen. Der differentielle Bandpass ist vom MFB-Typ.

Bandpässe höherer Ordnung

Bandpässe ab 4. Ordnung kann man direkt als einzelne Schaltung oder als Kombination

von Tief- und Hochpass entwerfen. Letzteres hat wieder den Vorteil, dass man beide Filterkanten getrennt einstellen kann.

Man kann bei solchen seriellen Anordnungen sogar Hoch- und Tiefpässe verschiedener Ordnung kombinieren. Dies eignet sich zum Beispiel gut für die Filter eines Fledermausdetektors: Die relativ lauten „tiefen“ Frequenzen $< 20 \text{ kHz}$ kann man mit einem Hochpass 4. Ordnung unterdrücken. Ein Tiefpass 2. Ordnung genügt, um Frequenzen $> 80 \text{ kHz}$ abzublocken.

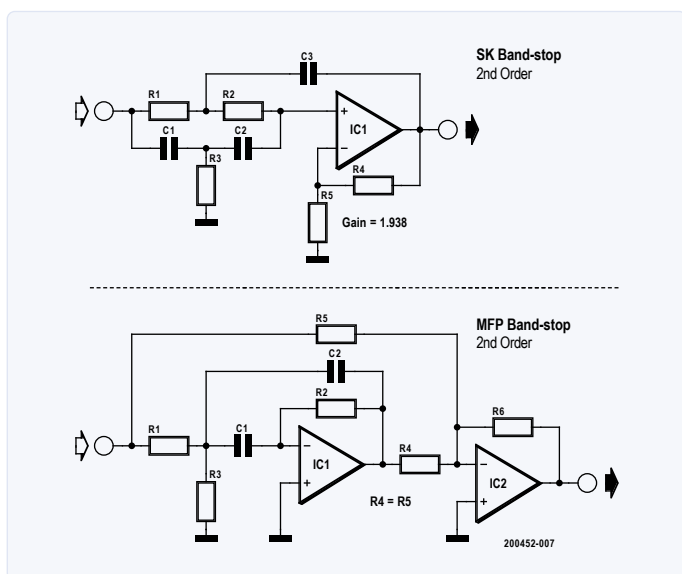


Bild 7. Prinzipschaltungen von Bandsperren 2. Ordnung.

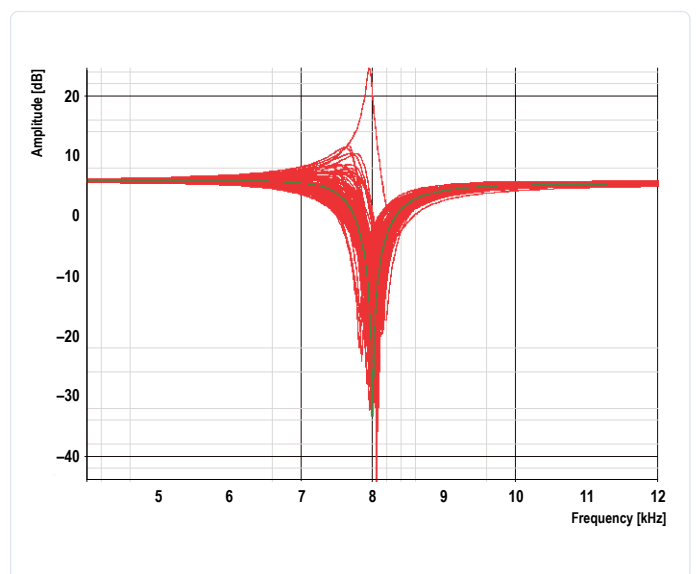


Bild 8. Simulierter Frequenzgang der SK-Bandsperre 2. Ordnung. Die grüne Kurve entspricht dem idealen Verlauf. Der Schwarm roter Kurven entstand durch die Monte-Carlo-Simulation innerhalb der angegebenen Bauteiltoleranzen.

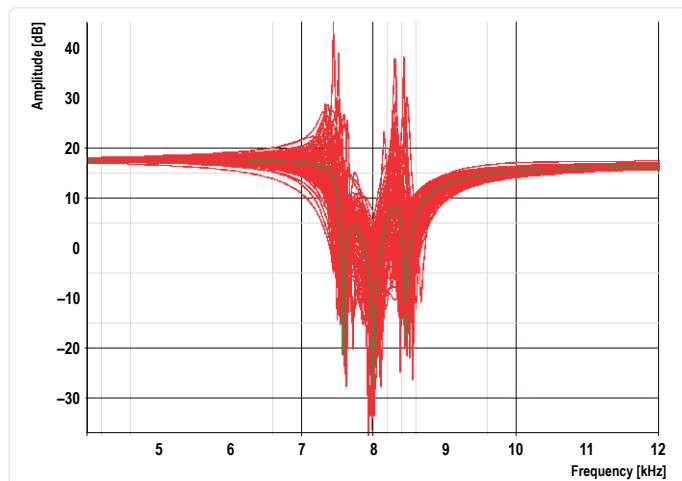


Bild 9. Simulierter Frequenzgang einer SK-Bandsperre 6. Ordnung. Legende: grün = idealer Verlauf; rot = Monte-Carlo-Simulation der Bauteiltoleranzen.

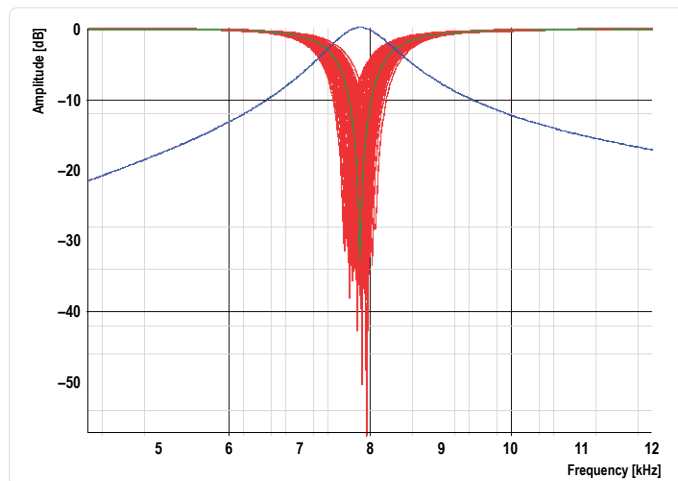


Bild 10. Simulierter Frequenzgang einer MFB-Bandsperre 2. Ordnung. Legende: grün = idealer Verlauf; rot = Monte-Carlo-Simulation der Bauteiltoleranzen; blau = Kurve des Bandpasses (siehe Text).

Bandsperrern

Für eine Bandsperre 2. Ordnung gibt es außer SK und MFB (Bild 7) noch die sogenannte Bainter-Struktur. Die SK-Implementierung kommt ebenfalls mit einem einzigen Opamp aus und beherbergt eine einfache Doppel-T-Anordnung bestehend aus drei Widerständen und drei Kondensatoren. Leider reagiert dieses Konzept empfindlich auf Bauteiltoleranzen. Die verwendeten Widerstände und ganz besonders die Kondensatoren sollten daher aus einer Charge stammen. Selbst durch Alterungserscheinungen kann sich der Frequenzgang noch dramatisch verändern.

Der Frequenzgang einer SK-Bandsperre für 8 kHz ist in Bild 8 zu sehen. Der ideale, hypothetische Frequenzgang ist grün eingezeichnet. Besonders interessant sind die zufälligen Abweichungen bei Bauteiltoleranzen von 1% für Widerstände und 5% für Kondensatoren. Wie der Frequenzgang von Bild 9 eindrucksvoll zeigt, sind SK-Bandpässe höherer Ordnung dafür noch empfindlicher. Dazu kommt, dass der Verlauf im Sperrbereich mehrere Täler hat. Die Verstärkung ist nicht frei wählbar und liegt pro Stufe etwa bei Faktor 2. Insgesamt kann man sagen, dass man für Bandsperrern besser einen anderen Typ als die Sallen-Key-Implementierung wählt. Bei der MFB-Bandsperre handelt es sich eigentlich um einen Bandpass, dessen Ausgangssignal vom Eingangssignal subtrahiert wird, und damit quasi indirekt eine Bandsperre ergibt. Diese Variante ist weniger empfindlich gegen Streuungen der Bauteilwerte, wie der Frequenzgang einer so aufgebauten MFB-Bandsperre 2. Ordnung in Bild 10 beweist. Hier ist zusätzlich der idealisierte Verlauf des Bandpassteils in blau dargestellt.

Ein Bainter-Filter benötigt gleich drei Opamps. Weitere Infos zu dieser Variante finden sich z.B. unter [2][3]. Bei der Schaltung von Bild 11 sind die konkreten Werte für eine Bandsperre bei knapp unter 8 kHz angegeben. Der zugehörige Frequenzgang ist in Bild 12 zu bewundern. Man sieht deutlich den „schöneren“ Verlauf der Kurven und insbesondere die Unabhängigkeit der Unterdrückung von der Streuung der Bauteilwerte.

Interessanterweise berechnet die ältere Desktop-Version meines Filterprogramms nur die SK- und MFB-Implementierung. Mittlerweile präferiert Texas Instruments die neuere Web-Version [3], die lediglich die Bainter-Variante unterstützt.

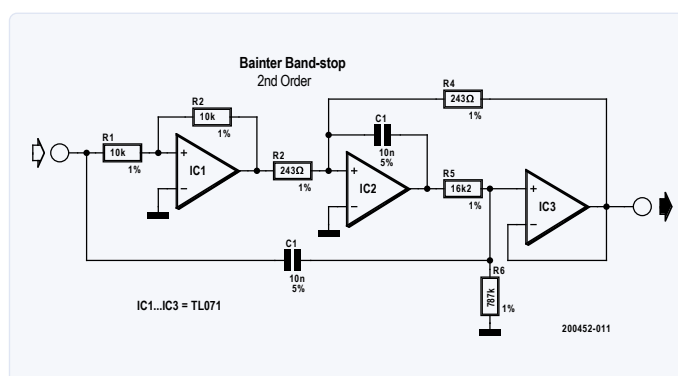


Bild 11. Prinzipschaltung einer Bainter-Bandsperre 2. Ordnung. Es werden drei Opamps benötigt.

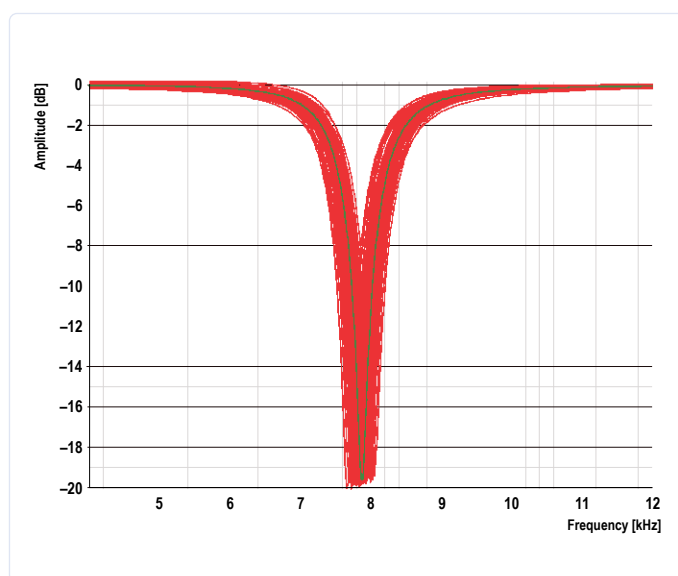


Bild 12. Simulierter Frequenzgang einer Bainter-Bandsperre 2. Ordnung. Legende: grün = idealer Verlauf; rot = Monte-Carlo-Simulation der Bauteiltoleranzen.

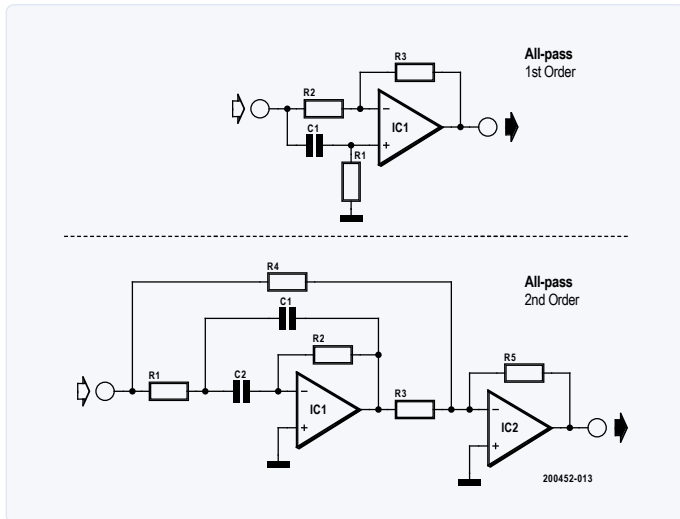


Bild 13. Prinzipschaltungen von Allpässen 1. und 2. Ordnung.

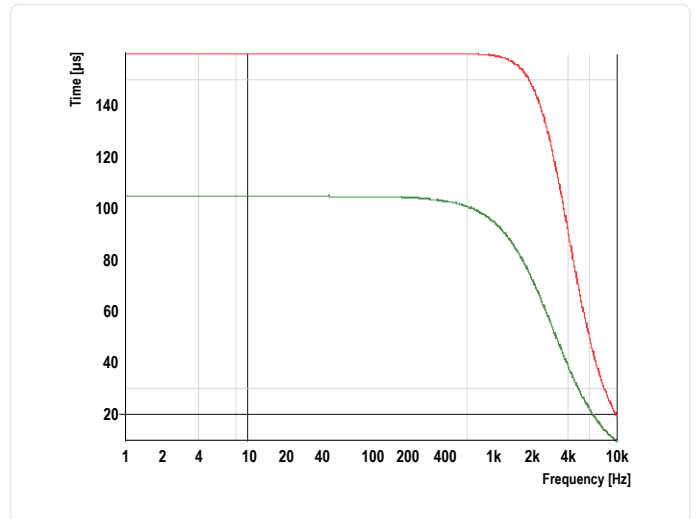


Bild 14. Frequenzgänge der Gruppenlaufzeit der Allpässe 1. (grün) und 2. Ordnung (rot) nach Bild 13.

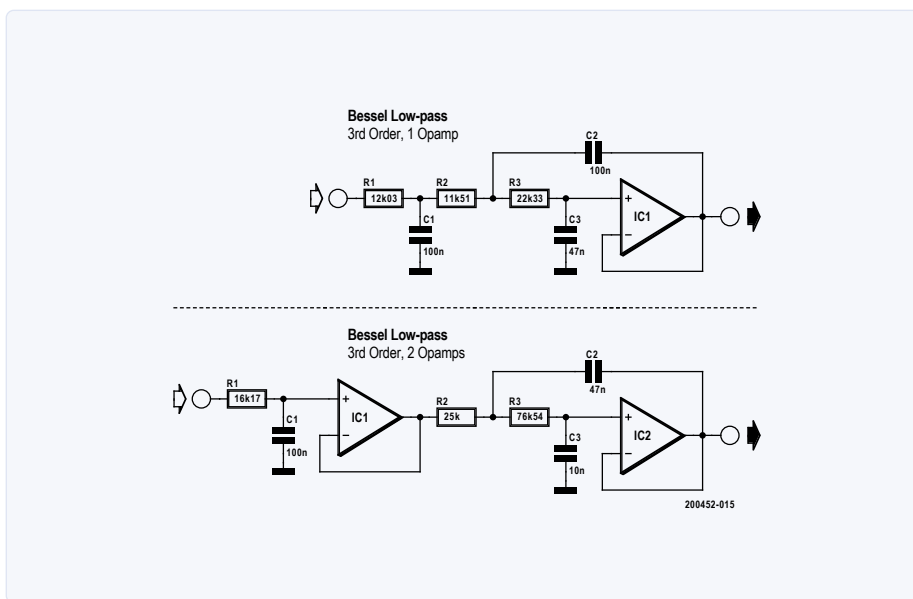


Bild 15. Dimensionierte Schaltungen von Bessel-Tiefpässen mit einem und zwei Opamps.

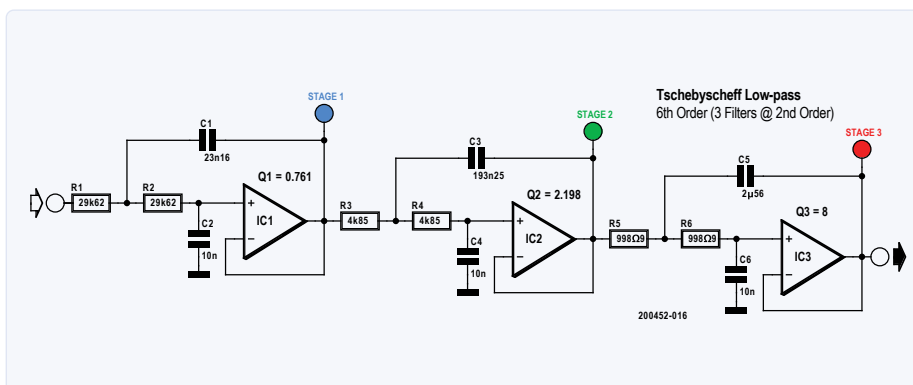


Bild 16. Dimensionierte Schaltung eines Tschebyscheff-Tiefpasses 6. Ordnung.

Allpässe

Die bisher besprochenen Filter zielen auf eine frequenzabhängige Änderung der Amplitude. Die Frequenzabhängigkeit von Phasenverlauf bzw. Gruppenlaufzeit sind eher sekundär oder gar unerwünscht (außer beim Bessel-Filter).

Bei Allpässen ist die Amplitude des Signals nicht abhängig von der Frequenz, sondern konstant. Die Definition einer auf die Amplitude bezogenen Grenzfrequenz würde keinen Sinn machen. Stattdessen wird zur Charakterisierung der Punkt herangezogen, an dem die Gruppenlaufzeit um -3 dB (auf 70,7 %) absinkt.

Bild 13 zeigt die Prinzipschaltungen von Allpässen 1. und 2. Ordnung. In **Bild 14** sind die zugehörigen Gruppenlaufzeiten über der Frequenz zu sehen.

Höhere Ordnungen

Filter höherer Ordnung kann man in der Regel auch schlicht durch Hintereinanderschalten von zwei oder mehr Filtern erster und zweiter Ordnung realisieren. Ein Tschebyscheff-Filter 4. Ordnung wird allerdings nicht durch zwei identische serielle Tschebyscheff Filter zweiter Ordnung konstruiert. Die Dimensionierung der Bauteile ist nämlich unterschiedlich und logischerweise auch das Filterverhalten.

Pro Filterstufe erster oder zweiter Ordnung braucht es in der Regel einen Opamp – zumindest die gängigen Tools zur Filterberechnung schlagen dies vor. Dennoch

lassen sich auch Filter dritter und sogar vierter Ordnung mit nur einem einzigen Opamp (mit Einschränkungen) realisieren. Zumindest die neuere Web-Version des TI-Filter-Tools bietet diese Möglichkeiten aber nicht mehr an, denn Opamps sind preiswert und der höhere Aufwand wird mit weniger sensiblen Filtern belohnt. **Bild 15** zeigt einen Tiefpass 3. Ordnung mit einem und zwei Opamps zum Vergleich. Die Werte der Bauteile beider Filter hat das Filter-Tool für eine Grenzfrequenz von 133 Hz berechnet.

Es spricht nichts dagegen, ein Filter höherer Ordnung nicht homogen zu gestalten, sondern aus unterschiedlichen Filtern wie etwa einem SK-Tiefpass und einem zweiten MFB-Tiefpass zusammenzusetzen. Auch kann man die Teilfilter je nach Bedarf invertierend oder nichtinvertierend realisieren, wie das ja bei SK- und MFB-Filtern eh der Fall ist. Der Fantasie sind keine Grenzen gesetzt. Es spricht bei solchen eher exotischen Anordnungen dann aber viel dafür, die Schaltung vor dem Aufbau zu simulieren [4]. Dann weiß man, ob die gewünschten Filtereigenschaften annähernd dabei herauskommen.

Bei Filtern höherer Ordnung ist noch ein weiterer Aspekt relevant: die Filtergüte Q . Man kann sie sich vom Filter-Tool für jede Stufe getrennt ausgeben lassen. Im Prinzip kann man die Reihenfolge der Filterstufen frei wählen, da die Filterausgänge ja durchweg niederohmig sind. Kommen die Stufen mit hohem Q zuerst, kann es eher zu Übersteuerungen kommen als bei umgekehrter Reihenfolge. Das von mir benutzte Tool [5] sortiert die einzelnen Stufen praktischerweise von selbst nach aufsteigendem Q .

Bild 16 zeigt ein Beispiel eines mehrstufigen Filters in Form eines Tschebyscheff-Tiefpasses 6. Ordnung für eine Grenzfrequenz von 1 kHz. **Bild 17** präsentiert den Frequenzgang nach jeder der drei Stufen in verschiedenen Farben. Simuliert man die drei Stufen des Filters einzeln, ergeben sich die hypothetischen Einzel-Frequenzgänge von **Bild 18**. Vertauscht man in der Schaltung von Bild 16 die erste mit der dritten Stufe, ergeben sich die Frequenzgänge von **Bild 19**. Zu beachten ist, dass die Farben für jede Stufe mitwandern. Die jetzt erste Stufe bleibt rot und das Gesamtsignal am Ende blau. Die rote Kurve von Bild 17 und die blaue Kurve von Bild 19 sind daher nahezu identisch. Man kann in Bild 19 gut sehen, dass es bei hohen Pegeln rund um die Grenzfrequenz in der ersten (rot) und zweiten Stufe (grün) leicht zu Übersteuerungen kommen kann. Deshalb zieht man die Anordnung von Bild 16 der umgekehrten vor.

Verstärkung

Aktive Filter können neben der eigentlichen Filterfunktion auch Verstärkungen > 1 erzeugen. Die Verstärkung hat bei manchen Filtern einen Einfluss auf die Werte der frequenzbestimmenden Bauteile – gelegentlich bestimmt ein Bauteil nicht nur die Verstärkung, sondern beeinflusst auch die Grenzfrequenz. Man kann also z.B. bei den SK-Filtern von Bild 4 (mit einer Ordnung > 1) nicht einfach R_3 oder R_4 ändern, ohne auch die Filtereigenschaften zu beeinflussen. Mit steigender Verstärkung nimmt üblicherweise die „Spreizung“ der Werte (Unterschied zwischen kleinstem und größtem Wert) von Widerständen und Kondensatoren zu. Diesen Sachverhalt sollte man beim Design berücksichtigen, da diese Spreizung wiederum Einfluss auf die erforderliche Bauteiltoleranz hat. Glücklicherweise verteilen die meisten Filter-Tools die Gesamtverstärkung eines Filters höherer Ordnung auf die verfügbaren Opamps. Durch eine Simulation mit Toleranzwerten der

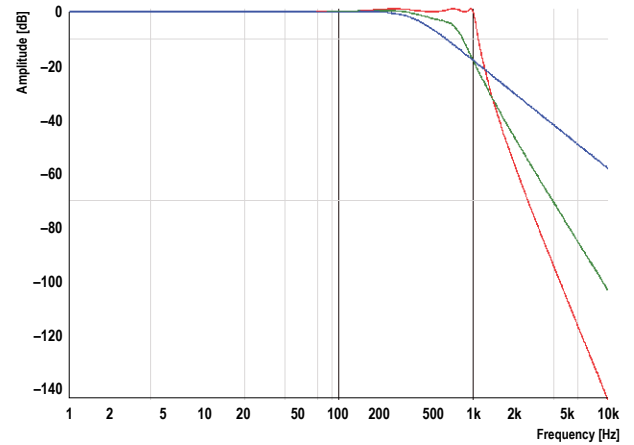


Bild 17. Frequenzgänge der drei Stufen des Tschebyscheff-Tiefpasses 6. Ordnung von Bild 16. Legende: blau = Stufe 1; grün = Stufe 2; rot = Stufe 3 (Ausgang).

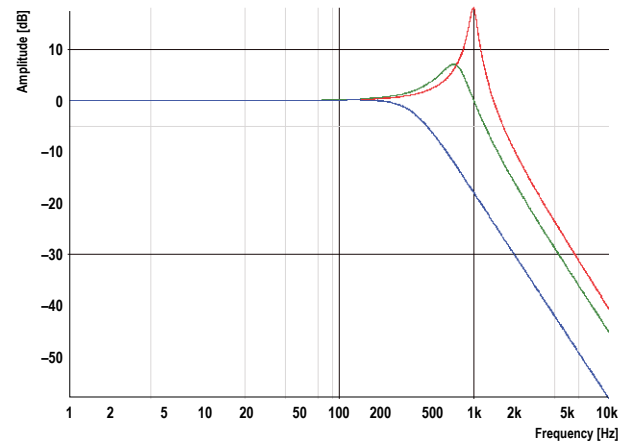


Bild 18. Simulierte Frequenzgänge der drei einzelnen Teilfilter des Tschebyscheff-Tiefpasses 6. Ordnung von Bild 16. Legende: blau = Stufe 1; grün = Stufe 2; rot = Stufe 3.

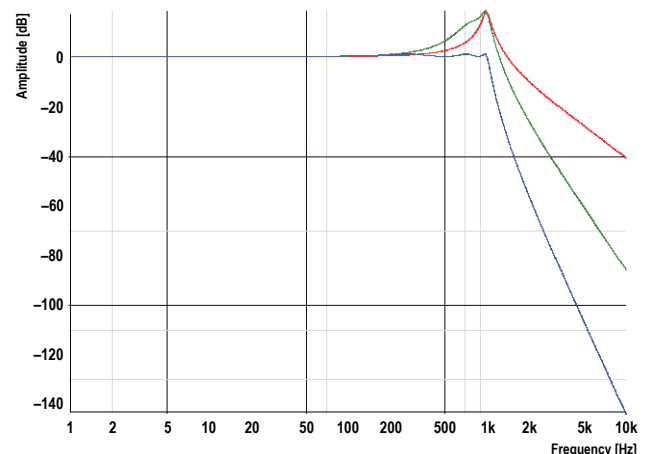


Bild 19. Frequenzgänge der drei Stufen des Tschebyscheff-Tiefpasses 6. Ordnung mit gegenüber Bild 16 vertauschter 1. und 3. Stufe. Legende: rot = Stufe 1; grün = Stufe 2; blau = Stufe 3 (Ausgang).

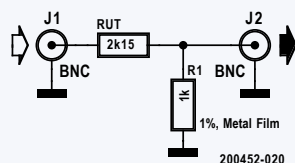


Bild 20. Testschaltung zum Messen der Verzerrungen eines Widerstands (RUT) mit einem Audio-Spektrum-Analyzer.

Bauteile nach der Zufallsmethode (Stichwort: Monte Carlo) kann man sich anschauen, wie sensibel ein Filter auf die Streuung der Werte reagiert.

Bauteildimensionierung

Das gleiche Frequenzverhalten kann man mit verschiedenen Werten der Bauteile erreichen. Nimmt man größere Widerstandswerte so werden die Kondensatorwerte kleiner und umgekehrt. Größere Widerstandswerte erhöhen das Rauschen, aber zu kleine Werte könnten Verzerrungen erzeugen, da die Opamps größere Ströme liefern müssen. Für Audiofilter sind Werte von 2...4 k Ω ein guter Kompromiss.

Ein gutes Filter-Tool lässt es zu, die Bauteilwerte zu variieren. Besonders wichtig ist, dass man nicht nur mit idealen Bauteilwerten vorlieb nehmen muss (siehe die Bilder 15 und 16), wie sie besonders bei Kondensatoren kaum erhältlich sind. Schön ist, wenn man die verfügbaren E-Reihen für Widerstände und Kondensatoren getrennt spezifizieren kann (wie bei Bild 11). Manchmal hilft das aber alles nichts, und man kann dann versuchen, einem „krummen“ Wert durch Parallel- oder Serienschaltung von Standardwerten möglichst nahe zu kommen. Das TI-Tool zeigt bei den gewählten Bauteilwerten auch die Abweichung der realen Filterkurve vom Ideal. Man kann durch Änderungen versuchen, die Abweichungen zu minimieren.

Eine Monte-Carlo-Simulation berücksichtigt die gewählten Bauteiltoleranzen und zeigt mit Kurvenscharen plastisch, wie das Filterverhalten durch zufällige Werte streuen kann. Sind hier große Abweichungen im Verhalten feststellbar, empfiehlt sich das Ausweichen auf einen weniger sensiblen Filter.

Bauteilauswahl

Hat man ein Filter berechnet, muss man sich noch überlegen, welche Art von Bauteilen man einsetzen will, da dies Auswirkungen auf die Filter-Qualität und die Kosten hat. Beim angeführten Beispiel

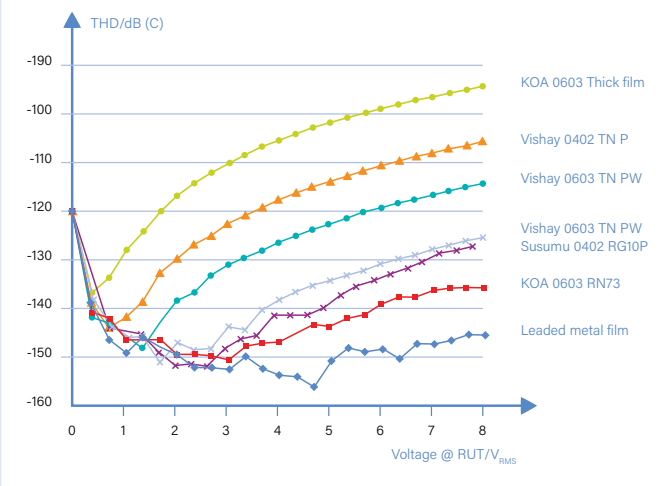


Bild 21. Verzerrungen unterschiedlicher Widerstandsarten bei 1 kHz in Abhängigkeit von der Signalamplitude.

eines Fledermausdetektors braucht es vermutlich keine besonders verzerrungsarmen Filter. Will man hingegen per Filter ein möglichst reines Sinus-Signal für Audio-Messzwecke erzeugen, sollte man mehr Aufwand treiben.

Widerstände:

Bei der Auswahl der Bauteile gilt: Widerstand ist nicht gleich Widerstand. Eine gute Metallfilm-Ausführung bringt messbar weniger Verzerrungen und Rauschen als ein 0815-Schicht-Exemplar. **Bild 20** zeigt eine Testschaltung, mit der die Auswirkungen eines bestimmten Widerstands (RUT = Resistor Under Test) gemessen werden, indem man ihn so an einen hochwertigen Audio-Spektrum-Analyzer anschließt. R1 sollte selbstverständlich ein Metallfilm-Widerstand mit bekannten, sehr guten Daten sein.

Bild 21 zeigt die Verzerrungen diverser Widerstandsarten in Abhängigkeit von der Signalamplitude. Wie man sehen kann: SMD-Dickfilmwiderstände sind sowohl bezüglich Verzerrungen als auch Rauschen mit Sicherheit eine schlechte Wahl. Bei SMD-Dünnschichtwiderständen sind größere Bauformen zu bevorzugen. Das beste Ergebnis zeigen bedrahtete Metallfilmwiderstände. Raten Sie mal, welche Art Widerständen in dem von mir eingesetzten Audio-Precision-Analyzer verbaut sind.

Kondensatoren:

Ähnliche Überlegungen gelten für Kondensatoren. SMD-Kondensatoren mit ≤ 1 nF sind überwiegend in NPO-Material gefertigt

WEBLINKS

- [1] Design analoger Filter (Teil 1), Elektor 9-10/2020: www.elektormagazine.de/200318-02
- [2] Halbleiter-Schaltungs-Technik 13. Auflage, U. Tietze, Ch. Schenk
- [3] Bandstop Filters and the Bainter Topology, Bonnie C. Baker: www.ti.com/lit/an/slyt613/slyt613.pdf
- [4] Simetrix: www.simetrix.co.uk/
- [5] TI-Filter-Tool, Web-Version: <https://webench.ti.com/filter-design-tool/>

Sie haben Fragen oder Kommentare?

Gerne können Sie sich an den Autor des Artikels wenden unter der E-Mail-Adresse: alfred_rosenkraenzer@gmx.de.

und damit sehr gut. Bei größeren Werten sind andere Materialien wie X7R, X5R etc. typisch - leider sind diese für verzerrungsarme Filter ungeeignet. Kondensatoren im Bereich von 10 nF bis $\leq 1 \mu\text{F}$ gibt es aus CoG-Material – leider sind hier größere Kapazitäten sehr teuer. Auch bei Kondensatoren sind bedrahtete Typen wie MKT oder Styroflex immer noch erste Wahl. Gerade bei SMD-Ausführungen und großen Werten sollte man die Spannungsfestigkeit im Auge behalten.

Opamps:

Auch die Auswahl des richtigen Opamps ist nicht ganz so einfach, wie man vielleicht annehmen mag. Zunächst sollte man auf die Versorgungsspannungen achten. In den heutigen, modernen Zeiten geht der Trend hin zu kleineren Spannungen. Statt den lange üblichen $\pm 15\text{V}$ sieht man zunehmend $\pm 5\text{V}$ oder noch weniger. Geringe Spannungen schränken den möglichen Signalpegel ein und nicht nur das: Sie verschlechtern auch das Signal/Rausch-Verhältnis. Bei hohen Pegeln steigt zudem die Gefahr von Verzerrungen oder gar Clipping.

Nicht umsonst ist in professionellem Audio-Equipment immer noch die Versorgung mit $\pm 15\text{V}$ obligatorisch. Generell sollte man selbstverständlich rausch- und verzerrungsarme ICs wählen. Ein weiterer Aspekt ist das vom Filterdesign geforderte Verstärkungs-Bandbreiten-Produkt eines Opamps. Nimmt man einen zu langsamen Opamp, so erreicht man unter Umständen nicht das anvisierte Filterver-

halten. Man sollte daher das berechnete Filter-Design simulieren, bevor man einen Prototypen baut – und natürlich nicht nur simulieren, sondern einen Prototypen bauen und vermessen, bevor man in die Serienproduktion einsteigt. ◀

200452-01



PASSENDE PRODUKTE

- **OWON SDS1102 2-Kanal-Digital-Oszilloskop (100 MHz)**
www.elektor.de/owon-sds1102-2-ch-digital-oscilloscope-100-mhz
- **Siglent SAG1021 Function/Arbitrary Waveform Generator (25 MHz)**
www.elektor.de/siglent-sag1021-function-arbitrary-waveform-generator-25-mhz
- **OWON XSA1015-TG Spectrum Analyser (9 kHz – 1.5 GHz)**
www.elektor.de/owon-xsa1015-tg-spectrum-analyser-9-khz-1-5-ghz

Anzeige

Sie haben eine Idee für ein Elektronikprojekt? Oder schon eine fertige Schaltung?

Posten Sie Ihr Projekt unter www.elektor-labs.de
(oder senden Sie es an redaktion@elektor.de).
Auch kleine Projekte sind willkommen!

Elektor Labs: Von Lesern für Leser!



Erstellen Sie **jetzt** ein neues Projekt auf
www.elektor-labs.de

design > share > sell



Zutritt für Unbefugte verboten!

Ein Blick ins Allerheiligste aller Elektroniker

Rainer Schuster

Sie werden es vielleicht nicht glauben, aber auch Elektronik-Enthusiasten haben Familie plus all die Verpflichtungen, die das Familienleben mit sich bringt...

Das gilt auch für Rainer Schuster, dessen Labor wir in dieser Artikelserie schon einmal besucht hatten (siehe Kasten „**Labor gesucht!**“). Letztes Jahr zu Weihnachten schenkte er seinen Enkeln eine Modellrennbahn, aber wie es sich für Kinder gehört, beschwerten sie sich bald, dass sie keinen Rundenzähler enthielt. Natürlich rennt ein Elektroniker, der etwas auf sich hält, nicht direkt in einen Laden, um ein fertiges Produkt zu kaufen - nur ein Rundenzähler Marke Eigenbau kommt in Frage. Und dann natürlich nicht nur mit den notwendigen, sondern auch mit Luxusfunktionen:

- Anzeige von Runde und Rundenzeit auf einem grafischen Display
- „Flaggen“-LEDs (rot/gelb/grün) für Start und Abbruch eines Rennens
- akustisches Signal (Summer)
- Anzeige des Rennverlaufs auf einem Smartphone (Bluetooth).

Der Rundenzähler ist für eine Carrera-Go-Modellrennbahn konzipiert, kann aber leicht an andere Marken angepasst werden. Die



Bild 1. Der Rundenzähler passt genau über eine zweispurige Rennstrecke.

Anzahl der Runden wird (jeweils für zwei Rennwagen) mit Infrarot-Reflexlichtschranken ermittelt. Für jedes Auto wird auch die aktuelle Rundenzeit berechnet. Die Ergebnisse werden auf einem 128 x 64 Pixel großen Grafikdisplay angezeigt. Die gleichen Daten können auch auf einem (Android-)Smartphone angezeigt werden. Das Gerät wird über ein Netzteil mit Strom versorgt (8...12 V_{DC}).

Bild 1 zeigt den Rundenzähler in seiner voller Pracht. Beim Gehäuse wurde eher auf „Kloppfestigkeit“ als auf ein tolles Design ausgelegt. Es passt genau genau über eine zweispurige Rennstrecke; die

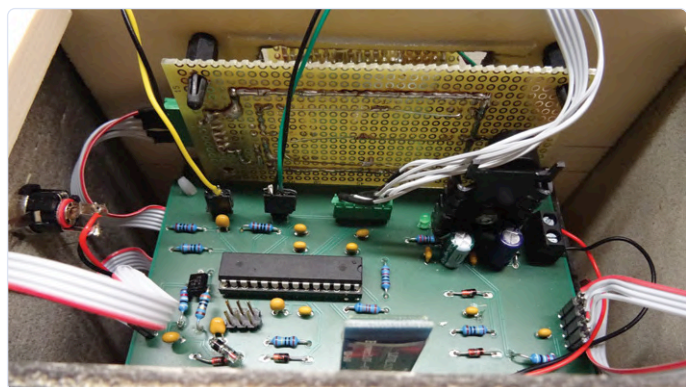


Bild 2. Die Elektronik ist auf zwei „echten“ Platinen und einem Stück Lochraster aufgebaut.



Bild 3. Das Display und die beiden Drucktasten sind im Gehäusedeckel montiert.

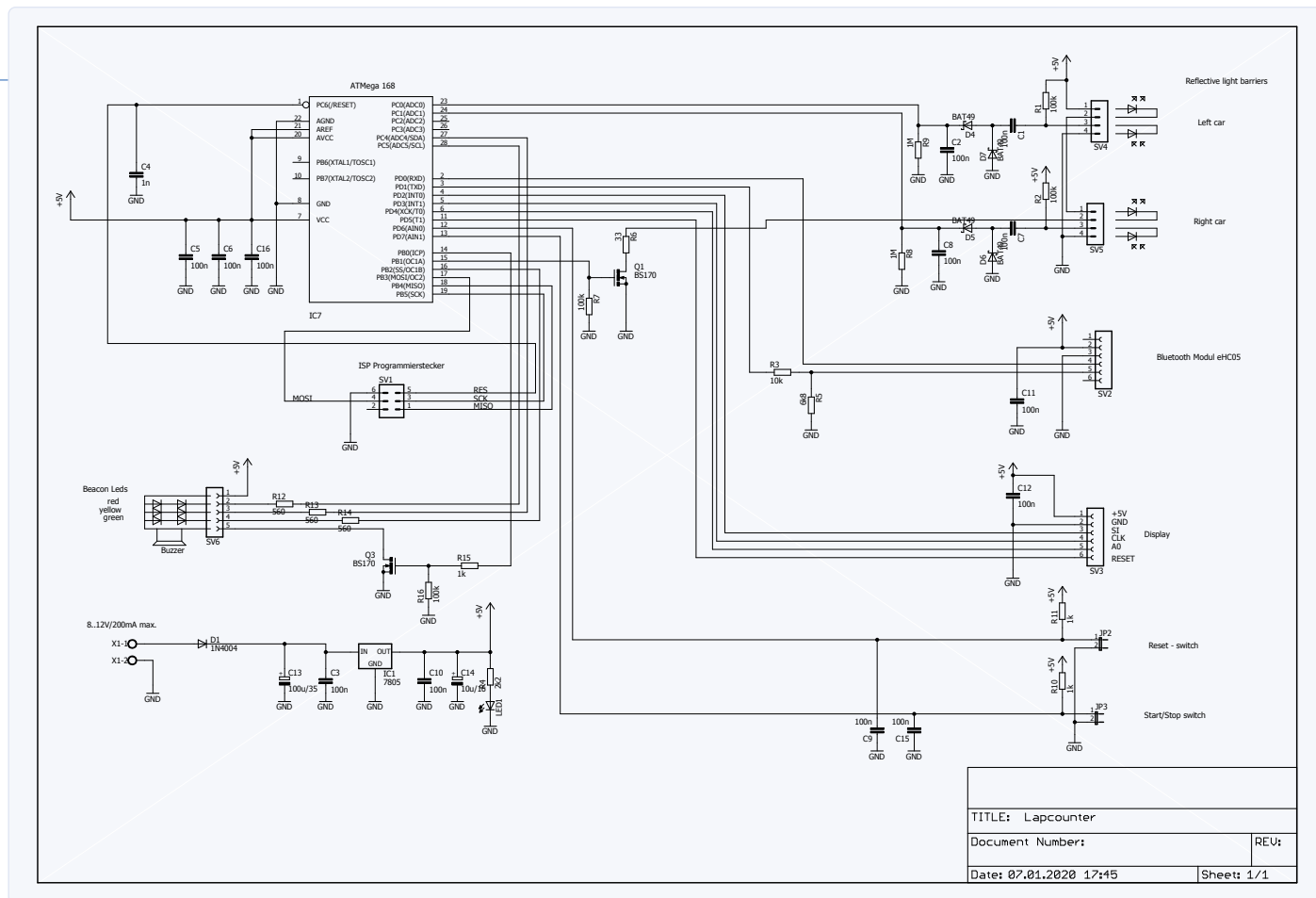


Bild 4. Die Elektronik ist bemerkenswert einfach.

LABOR UND PROJEKTE GESUCHT!

Sind Sie der glückliche Besitzer eines großen/kleinen/organischen/unordentlichen/interessanten Hobbylabors? Und entwickeln Sie dort große/kleine/schwere/notwendige/bekannte/unbekannte/weltbewegende Projekte? Schicken Sie uns (redactie@elektor.com) eine kurze Beschreibung und ein paar Bilder! Mit etwas Glück wird Ihr Labor oder Ihr Projekt in einer der nächsten Elektorausgaben vorgestellt.

IR-Lichtschranken befinden sich an der Innenseite des Tunnels. Die Hardware ist auf zwei Platinen verteilt: eine Platine für den Controller (in diesem Fall ein ATmega168) und eine Platine für das Display, die zusammen mit der Reset- und der Start/Stop-Taste im Deckel montiert ist. **Bild 2** gewährt einen Blick ins Innere des Startturms, die hier nicht sichtbare Anzeige ist in **Bild 3** dargestellt. Die Flaggen-LEDs werden auf ein Stück Experimentierplatine gelötet.

Um den Einfluss des Umgebungslichts zu minimieren, werden die IR-LEDs der hier verwendeten IR-Lichtschranken des Typs ITR9904 durch Impulse á 2 ms vom PWM-Ausgang des Mikrocontrollers gesteuert.

Das Schalt**bild 4** zeigt, dass es nicht viele Bauteile Elektronik braucht, um einen gut funktionierenden Rundenzähler zu konstruieren. Der größte Teil der „Intelligenz“ steckt in der Firmware des Mikrocontrollers. Der Autor zeigt unter [1] ein Video seines Rundenzählers in Aktion.

Sind Sie an diesem Projekt interessiert? Bitte lassen Sie es uns wissen (senden Sie uns eine E-Mail an redactie@elektor.com); wenn genügend Interesse besteht, werden wir nächstes Jahr darauf zurückkommen. Bis dahin können Sie Ihre Fragen (über die Redaktion) an den Autor stellen. ◀

200045-03

Ein Beitrag von

Idee, Entwicklung und Fotos:

Rainer Schuster

Text: **Eric Bogers**

Übersetzung:

Rolf Gerstendorf

Layout: **Giel Dols**

WEBLINK

[1] Video des Rundenzählers: <https://bit.ly/2ZxPwh0>



PASSENDE PRODUKTE

> Joy-Car Roboter (inkl. BBC micro:bit)
www.elektor.de/19408

Von Entwicklern für Entwickler

Tipps & Tricks, Best Practices und andere nützliche Infos

Von **Clemens Valens** (Elektor-Labor)

VON DER IDEE ZUM PRODUKT - TEIL 7 (ENDE)

In dieser Artikelreihe haben wir den Prozess der Markteinführung eines neuen Produkts diskutiert. Es begann mit der Vorbereitung eines Business-Plans und dem anschließenden Entwurf des Produkts. Wir gingen den ganzen Weg bis zur Produktion und Erprobung. Jetzt sollte es also endlich verkaufsfertig sein. Oder nicht?

instructions for use



BENUTZER- UND SERVICEHANDBÜCHER

Kein Produkt ohne Handbuch! Natürlich ist für Sie alles klar und logisch, aber der Benutzer denkt vielleicht nicht so wie Sie, und deshalb müssen Sie ihm etwas erklären. Dazu muss ein Benutzerhandbuch geschrieben werden, was keine triviale Aufgabe ist. Ein gutes Handbuch ist prägnant, klar und vollständig. Sind Sie dieser Aufgabe gewachsen oder wäre es eine gute Idee, dafür einen technischen Redakteur zu engagieren? Sobald Sie ein Handbuch, eine Bedienungsanleitung, haben, stellen Sie sicher, dass die Leute es leicht finden können. Vielleicht möchten Sie es dem Produkt beilegen, es zum Herunterladen zur Verfügung stellen oder beides.

Für bestimmte Produkte ist auch ein Servicehandbuch erforderlich. Ein solches Handbuch richtet sich an Personen, die Ihre Produkte im Außendienst warten. Es erklärt zum Beispiel, wie man Teile austauscht und wie die Dinge funktionieren sollten oder welche Art von Signalen an welchem Testpunkt verfügbar sein sollten. Auch die Erstellung eines solchen Handbuchs ist eine große Herausforderung, die Sie vielleicht auslagern möchten.

Dilemma: Stecken Sie Ihr Produkt in einen einfachen Karton oder bemühen Sie sich, den Benutzer zufrieden zu stellen?

PACKAGING

Wie Ihnen jeder Marketingmensch bestätigen kann, ist die Verpackung eines Produktes essentiell. Die Verpackung fasst das Produkt, eine Gebrauchsanweisung und oft auch einige Zubehörteile zusammen, und sie hilft zu erkennen, was sich in ihr befindet. Eine gute Verpackung ermöglicht eine sichere

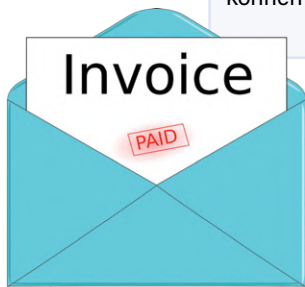
Lagerung und einen sicheren Transport des Produkts. Eine gute Verpackung macht Lust, das Produkt zu kaufen. Wie ein Produkt erfordert auch die Verpackung einen gewissen Gestaltungsaufwand. Die Abmessungen, die Form und das Material müssen definiert und die aufgedruckten Grafiken und Texte festgelegt werden. Wird Füllmaterial benötigt? Soll die Verpackung kundenspezifisch sein oder genügt eine einfache braune Pappschachtel? Wenn es eine Sonderanfertigung ist, wer wird sie dann herstellen? Was wird es kosten? Wie wird sich das auf den Verkaufspreis des Produkts auswirken? Und, nebenbei bemerkt, wer wird die Produkte in diese Kartons stecken?

BEISPIELE FÜR DIE VERWENDUNG

Wenn Sie Audioverstärker oder Autos herstellen, müssen Sie Ihren Kunden nicht sagen, wofür sie diese verwenden sollen. Allerdings sind nicht alle Produkte so gestaltet; einige erfordern Anwendungsbeispiele. Die Tatsache, dass der Benutzer Ihr Produkt gekauft hat, bedeutet nicht unbedingt, dass er oder sie versteht, wie er oder sie all die tollen Funktionen nutzen kann, die Sie in das Produkt eingebaut haben. Eine Zeile im Benutzerhandbuch, in der es heißt: *Dingsbums-Knopf: die Dingsbums-Funktion anwenden*, ist nicht sehr hilfreich, wenn Sie nicht erklären, was *Dingsbums* ist und warum oder worauf Sie es anwenden möchten oder sollten. Geben Sie nicht nur eine Anwendung, sondern mehrere Beispiele an, die über die einfachsten Anwendungsfälle hinausgehen.

VERSAND, VERTRIEB UND MARKETING

Eine andere Sache, über die man sich schon früh Gedanken machen sollte, ist, wie man sein Produkt vertreiben will. Werden Sie es selbst und direkt verkaufen oder überlassen Sie das lieber auf den Verkauf spezialisierten Personen wie Händlern und Ladengeschäften? Wenn Sie sich selbst darum kümmern, müssen Sie keine Provisionen zahlen, aber Sie müssen vielleicht jemanden einstellen, der jeden Tag zur Post geht. Lohnt es sich vielleicht, einen Deal mit einem Kurierdienst zu machen? Sind die Kosten abhängig vom Bestimmungsort oder soll ein einheitlicher Wert für alle berechnet werden? Wenn Sie sich für einen Einzelhändler entscheiden, woher wissen Sie, dass er sich genug Mühe gibt, Ihr Produkt zu verkaufen? Wer wird sich um die Werbung kümmern? Um wie viel können Sie den Verkaufspreis erhöhen, um dafür zu bezahlen? Oder werden Sie in den sauren Apfel beißen und stattdessen Ihren Gewinn schmälern?



RECHNUNGSSTELLUNG UND BUCHHALTUNG

Es klingt überraschend, aber viele junge Firmen sterben einen frühen Tod, weil sie sich nicht die Zeit nehmen, den Überblick über finanzielle Angelegenheiten zu behalten, zum Beispiel die Nachverfolgung unbezahlter Rechnungen oder die Anfrage von Angeboten (oder sogar die Buchhaltung im Allgemeinen). Sei es aus dem Wunsch heraus, die Kunden bestmöglich zu behandeln oder einfach nur, weil die Nachfrage überwältigend ist. Was auch immer der Grund dafür ist, wenn Sie die finanziellen Dinge zu lange ignorieren, wird Ihr Unternehmen nur ein kurzes Leben führen. Die unbezahlten Rechnungen mögen von Ihren Lieferanten stammen (wen kümmert's, die haben ja Geld), oder aber es sind Sie, der seine Lieferanten (noch) nicht bezahlt hat oder Sie sind ein Lieferant, der mit langsam oder gar nicht zahlenden Kunden zu tun hat. Schuldner zu jagen ist eine ärgerliche Tätigkeit, aber wenn Sie keine Mahnungen verschicken, können Sie keine Schuldner jagen, und das ist noch schlimmer. Die Erstellung von Kostenvoranschlägen kostet zwar Zeit, aber gute Angebote ziehen neue Kunden an. Jeder weiß, wie wichtig es ist, den Überblick über Ausgaben und Einnahmen zu behalten, aber es braucht ein wenig Zeit, um sich den Überblick zu verschaffen. Viele Menschen tendieren dazu, dies hinauszuzögern, weil es ja noch so viele andere wichtige Dinge zu tun gibt, um das Geschäft zum Laufen zu bringen. In gewisser Weise haben sie Recht: Wenn man zu lange zögert, verschwinden die Probleme von selbst ... zusammen mit all Ihren Träumen.

SUPPORT UND AFTERSALES

Kundendienst kann schnell zeitaufwändig werden, doch die Qualität Ihres Kundendienstes wirkt sich direkt auf den Ruf Ihres Unternehmens aus und spiegelt sich im Verkauf wider (es sei denn, Sie sind Apple). Also: Wer wird mit Ihren Kunden kommunizieren und wie? Heutzutage ist es eine Unsitte vieler Unternehmen, ihre Telefonnummer(n) und E-Mail-Adressen geheim zu halten. Die Idee ist, alle Service-Anfragen auf eine Webseite mit entweder einem Kontaktformular oder einer Liste häufig gestellter Fragen (FAQ) (vergessen Sie die Antworten nicht) oder einem Chat-Bot oder einem Forum oder was auch immer umzuleiten. Aber selbst wenn man das beste Selbsthilfe-Servicezentrum aller Zeiten einrichtet, wird es immer ein paar Kunden geben, die mit einer echten Person aus Fleisch und Blut sprechen wollen. Und in dem ganz speziellen Fall, dass es keine solche Service-Person gibt, wird immer noch eine reale Person benötigt, um die E-Mails zu lesen, die unter info@acme-products.com eingehen, die Kontaktformulare zu bearbeiten, die vom Chat-Roboter erstellten Berichte zu bearbeiten und die Post zu öffnen, die der Postbote heute Morgen in den Briefkasten gesteckt hat.



RÜCKSENDUNGEN, REPARATUREN UND GARANTIEEN

Auch wenn Sie Ihr Bestes getan haben, um sicherzustellen, dass all Ihre Produkte nach den höchsten Qualitätsstandards gebaut wurden und genau wie vorgesehen funktionieren, wird ein Teil von ihnen ausfallen. Hoffentlich wenige, aber sicher einige werden mit einer - wahrscheinlich vagen oder unklaren - Beschwerde des Benutzers zurückkommen. Wie handhaben Sie Produktrückläufer? Reparieren Sie sie? Oder ist es billiger, sie zu ersetzen? Und wer wird dies tun? Wenn Sie ein Produkt ersetzen, was machen Sie mit den defekten Produkten? Was ist gerechtfertigt und was nicht? Wie lang läuft überhaupt die Garantiezeit? In einigen Ländern gibt es Mindestgarantiezeiten für bestimmte Produkttypen, so dass Sie sich darüber informieren sollten.

ABFALLENTSORGUNG

In verschiedenen Ländern sind die Hersteller für die Entsorgung von Produkten verantwortlich, die von ihren Benutzern weggeworfen wurden. Das bedeutet, dass Benutzer Produkte zur Entsorgung oder zum Recycling zu Ihnen zurückschicken können (und werden). Je mehr Produkte Sie verkaufen, desto größer wird Ihr Abfallberg. Vielleicht können Teile des E-Mülls wiederaufbereitet werden? Vielleicht müssen einige kritische Teile mit besonderer Sorgfalt entsorgt werden? Aber unabhängig, wie Sie mit dem Abfall umgehen, es werden Kosten anfallen.



DENKEN SIE AN DIE ZUKUNFT!

Natürlich ist Ihr Produkt großartig. Es löst alle Probleme, für die es entwickelt wurde, aber irgendwie werden die Kunden nach einer unterschiedlich langen Zeit anfangen, nach anderen Funktionen zu fragen. Oder Sie selbst haben eine großartige Idee für eine Produktweiterentwicklung oder eine Erweiterung der Produktlinie. Leider sind Sie und all Ihre Mitarbeiter in Produktion, Test, Versand, Verpackung, Marketing, Buchhaltung, Verkauf und Kundendienst gebunden und haben daher keine Zeit mehr, sich mit Technik und Produktdesign zu beschäftigen. Wieder einmal wird durch Zögern dieses Problem verschwinden, aber wahrscheinlich auch Ihr Unternehmen.

BÜRO UND MITARBEITER

Wenn Sie allein arbeiten, macht es Ihnen wahrscheinlich nichts aus, in Ihrer Garage oder in der Küche zu arbeiten, aber wenn Sie anfangen, Leute einzustellen, werden Sie mehr Platz benötigen. Es gibt natürlich das Home-Office, beliebt gemacht durch Sie wissen schon was, aber es ist nicht geeignet, wenn die Kommunikation intensiv ist oder wenn Lötdämpfe zu viel Tageslicht absorbieren. Das Anmieten von Büroraum kann hier helfen. Dafür können Sie etwas Billiges am äußersten Ende einer Industriezone wählen, aber auch der Anschluss an einen Inkubator kann eine interessante Option sein. Der Standort ist in der Regel attraktiver und es werden nützliche Dienstleistungen angeboten. Inkubatoren können auch bei finanziellen Angelegenheiten helfen, aber lesen Sie das Kleingedruckte sorgfältig durch, bevor Sie etwas unterschreiben. Abgesehen davon, dass sie bezahlt werden müssen, müssen die Mitarbeiter auch „verwaltet“ werden, was Zeit kostet. Es kann verlockend sein, einen Freund einzustellen, aber ein Freund kann auch schwieriger zu behandeln sein. Wenn Sie also Ihre Freunde behalten wollen, ist es besser, eine Anzeige auf einer Social-Media-Plattform zu schalten.

Werden alle Ihre Mitarbeiter in Ihre Teeküche passen?

DAS ENDE

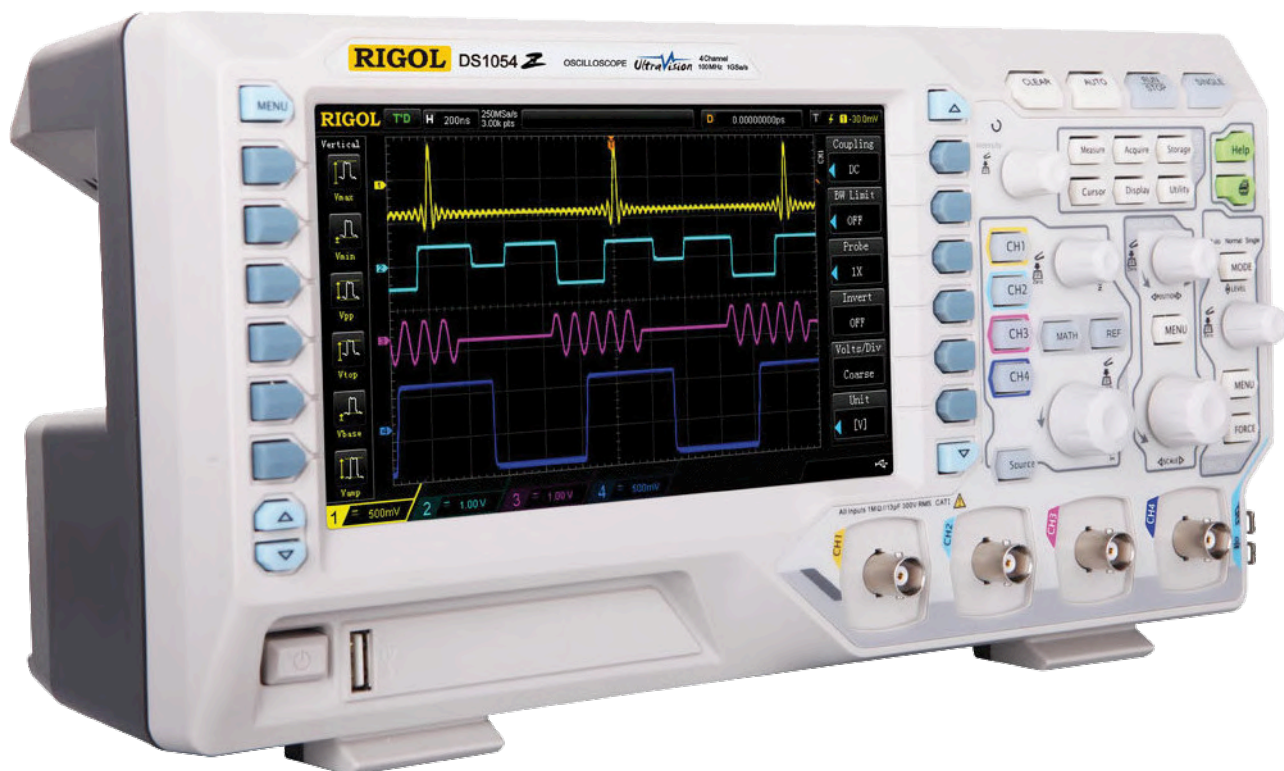
Das war's mit dieser Serie. Es gäbe noch viel mehr zu sagen und ich bin sicher, dass ich ein oder zwei Dinge vergessen habe. Aber Sie haben die allgemeine Idee verstanden, nicht wahr? Wenn Sie die Richtlinien aus der Artikelreihe befolgt und Ihren eigenen gesunden Menschenverstand eingesetzt haben, sind Sie jetzt vielleicht im Geschäft. Lassen Sie uns hoffen, dass Ihr Produkt erfolgreich sein wird. Natürlich muss nicht gleich am ersten Tag alles perfekt sein. Einige Dinge werden sich organisch entwickeln, aber es ist immer nützlich, auf sie vorbereitet zu sein. Ich wünsche Ihnen viel Glück! ◀

Sie haben Fragen oder Kommentare?

Haben Sie Fragen oder Anmerkungen zu dem Artikel? Schicken Sie eine E-Mail an den Autor unter clemens.valens@elektor.com oder kontaktieren Sie Elektor unter editor@elektor.com.

Vierkanal-Oszilloskop

Rigol DS1054Z



Von **Harry Baggen**

Wenn Sie auf der Suche nach einem zuverlässigen und dennoch erschwinglichen Vierkanal-Oszilloskop sind, werden Sie über kurz oder lang auf das Rigol-Modell DS1054Z stoßen. Für weniger als 400 € bietet dieses Gerät zusätzlich zu seinen vier Kanälen ein großes Display, viele Messfunktionen und viele andere Features. Genug Gründe, um es einmal in der Praxis zu testen.

Das Oszilloskop DS1054Z von Rigol ist bereits seit mehreren Jahren auf dem Markt, aber dennoch aktuell und eines der beliebtesten Vierkanal-Oszilloskope in seiner Preisklasse. Die wichtigsten Spezifikationen sind natürlich die Anzahl der Kanäle (4!), die Abtastrate von 1 GS/s bei einem Kanal und die Bandbreite von 50 MHz. Im Vergleich zur Konkurrenz ergibt das ein sehr gutes Preis/Leistungs-Verhältnis. Als Extra bekommt

man zurzeit vier Software-Optionen kostenlos: Eine Speichererweiterung auf 24 Mpts, eine erweiterte Trigger-Option sowie Protokoll-Decoder für RS232, I²C und SPI plus ein Aufzeichnungs-Modul. Das Gesamtpaket ist also für diesen Preis recht attraktiv.

Hardware

Das DS1054Z steckt in einem relativ kleinen Gehäuse von 31x16x12 cm und wiegt nur etwas mehr als 3 kg. Den größten Teil der

Vorderseite nimmt ein großes 7"-Display mit einer Auflösung von 800x480 Pixeln ein – der Standard in dieser Preisklasse. Auf der rechten Seite sind die Bedienelemente zusammengefasst. Wegen des begrenzten Platzes auf der Vorderseite gibt es nur einen Satz Tasten für die vertikale Einstellung der Kanäle, mit Hilfe von Drucktasten wählen Sie aus, welcher Kanal eingestellt wird. Weiterhin finden wir die Einstellungen für die Horizontalablenkung und den

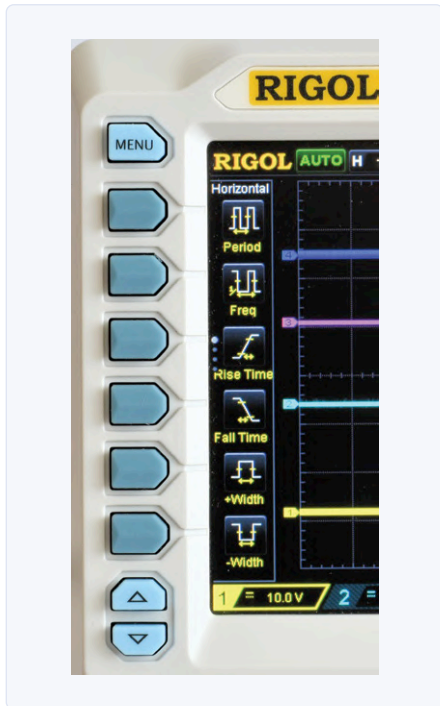


Bild 1. Die linken Funktionsmenüs.



Bild 3. Die rechten Funktionsmenüs.

Triggerteil sowie eine Reihe von Menütasten für Funktionen auf der rechten Seite des Displays: Measure, Acquire, Storage, Cursor, Display und Utility. Es gibt auch einen multifunktionalen Drehgeber zur Auswahl von Optionen in verschiedenen Menüs.

Links und rechts neben dem Display befindet sich eine große Anzahl von Funktionstasten. Die linken Tasten sind für das Mess-Menü mit einem Satz von Optionen für den horizontalen und den vertikalen Bereich. Die ausgewählten Messwerte

erscheinen unten auf dem Display. Die Schaltflächen auf der rechten Seite dienen den Funktionsmenüs.

Das robuste Gehäuse hat an der Vorderseite zwei Klappfüße. Auf der Rückseite befinden sich neben dem Netzanschluss ein USB- und LAN-Anschluss sowie ein Triggereingang für serielle Messungen. Ein Lüfter im DS1054Z sorgt für die notwendige Kühlung im Inneren des Geräts. Er ist hörbar, aber ich fand das Geräusch nicht störend.

Erste Schritte

Die Bedienung des Oszilloskops ist wegen der vielen Knöpfe rund um das Display anfangs gewöhnungsbedürftig. Das Design der Bedienelemente ähnelt dem anderer Geräte dieser Preisklasse. Dies gilt auch für die Auswahl der Kanäle über vier Taster. Unten im Display wird angezeigt, welcher Kanal gerade aktiv ist. Hier sollte man immer vor dem Drehen der Knöpfe für die vertikale Einstellung hinschauen!

Die Kombination aus verfügbarer Bandbreite und maximaler Abtastrate ist für dieses Oszilloskop gut gewählt. Falls Sie denken, dass eine Bandbreite von 50 MHz etwas niedrig ist: Das ist für fast alle Anwendungen ausreichend, solange man nicht wirklich mit HF-Schaltungen zu tun hat. Man beachte, dass die Abtastrate auf die Hälfte oder ein Viertel sinkt, wenn zwei oder drei/vier Kanäle gleichzeitig verwendet werden. Bei 50 MHz erzielt man mit 250 MS/s pro Kanal nur noch 5 Samples/Periode, was für eine gute Darstellung des Signals viel zu wenig ist. Diese Kombination aus Eingangsbandbreite und Abtastrate ist also wirklich harmonisch, denn bei einem Kanal bekommt man so ausreichende 20 Samples/Periode!

Das Display ist auch von der Seite gut lesbar. Mir gefiel eine gegenüber dem Standard etwas höher eingestellte Helligkeit besser. Auf dem Bildschirm sind sehr viele Informationen dargestellt, vor allem durch die Präsenz der Menüs, die zu den Tastern auf beiden Seiten des Displays gehören. Alles in allem wirkt das leicht überladen, da auch noch oben und unten eine Reihe von Infos angezeigt werden. Der verfügbare Platz für die Darstellung der Wellenformen wird daher eingeschränkt.

Das DS1054Z bietet nicht weniger als 15 verschiedene Triggerfunktionen, wie

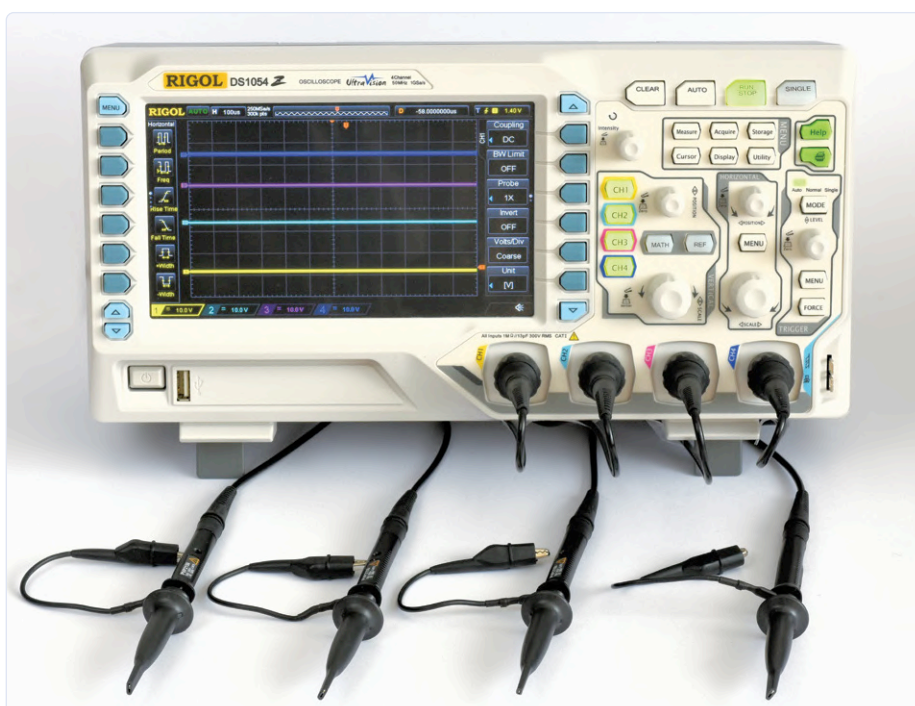


Bild 2. Vier Kanäle für weniger als 100 €/Kanal.

z. B. Pattern, Time-out, Window, Delay und weitere für die unterstützten seriellen Protokolle. Das Gerät verfügt über einen internen Samplespeicher von 24 Mpts. Das ist ziemlich groß, aber genau wie beim A/D-Wandler muss auch dieser Speicher auf die aktive Anzahl von Kanälen aufgeteilt werden. Aber selbst dann ist die Kapazität immer noch mehr als ausreichend.

Rigol hat sein Bestes getan, um möglichst viele Messfunktionen unterzubringen. Mit den Tasten auf der linken Seite des Displays kann man schnell aus 32 Parametern auswählen, deren Messwerte (max. fünf Werte auf einmal) dann unten im Bild erscheinen. Zusätzlich gibt es einen „normalen“ Frequenzzähler mit bis zu 20 anzeigbaren Parametern plus eine Funktion, welche die Abweichungen über eine Reihe von Perioden für bis zu fünf Parameter anzeigt. Man kann sowohl automatische als auch manuelle Cursor-Messungen vornehmen.

Unter dem Taster für Mathe gibt es eine große Anzahl mathematischer Operationen, die von einer einfachen Summierung über Logarithmus und Exponent bis hin zu Integration und Differenzierung reichen. Hier verbirgt sich auch die FFT-Funktion zur Anzeige des Frequenzspektrums eines Signals. Ein nettes Extra, aber ich hatte einige Schwierigkeiten, alle Parameter so einzustellen, dass ein brauchbares Diagramm entstand. Daran sollte Rigol wohl noch etwas schrauben.

Fazit

Das DS1054Z von Rigol ist ein vielseitiges Oszilloskop, das viel für sein Geld zu bieten hat. Dies gilt insbesondere zurzeit, da es aktuell mit verschiedenen Software-Extras ausgestattet ist. Obwohl es bereits seit mehreren Jahren auf dem Markt ist, kann es noch immer problemlos mit neueren Geräten in dieser Preisklasse mithalten. Für weniger als 400 € erhalten Sie ein zuverlässiges Gerät mit vier Kanälen und sehr vielen Funktionen, an dem Sie jahrelang Freude haben werden. 

200463-01E

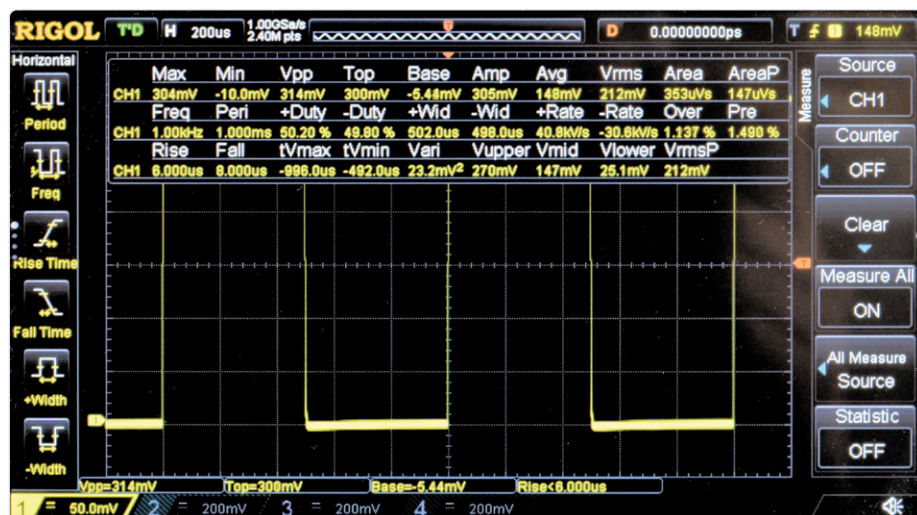


Bild 4. Alle Messwerte gleichzeitig anzeigen.

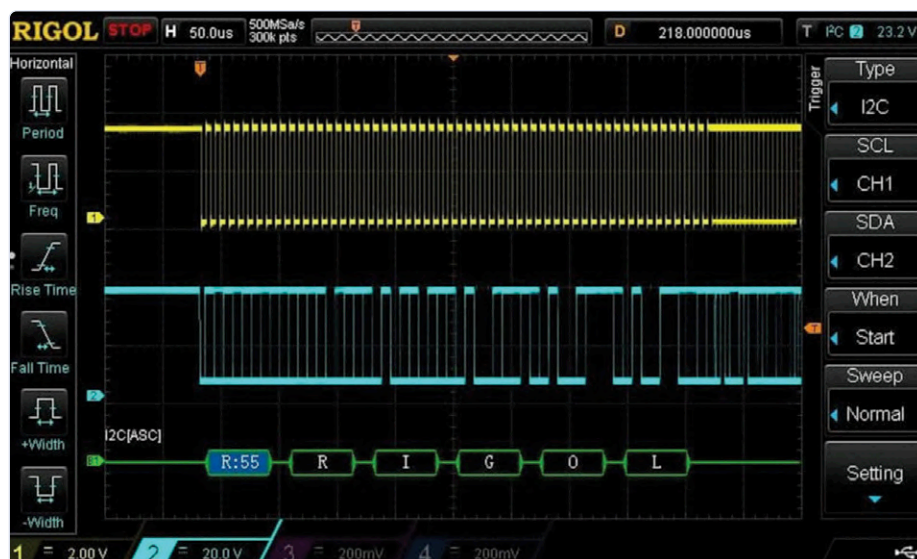


Bild 5. Das Gerät kann I²C dekodieren.



PASSENDE PRODUKTE

➤ **Rigol DS1054Z 4-Kanal-Oszilloskop**
www.elektor.de/rigol-ds1054z-4-ch-oscilloscope-50-mhz

Weihnachts-Wunderkerze

Puste sie aus!

Jedes Jahr geben wir unser Bestes, um in unserer Zeitschrift ein neues Weihnachtsgadget zu präsentieren, was sicherlich keine leichte Aufgabe ist. Es soll etwas Nettes, Lustiges, Erschwingliches, Einzigartiges sein, relativ einfach zu bauen und vorzugsweise eine perfekte Mischung all dieser Aspekte. Und wir sind uns der Tatsache bewusst, dass der Markt mit allen möglichen Weihnachtsgadgets zu Preisen überschwemmt wird, mit denen man nur schwer, eigentlich gar nicht konkurrieren kann. Dieses Mal haben wir beschlossen, zwei ältere Elektor-Projekte zu etwas Neuem zu kombinieren!

Ein Beitrag von

Idee: **Clemens Valens, Luc Lemmens**

Gestaltung und Text: **Luc Lemmens**

Illustrationen: **Jan Visser, Luc Lemmens**

Schaltbild: **Patrick Wielders**

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

Übersetzung: **Rolf Gerstendorf**

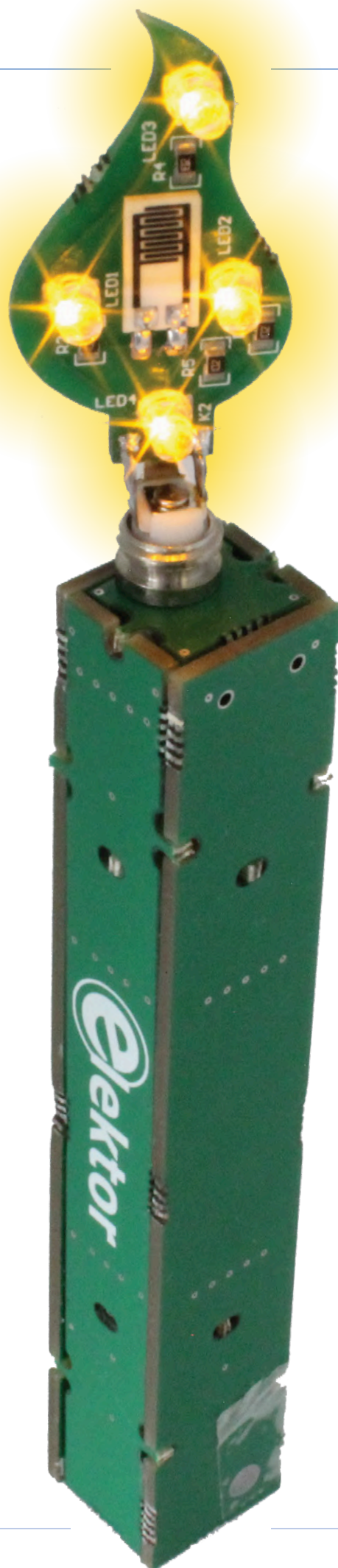
Layout: **Giel Dols**

Das erste Projekt, das wir für das diesjährige Weihnachtsgadget recycelt haben, stammt aus der Dezember-Ausgabe von Elektor 2011: die elektronische LED-Kerze [1]. Von diesem Artikel hat nur die Idee des „Ausblasens einer elektronischen LED-Kerze“ im diesjährigen Entwurf überlebt. Die Elektronik wurde komplett verändert. Im Jahr 2011 steuert ein Mikrocontroller des Typs PIC16F1827 gelbe Standard-SMD-LEDs für den flackernden Lichteffekt, während im neuen Entwurf 3-mm-LEDs mit eingebauter Elektronik diese Aufgabe übernehmen.

Die Möglichkeit, die Kerze auszublasen, wurde damals mit einem recht genialen Trick erreicht. Der interne ADC des Mikrocontrollers maß die Spannung über einem NTC, der angepasst werden musste. Leider unterschied sich die Temperatur der Atemluft aber zu wenig von der Umgebungstemperatur, als dass das

Atmen auf einen NTC unter normalen Bedingungen mehr als eine minimale Auswirkung auf den Widerstand gehabt hätte. Deshalb wurde der NTC durch seine eigene Verlustleistung vorgeglüht und durch das Pusten auf den NTC abgekühlt, was zu einer erkennbaren Spannungsänderung am Analogeingang des Mikrocontrollers führte. Unser neues Design sollte jedoch von einer einzigen 1,5-V-Batterie mit einer möglichst langen Standzeit versorgt werden, so dass die Entnahme von mindestens 15 mA aus der Batterie allein für die Erwärmung des „Sensors“ eine ziemliche Energieverschwendung bedeuten würde. Und wenn schon die Notwendigkeit des ADCs wegfiele, warum nicht den ganzen Mikrocontroller inklusive Software weglassen?

Der Elektrosmog-Schnüffler „Totally Archaic but Practical Interceptor of Radiation“ (aka TAPIR) [2] stammt aus dem Jahr 2012.



INFOS ZUM PROJEKT

Tags

Weihnachten, Feiertage, Haus & Garten

Niveau

Einsteiger – Fortgeschrittene – Experte

Zeit

Etwa 4 Stunden

Werkzeuge

Lötwerkzeuge (SMD und Durchkontaktierung)

Kosten

etwa 20 €

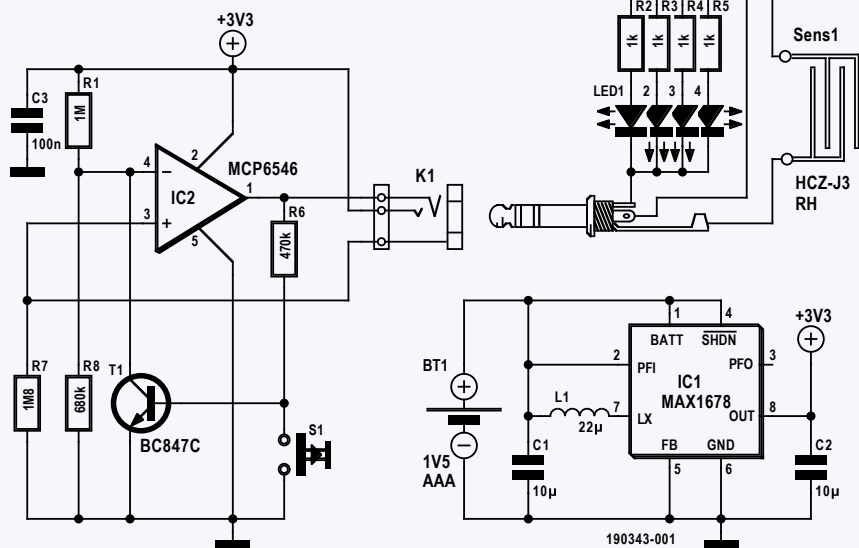


Bild 1. Schaltbild der Weihnachtskerze.

Wir haben dieses Projekt hauptsächlich aufgrund seines „mechanisches“ Charakters“ vorgestellt: ein Gehäuse aus kleineren Platinenstücken, das unter anderem einen Batteriehälter für eine AAA-Zelle und einen Stereo-Klinkenstecker enthält. In unserem neuen Projekt verbindet der Klinkenstecker eine kleine, kerzenflammenförmige Platine mit einem Sensor und vier flackernden 3-mm-LEDs. Zugegeben, das rechteckige Gehäuse sieht nicht gerade nach Kerze aus, aber man kann es ja vollständig „camouflagieren“, solange der Druckknopf und das Batteriefach zugänglich bleiben.

Advent, Advent, ein LEDlein brennt!

Die „Flackernde-Kerzen-LEDs“, die in unserem neuen Weihnachtsprojekt verwendet werden, sind billig und leicht im Internet erhältlich und stammen, man braucht es nicht zu erwähnen, aus im Fernen Osten beheimateten Webshops. Wie der Name schon errahnen lässt, besitzen sie eine eingebaute Elektronik, die das Licht wie eine echte Kerzenflamme flackern lässt. Es wird wie bei LEDs üblich nur ein externer Strombegrenzungswiderstand benötigt. Von außen sehen sie wie gewöhnliche 3-mm- oder 5-mm-LEDs aus und sind in verschiedenen Farben erhältlich. Für unsere Weihnachtskerze benötigen wir die 3-mm-Version, vorzugsweise in gelb oder orange.

Diese Bauteile werden in (fast) allen billigen batteriebetriebenen elektronischen Kerzen verwendet, so dass Sie, wenn Sie keinen Lieferanten für Einzelstückzahlen finden, ruhigen Gewissens ein kommerziell-

les Modell kaufen und dieses ausschalten können. Abgesehen von der LED werden Sie dabei nur einen Widerstand und eine 3-V-Knopfzelle oder zwei AA- oder AAA-Batterien vorfinden.

Und natürlich noch etwas mehr Hardware

Eine Versorgungsspannung von 1,5 V aus nur einer Batterie reicht aber nicht aus, um

die Flackernde-Kerzen-LEDs zum Flackern zu bringen. Deshalb haben wir einen kleinen Aufwärtswandler (IC1, ein MAX1678) verwendet, um die Batteriespannung auf bis zu 3,3 V zu erhöhen (**Bild 1**). Mit einer minimalen Eingangsspannung von 0,7 V ist dieses IC perfekt geeignet, um Ihren alten, anderweitig ausgesaugten Batterien den allerletzten Ladungsträger abzupressen. Bei unserem Prototyp reichte sogar bereits eine Eingangs-

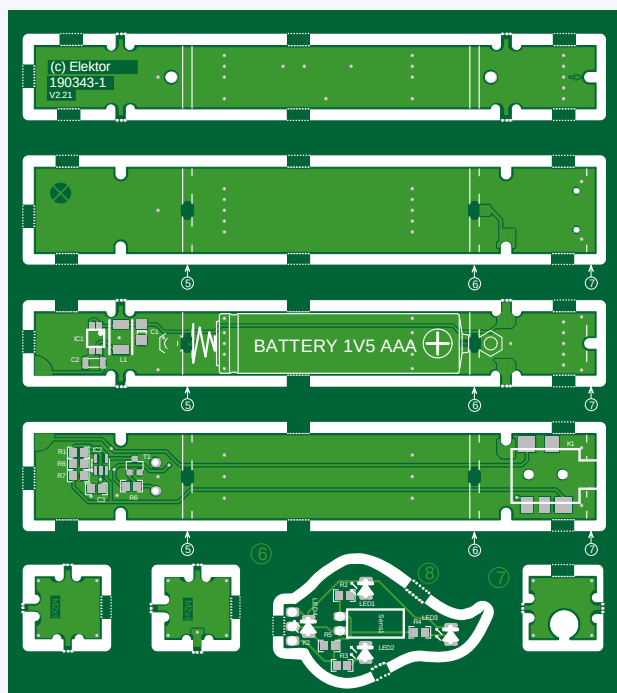


Bild 2. Acht Platinen in einem Stück!

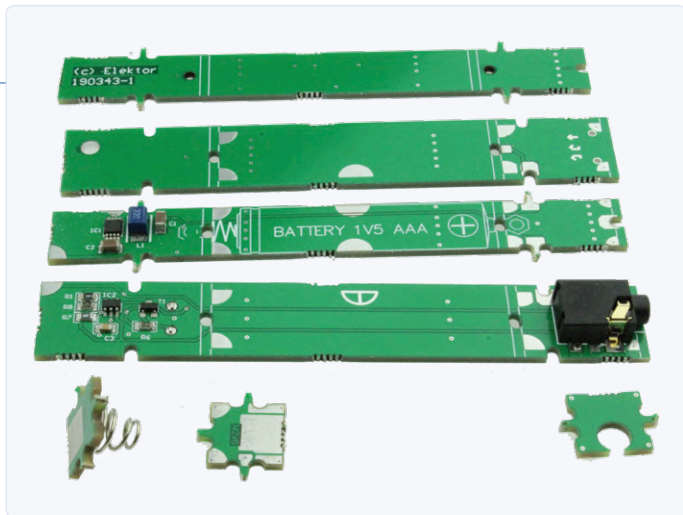


Bild 3. Verlöten Sie die (meisten) Bauteile, bevor Sie die Sammelplatine auseinandernehmen.



Bild 4a. Die Flammen-Platine mit den LEDs und dem Sensor mit angelötetem Klinkenstecker.

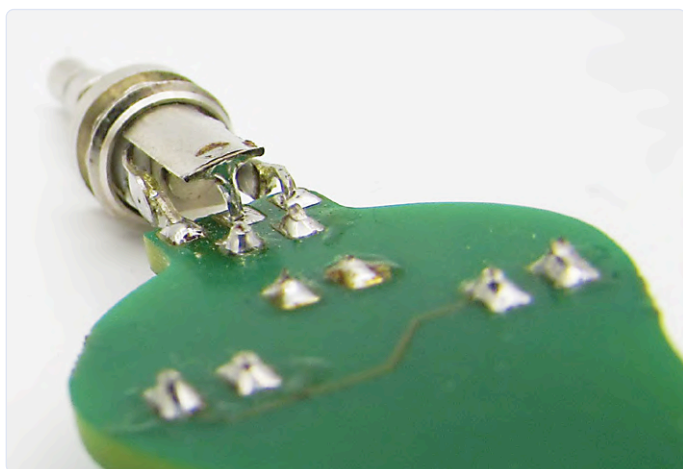


Bild 4b. Das Detail der Flammen-Platine zeigt die Verbindung zum Klinkenstecker.

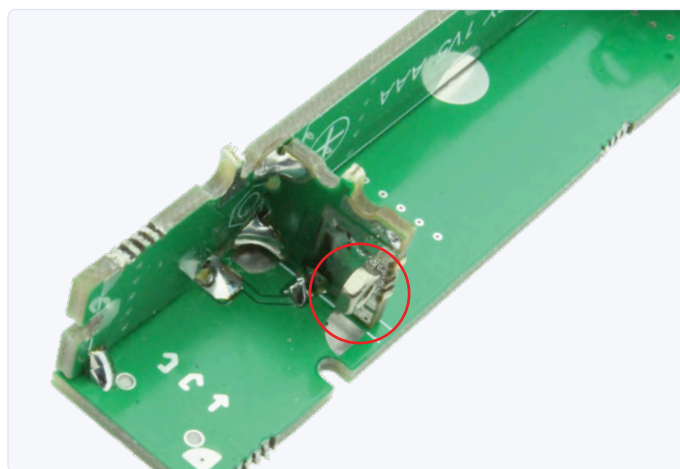


Bild 5. So wird die Batteriekontaktplatine montiert. Beachten Sie die Mutter zur Befestigung des Deckels.

spannung von 300 mV aus, um die Kerze anzuzünden!

Wie bereits erwähnt, ist die Messung der Temperatur, um festzustellen, ob die Kerze ausgeblasen wird, keine Option für ein batteriebetriebenes Gerät. Um die Dinge einfach, klein und erschwinglich zu halten, haben wir uns für einen Feuchtigkeitssensor HCZ-J3 von Multicomp zur Erfassung des Pustens entschieden. Obwohl dieser Sensor für den Einsatz in Wechselstromkreisen konzipiert ist, konnte er sich auch in unserer Anwendung auszeichnen. Der einzige Nachteil eines Feuchtigkeitssensors besteht darin, dass es einige Sekunden dauert, bis sein Widerstand wieder auf das Niveau der Umgebungfeuchtigkeit zurückgeht, mit anderen Worten, es dauert eine Weile, bis die ausgeblasene Kerze wieder angezündet werden kann.

Der Rest der Elektronik ist ziemlich einfach: Ein Komparator mit geringer Verlustleistung (IC2, MCP6546) triggert bei der Spannung an seinem nicht-invertierenden Eingang. Die

Spannung an diesem Pin steigt an, wenn der Feuchtigkeitspegel am Sensor steigt, also wenn man am Sensor ausatmet. Wenn sie höher ist als die Spannung am invertierenden Eingang (+/- 1,35 V vom Spannungsteiler R1 und R8), geht der Ausgang des Komparators auf high. Der Transistor T1 zieht dann den invertierenden Eingang auf Masse und hält die LEDs immer ausgeschaltet, bis der Taster S1 gedrückt wird, vorausgesetzt natürlich, dass der Spannungspegel am nichtinvertierenden Eingang (wieder) unter der Schwelle von 1,35 V liegt.

Aufbau der Kerze

Zuerst ein besonderer Hinweis zum Komparator-IC: Wenn Sie dieses Bauteil bestellen, stellen Sie sicher, dass es sich tatsächlich um den MCP6546R handelt. Das Suffix ist sehr wichtig, da Microchip drei SOT23-5-Versionen dieses Bauteils anbietet, die NICHT pinkompatibel sind! Die R-Version ist die einzig wahre!

Die Kerze und ihr Gehäuse bestehen aus acht kleineren Platinen, die zur einfacheren Produktion in einem Panel zusammengefügt sind (**Bild 2**). Sie können die Gerber-Dateien von der Elektor-Webseite herunterladen und einen Platinservice Ihrer Wahl die ätzenden Arbeiten überlassen.

Zu Ihrer eigenen Bequemlichkeit lassen Sie dieses Panel intakt, bis die (meisten) Komponenten wie in **Bild 3** gelötet sind. Die einzelnen Platinen sind von eins bis acht nummeriert und es gibt eine Art Legende auf dem Panel, die anzeigt, welches Teil der Platine wohin passt.

Es gibt silberfarbene Öffnungen in der grünen Lötmaske an mehreren Kanten und Ecken der Platinen. Dies sind Bereiche, die zur Verbindung der Platinen durch Löten verwendet werden. Einige der Verbindungen sind nur notwendig, um die Platinen zusammenzuhalten, andere sind auch wichtig, um die Schaltkreise auf den Platinen miteinander zu verbinden.

Obwohl die meisten der in diesem Projekt verwendeten Komponenten SMD-Bauteile sind, kann das Löten mit einem normalen LötKolben mit einer kleinen Spitze unter Verwendung eines dünnen Lötdrahts durchgeführt werden. Die Arbeit wird einfacher, wenn Sie einen Lotpastenspender, ein Heißluft-Lötwerkzeug oder natürlich einen Reflow-Ofen haben, aber es kann mit Standardwerkzeugen durchgeführt werden. In allen Fällen kann das Entlöten von Docht und Lötflusmittel nützlich sein, um überschüssiges Lot und (meistens unvermeidliche) Kurzschlüsse zu entfernen, insbesondere zwischen den Pins der beiden winzigen ICs. Montieren Sie zuerst die letzteren Bauteile und überprüfen Sie die Lötung, bevor Sie die anderen SMD-Bauteile hinzufügen. Schließlich ist es an der Zeit, die durchkontaktierten Bauteile einzulöten.

Vor dem Verlöten von K2, dem 3,5-mm-Stereo-Klinkenstecker, der die Flammen-Platine mit dem Kerzenstab verbindet, muss diese Platine aus dem Rahmen geschnitten werden. Entfernen Sie die Kappe vom Stecker und verwenden Sie einige kurze Drahtstücke, um die Leiterplatte mit dem Stecker zu verlöten. Schauen Sie sich **Bild 4** genau an, um zu sehen, wie sie miteinander verbunden werden müssen.

Um den „Deckel“ des Batteriefachs zu schließen, müssen Sie zwei Sechskant-Abstandshalter oder Muttern an die kleineren (Batteriekontakt-)Leiterplatten anlöten, wie in **Bild 5** gezeigt. Bei unseren Prototypen erwies sich dies jedoch als nicht unbedingt notwendig, da das Gehäuse fest genug sitzt, um den Deckel ohne Verwendung von Schrauben an seinem Platz zu halten.

Wenn sich alle Bauteile auf den Platinen befinden, ist es an der Zeit, alle Platinen aus dem Panel herauszunehmen, aber fangen Sie nicht sofort an, das Gehäuse zu löten! Setzen Sie zunächst die einzelnen Platinen zusammen, ohne irgendetwas zu verbinden, und vergewissern Sie sich, welche Platinen wie und wo passen. Eventuell müssen Sie kleine Reste des Panels entfernen (schneiden oder feilen), damit es perfekt passt.

Wenn Sie die Abstandshalter oder Muttern

zum Schließen des Kerzenhalters verlöten, ist dies auch der Zeitpunkt, um zu prüfen, ob sie gut auf die Schraubenlöcher im Deckel ausgerichtet sind. Falls nötig, justieren Sie sie nach. Löten Sie dann einige Drähte an, um die Platinen an ihren Lötflächen miteinander zu verbinden, und verwenden Sie zum Testen ein Tisch-Netzteil. Dadurch entsteht mehr Platz für Messungen und Reparaturen, falls erforderlich. Natürlich werden die Drähte später entfernt und die Leiterplatten zusammengeklötet, wenn die Kerze so funktioniert, wie sie sollte.

Und dann ist die Kerze schließlich einsatzbereit. Zugegeben, es geht nichts über echtes Kerzenlicht, um eine weihnachtliche Atmosphäre zu schaffen, aber wir hoffen,

Sie haben Fragen oder Kommentare?

Haben Sie Fragen oder Anmerkungen zu dem Artikel? Schicken Sie eine E-Mail an den Autor unter luc.lemmens@elektor.com oder kontaktieren Sie Elektor unter editor@elektor.com.

dass auch diese elektronische Version geschätzt wird. Wir wünschen Ihnen viel Spaß beim Bau dieses Weihnachtsgadgets und - natürlich - ein sehr frohes Weihnachtsfest! 

190343-02



STÜCKLISTE

Widerstände:

(alle SMD 0805)

R1 = 1 M

R2...R5 = 1 k

R6 = 470 k

R7 = 1M8

R8 = 680 k

Induktivität:

L1 = 22 µ, Größe 1812, TDK

NLC45323232T-220K-PF

Kondensatoren:

C1,C2 = 10 µ, 10%, 25 V, X7R, Größe 1206

C3 = 100 n, 10%, 50 V, X7R, Größe 0805

Halbleiter:

LED1...LED4 = LED „flickering candle“ gelb

3 mm (siehe Text)

T1 = BC847C

IC1 = Aufwärtsregler MAX1678EAU+

IC2 = Analog-Komparator MCP6546RT-E/OT

Außerdem:

K1 = 3,5-mm-Stereo-Klinkenstecker SMD
(CUI SJ1-3514-SMT-TR)

K2 = 3,5-mm-Stereo-Klinkenstecker
Batterie-Kontaktfeder (Keystone Electronic 211)

S1 = Taster, 6 mm (Alcoswitch FSM2JRT)

Sens1 = Feuchtigkeitssensor HCZ-J3
(Multicomp)

Platine 190343-1 V2.21



PASSENDE PRODUKTE

➤ Gerber-Platinendateien zum kostenlosen Download unter www.elektormagazine.de/190343-02

WEBLINKS

[1] Elektronische LED-Kerze, Elektor 12/2011: www.elektormagazine.de/magazine/elektor-201112/3983

[2] Ein TAPIR kann's erschnüffeln! Elektor 7-8/2012: www.elektormagazine.de/magazine/elektor-201207/4144

[3] Download: www.elektormagazine.de/190343-02

Durchstimmbarer Röhren-Sinusgenerator

Retro ist in...

Von Prof. Dr. Martin Ossmann

In meiner Jugend habe ich einen sogenannten O-V-2-Empfänger mit Röhren gebaut. Für einen Rückblick ins Röhren-Zeitalter wollte ich nun mal wieder etwas mit diesen Transistorvorläufern bauen. Einfache Empfänger und edle Verstärker gibt es wie Sand am Meer, weshalb mir der Sinn nach Ungewöhnlicherem stand.

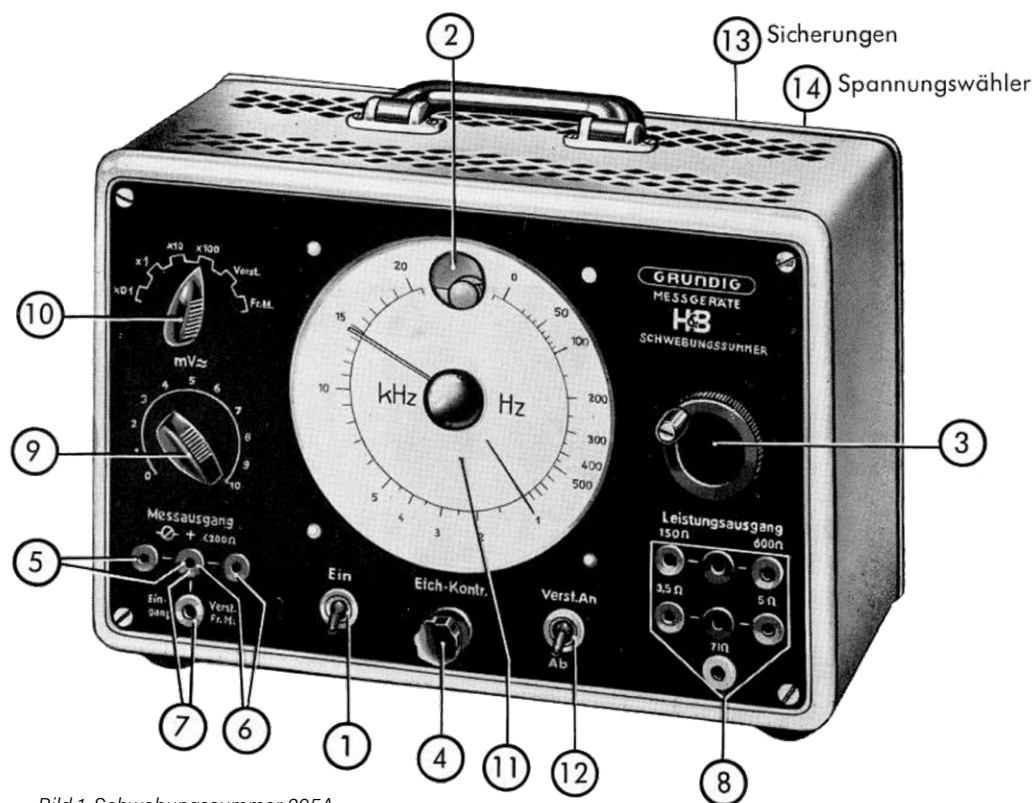


Bild 1. Schwebungssummer 295A.

Mit diesen Gedanken beschäftigt fiel mir die Anleitung des „Schwebungssummers 295A“ von Grundig, Hartmann und Braun in die Hände (siehe **Bild 1**). Bei diesem Gerät kann das Audioband von 20 Hz bis 20 kHz locker in einem Rutsch ohne Bereichsumschaltung durchgestimmt werden. Diese enorme Frequenzvariation von 1:1.000 ist mit einfachen Sinusgeneratoren nicht zu schaffen. Aus diesem Grund wurde hier auf das Überlagerungsprinzip zurückgegriffen.

Das Überlagerungsprinzip

Bild 2 zeigt die Prinzipschaltung. Oszillator 1 erzeugt die feste Frequenz f_1 (hier ca. 235 kHz). Oszillator 2 hingegen ist abstimmbar und

erzeugt daher die von 235 bis 270 kHz einstellbare Frequenz f_2 . Das sind weniger als 20 % Variation. Diesen kleinen Abstimmbereich kann man gut mit einem einfachen Oszillator mit Hilfe eines Drehkos realisieren. Die Ausgangssignale der beiden Oszillatoren speisen einen Mischer. An dessen Ausgang steht – neben einigen anderen Mischprodukten – ein Signal mit der Differenzfrequenz $f_2 - f_1$ zur Verfügung. Die anderen Mischprodukte (Summe von f_1 und f_2 sowie Summen und Differenzen ihrer Vielfachen) entfernt man mit einem passend dimensionierten Tiefpass. Dessen Ausgangssignal wird dann abschließend noch verstärkt, so dass es auch niederohmig zur Verfügung steht. Mit dem Überlagerungsprinzip kann man auch durchstimmbare HF-Generatoren bis in den GHz-Bereich aufbauen.

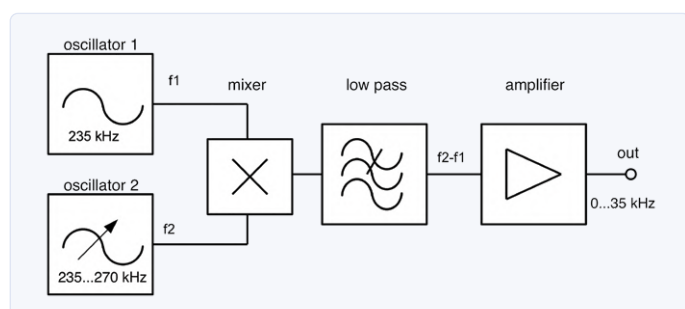


Bild 2. Funktionsprinzip des Überlagerungsgenerators.

Bild 3. Auszug aus dem Schaltplan des 295A.

Das Original

In **Bild 3** sieht man einen Auszug aus dem Schaltplan. Gezeigt werden die beiden Oszillatoren, der Mischer, der Tiefpass und die abschließende Verstärkerröhre. Röhre R6 2 erzeugt die feste Frequenz. Im Anodenkreis liegt ein Schwingkreis, der die Frequenz bestimmt. Die Rückkopplung erfolgt induktiv auf das Gitter.

Die variable Frequenz wird mit dem Trioden-Teil der ECH81 (R6 1, links) auf gleiche Weise erzeugt. Die Frequenz ist mit einer variablen Induktivität (Variometer L5) einstellbar. Die beiden Oszillatorsignale liegen dann an den Gittern g1 und g3 des rechten Heptoden-Teils der ECH81. Die Mischprodukte stehen an der Heptoden-Anode bereit.

Das Tiefpassfilter besteht aus L1, L2 und L3 plus C1, C2, C3 und C4. Das gefilterte Ausgangssignal gelangt über ein Poti zur Einstellung der Signalamplitude an die als Kathodenfolger geschaltete Röhre R6 3. An ihrer Kathode steht schließlich das niederohmige Sinussignal zur Verfügung.

Mein Nachbau

Um den Nachbau leichter zu gestalten habe ich ein paar Modifikationen erprobt. Statt mit einem Variometer wird in meiner Version die Frequenz mit einem Drehkondensator eingestellt. Die Rückkopplungen der Oszillatoren wurden so modifiziert, dass man statt Trafos Spulen mit Anzapfung verwenden kann. Meine Tests fanden auf dem Basteltisch in freifliegender Verdrahtung statt, wie **Bild 4** beweist.

In **Bild 5** ist der vollständige Schaltplan zu bewundern. Der variable Oszillator ist mit der Triode V1 realisiert. Sein Signal gelangt über einen variablen Spannungsteiler an Gitter 3 der Mischerröhre V2. Der

Bild 4. Experimente auf dem Basteltisch.

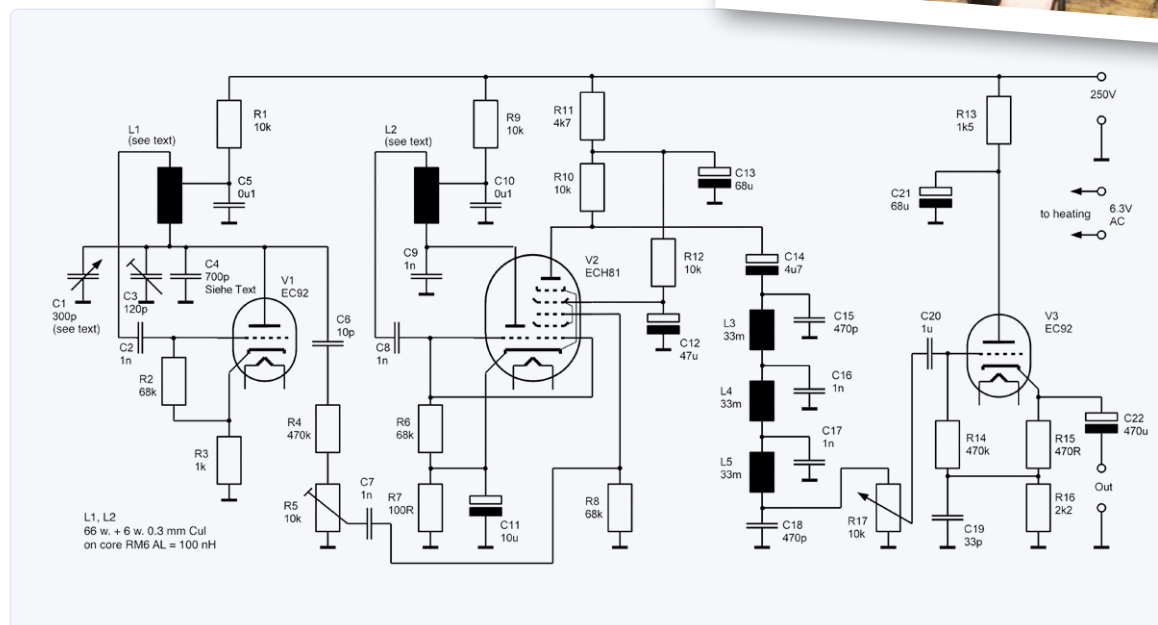
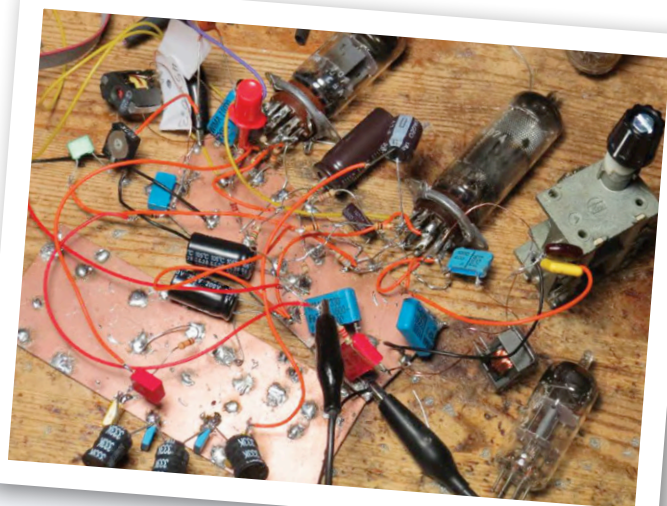
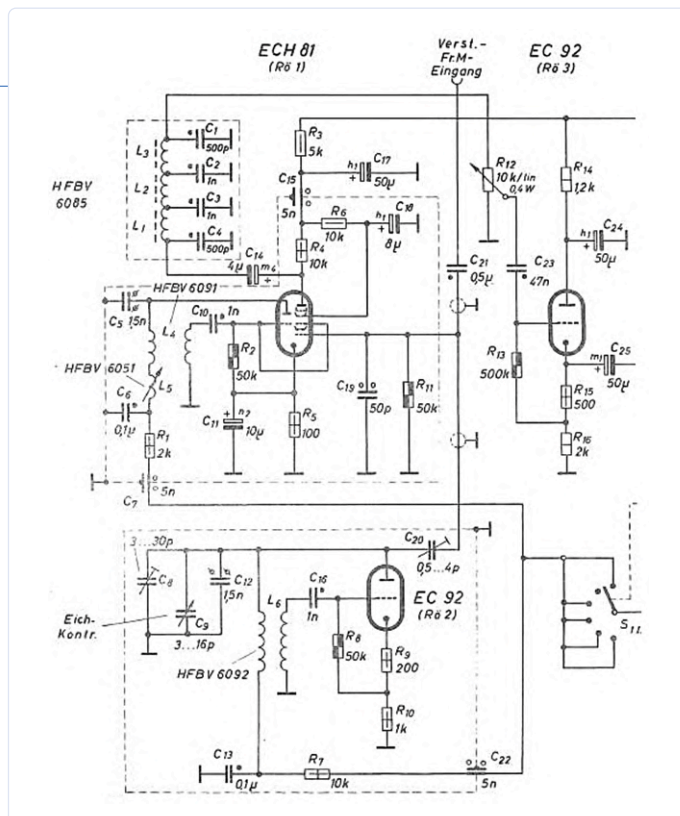


Bild 5. Schaltplan des optimierten Nachbaus. Der vollständige Aufbau ist in Bild 6 zu sehen (ohne Netzteil).

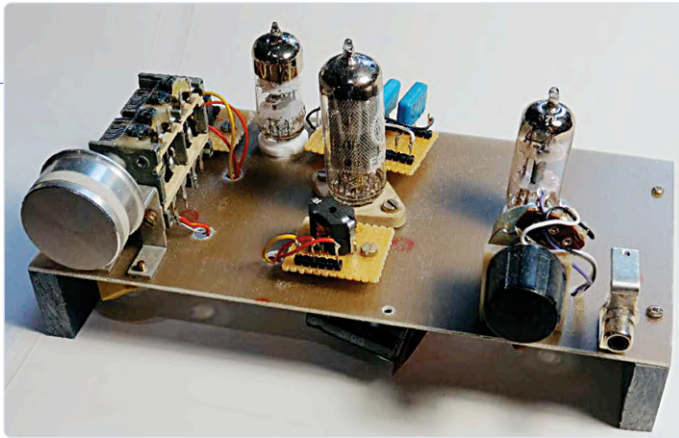


Bild 6. Mein konkreter Aufbau des Sinusgenerators.

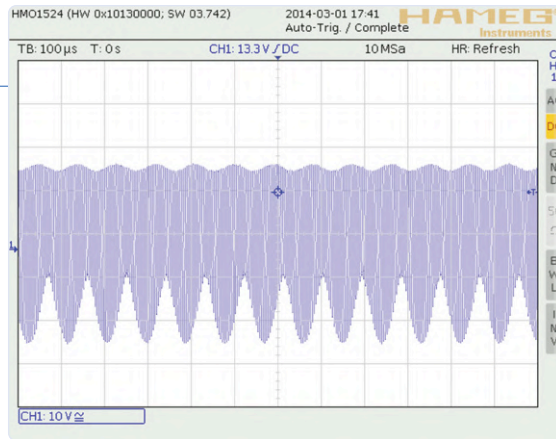


Bild 7. Signal an der Anode der Mischerröhre.

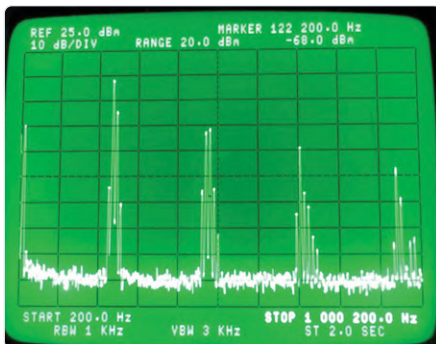


Bild 8. Spektrum des Mischersignals.

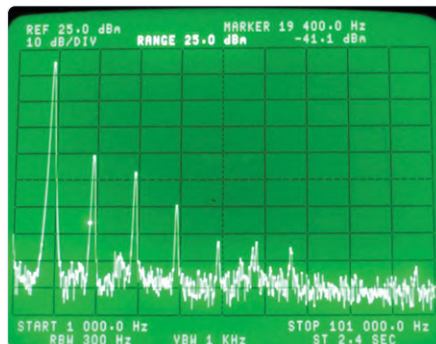


Bild 9. Ausgangsspektrum bei hoher Aussteuerung.

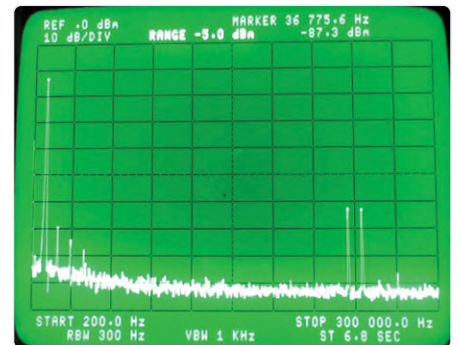


Bild 10. Ausgangsspektrum bei niedriger Aussteuerung.

Triodenteil von V2 wird für den Festfrequenzoszillator genutzt. Sein Ausgangssignal gelangt an Gitter G1 der Heptode. An der Heptoden-Anode steht dann das Mischersignal zur Verfügung.

Messungen

Um die Signalqualität zu überprüfen und die Funktionsweise dieses Prinzips nachzuvollziehen habe ich an meinem Prototyp (Bild 6) ein paar Messungen durchgeführt. An den Anoden der Oszillatoren beträgt die Amplitude der Signale ca. 200 V_{SS}. Besonders interessant ist natürlich das Anodensignal der Mischröhre, das in Bild 7 zu sehen ist. Die schnelle Schwingung entspricht dem hochfrequenten Zweiton-signal. Die untere Hüllkurve entspricht dem erzeugten Sinussignal. Letzteres wird mit einem Tiefpass herausgefiltert.

Bild 8 zeigt das Spektrum des Anodensignals der Mischröhre. Man kann gut die zahlreichen Mischprodukte der beiden Sinussignale erkennen. Mit dem Trimpoti R5 lässt sich die Amplitude des Ausgangssignals einstellen. Gleichzeitig ändert sich damit die Aussteuerung der Mischröhre.

Stellt man z.B. R5 so ein, dass an Gitter G3 eine Spannung von 8 V_{SS} liegt, ergibt sich eine Ausgangsamplitude von etwa 7 V_{SS}. Dieser hohe Pegel hat natürlich einen hohen Klirrfaktor des Ausgangssignals zur Folge. **Bild 9** zeigt das zugehörige Spektrum. Die erste Oberwelle liegt nur ca. 35 dB unter dem Nutzsignal. Der resultierende Klirrfaktor von über 1 % ist nicht sonderlich gut.

Stellt man an G3 aber eine Amplitude von nur 1 V_{SS} ein, beträgt die Ausgangsspannung immerhin noch 2 V_{SS} und das zugehörige Spektrum von **Bild 10** sieht viel besser aus. Die erste Oberwelle ist nun mehr als 50 dB schwächer als die Grundschwingung. Das Resultat entspricht in etwa einem Klirrfaktor von 0,3 %. Das ist gar nicht so schlecht, doch

im Ausgangssignal sind auch noch Reste der beiden Sinussignale bei 235 und 250 kHz enthalten. Diese gelangen durch kapazitive Kopplung über das Tiefpassfilter hinweg ins Ausgangssignal. Hier würde eine gute Abschirmung um die HF führenden Teile helfen.

Abschließend kann man feststellen, dass man mit drei Röhren und einer Handvoll Bauteile einen durchstimmbaren Sinusgenerator bauen kann, der ein brauchbares Sinussignal über den gesamten Audiobereich erzeugt.

PS. Da sich ja nicht jeder Elektroniker auch als Amateurfunker betätigt: „0-V-2“ ist klassischer HF-Speak. Die erste Zahl gibt an, wie viele HF-Verstärkerstufen vor dem Detektor vorhanden sind. Die Zahl nach dem „V“ bezeichnet die Anzahl an NF-Verstärkerstufen. Ein Empfänger des Typs 0-V-2 ist also ein sogenanntes Audion mit einer Röhre ohne HF-Vorstufen und zwei NF-Verstärkerstufen. ❏

190312-01



PASSENDE PRODUKTE

- > **Röhren-Projekte von 6 bis 60 V (PDF)**
www.elektor.de/roehren-projekte-von-6-bis-60-v-pdf
- > **Röhrenverstärker-Schaltungen (PDF)**
www.elektor.de/roehrenverstaerker-schaltungen-pdf
- > **JOY-iT JDS6600 Signal Generator (60 MHz) & Frequency Counter (100 MHz)**
www.elektor.de/18714



Das Picoscope 2204A-D2.

Oszilloskope der Serie Picoscope 2000

Einfach ausgedrückt werden Oszilloskope eingesetzt, um die Wellenformen elektronischer Signale zu messen und zu analysieren, so dass man die Zuverlässigkeit und ordnungsgemäße Funktion der entworfenen elektronischen Systeme überprüfen kann. Die häufigsten Benutzer sind Ingenieure, Wissenschaftler, Servicetechniker, Lehrer und Schüler. Einer der renommiertesten Hersteller von Oszilloskopen ist ein britisches Unternehmen: Pico Technology.

Hauptigenschaften und -Funktionen

PicoScope 2000 ist eine Basisserie von PC-Oszilloskopen aus einem breiten Angebot von Pico Technology. Die Geräte haben kein Display und für den Betrieb ist ein Computer oder Laptop mit Windows, Mac oder Linux erforderlich. Die Serie umfasst 2-Kanal-Geräte, 4-Kanal-Geräte sowie Mischsignal-Oszilloskope (MSO) mit zwei analogen Eingängen und 16 digitalen Kanälen. Alle Oszilloskop-Modelle sind vom Hersteller mit einem Funktions- und einem *Arbitrary Waveform Generator* für beliebige Wellenformen ausgestattet. Die mitgelieferte Software ermöglicht die Analyse von Spektren und seriellen Busprotokollen (CAN, LIN, SENT, SPI, I2C, I2S und einige andere). Die MSO-Modelle ermöglichen auch die Analyse von logischen Zuständen. Alle diese Geräte bieten ein attraktives Verhältnis von Qualität und Parametern zum Preis. Für alle PicoScope-Modelle ist die Software kostenlos im Paket enthalten, mit der Möglichkeit eines kostenlosen Updates von der Website des Herstellers. Darüber hinaus erhalten Kunden, die sich

für eines der Modelle entscheiden, die umfangreichste und im Grunde einzige funktionierende Version, die alle dafür verfügbaren Optionen enthält. Dies bedeutet, dass keine zusätzlichen Erweiterungen zu kaufen sind - der Hersteller hat jedes Modell mit allen Funktionen ausgestattet, die er anbieten konnte, und der Kunde erhält eine Version mit vollständiger Software, USB-Kabel (Netzteil) und Messsonden, die ebenfalls im Paket enthalten sind. Ein weiteres Merkmal, das diese kompakten Geräte von der Konkurrenz unterscheidet, ist die perfekte Visualisierung von Messungen auf einem PC-Bildschirm und die Messung von Wellenformen bis zu 100 MHz (analoger und digitaler Bereich). Die Modelle PicoScope 2000B bieten zusätzliche Vorteile wie z.B. einen großen Speicher (bis zu 128 Millionen Abtastwerte), eine höhere Bandbreite (von 50 bis 100 MHz) und eine schnellere Abtastung, was besonders für die erweiterte Wellenformanalyse (einschließlich serieller Dekodierung) nützlich ist.

Kompakt, robust und mobil

Die ultrakompakten Oszilloskope von PicoScope, unter denen die Serie 2200A die günstigsten und kleinsten Geräte in der gesamten Familie umfasst, können überallhin mitgenommen werden; die robuste Konstruktion sorgt für Sicherheit beim Transport. Sie müssen sich auch keine Gedanken über den Zugang zum Stromnetz oder einer externen Batterie machen, da die Geräte über ein USB-Kabel versorgt werden, das an den Computer angeschlossen ist (USB 2.0-Anschluss). Der große Vorteil der gesamten PicoScope-2000-Familie ist die von MS Windows-Systemen bekannte Schnittstelle, die die Software intuitiv und benutzerfreundlich macht (Datenexport in eine Excel-Datei). Der große Vorteil der Software sind kostenlose Updates, die immer die neuesten Funktionen und Verbesserungen bieten.

Die Oszilloskope PicoScope 2000 sind klein genug, um in eine Laptasche zu passen, und bieten eine ähnliche Leistung wie größere Geräte anderer Hersteller. Jene kosten häufig ein Vielfaches mehr als Produkte von PicoScope. Die Spezifikation der Oszilloskope von PicoScope spricht für sich: Selbst 2-Kanal-Modelle ermöglichen die Erfassung von Signalen mit einer Abtastrate von bis zu 1 GigaSamples pro Sekunde (1 GS/s). Darüber hinaus kann für periodische Signale bei Verwendung des ETS-Modus die Abtastfrequenz auf 10 GS/s erhöht werden, was in Kombination mit einem Puffer für 40.000 Abtastungen und einer Bandbreite von 100 MHz die Verwendung von Oszilloskopen aus dieser Familie in einem sehr breiten Anwendungsbereich ermöglicht.

Jedes Oszilloskop der PicoScope 2000-Serie verfügt über einen integrierten Ausgangssignalgenerator, der Standardwellenformen aus der Wellenformbibliothek erzeugen kann. Aber das ist nicht alles: Sie können auch Ihre eigenen, beliebigen Wellenformen definieren, wobei Ihnen die volle Leistung des Generators zur Verfügung steht.

200508-01



Das Picoscope 2206B.

Mehr Infos: www.tme.eu/de/news/library-articles/page/42083/picoscope-2000-series-oscilloscopes

Generator Rigol DG2072



Von **Philippe Demerliac** | Cyrob (Frankreich)

Legen Sie Wert auf eine hohe Signaltreue bei einem zuverlässigen und leistungsfähigen Signalgenerator? Die DG2000-Serie von Rigol könnte eine gute Lösung sein. Die hybride Touchscreen-Schnittstelle und das industrielle Design macht sie zu einer intelligenten Option für Ihr Labor; über Standard-Ethernet und USB wird die programmatische Steuerung des Geräts ermöglicht. Werfen wir einen genaueren Blick auf den Funktionsgenerator DG2072.

Philippe Demerliac betreibt Cyrob.org und veröffentlicht Videos unter https://youtu.be/tm_VqCWMY34.



Bild 1. Die Vorderseite des DG2072.



Bild 2. Die Rückseite des DG2072

Ich habe zwar gezögert, den Funktionsgenerator DG2072 in die „Profi-Kategorie“ einzuordnen, aber aufgrund des Preis/Leistungsverhältnisses stellt er nicht nur eine solide Investition für Hobbyelektroniker dar, die den größten Teil ihrer Bedürfnisse über viele Jahre hinweg decken sollte, sondern dürfte auch den Weg in das eine oder andere professionelle Labor finden.

Schon vor ein paar Jahren habe ich den hochwertigen Generator Rigol DG4162 gekauft, aber der DG2072 ist noch besser! Er verfügt über ein Touchinterface, wie es zunehmend zum Standard für Messgeräte wird, an dessen Bedienung ich mich sehr schnell gewöhnt habe. Ich bin sogar oft enttäuscht, wenn ich auf einen Bildschirm eines Messgeräts drücke, das überhaupt keine Touchfunktion besitzt. Alle Messaufgaben, für die ich den Generator eingesetzt habe, zeigten, dass der DG2072 gut innerhalb der Spezifikationen lag und gut kalibriert war. Die DG2000-Reihe ist recht innovativ. Der Dual-Tone-Modus, die Erzeugung von Logiksignalen und der Sequenzmodus sind sehr nützlich. Der Preis

ist sicherlich höher als bei anderen konkurrierenden Modellen, aber wenn Sie auf der Suche nach einem hochwertigen DDS-Generator mit vielfältigen Funktionen und hochqualitativen Signalen sind, sollte das Geld für einen Generator der DG-2000-Reihe wie den DG2072 gut angelegt sein. Die Generatoren sind zu verschiedenen Preisen erhältlich, je nach maximal möglicher Frequenz. Für den Amateurbereich sollte in der Regel die 50-MHz-Version völlig ausreichend sein. Man könnte meinen, dass Geräte-Design wäre ein wenig „aggressiv“, aber das hängt einfach von Ihrem Geschmack ab.

Vor- und Nachteile

Lassen Sie uns mit einigen Vorteilen beginnen:

- Eine angenehme Touchscreen-Oberfläche, die sehr schnell reagiert.
- Modulationsmöglichkeiten.
- Funktionsreichtum.
- Die große Anzahl von Wellenformen ist Standard.



Bild 3. Bildschirm nach dem Einschalten.



Bild 4. Kanal 2 im Harmonischen Modus.



Bild 5. Im Wobbelmodus ist ein Marker verfügbar.



Bild 6. Der Sequenz-Modus ermöglicht die Verkettung mehrerer Wellenformen im „Direkt“-Modus.

- > Der Benutzer kann die Ausgangsimpedanz wählen.
- > Der Benutzer kann die Amplituden in V_{SS} , V_{rms} , dBm usw. definieren.
- > Die „Marker“-Funktion im Wobbelmodus.
- > Erweiterte Modi (Bit, RS232, Sequenz).
- > Die „Dual Tone“-Funktion.
- > SiFi-II-Technik für eine bessere Wiedergabe.
- > Der Netzschalter befindet sich auf der Frontplatte.
- > Die wunderbare Stille (kein Lüftergeräusch).
- > Die Qualität des Touchscreens und der sehr deutlichen Anzeige.
- > Die Auflösung von 16 bit (65536 Werte oder 152 μV pro Punkt unter 50 Ω).
- > 16 Mega-Punkte bei einer Abtastung und einer Geschwindigkeit von 250 MSa/s.
- > Geringes Phasenrauschen (10 MHz: ≤ 105 dBc/Hz).
- > Allgemeine Qualität.

Und nun wollen wir auch die Nachteile betrachten:

- > Buchsen auf der Rückseite.
- > Die Buchsen, die je nach Modus als Ein- oder Ausgänge funktionieren.
- > Der Schalter auf der Vorderseite ist „billig“. Warum nicht einen Druckknopf?
- > Die Taster auf der Vorderseite sind etwas hart.
- > Der Frequenzmessmodus: die Frequenzanzeige ist im Frequenzmessmodus zu klein, es können kein Offset definiert und keine Screenshots erstellt werden, die begrenzte Eingabedynamik.

Der erste Kontakt

Bild 1 zeigt die Vorderseite des Geräts. Beachten Sie, dass die BNC-Buchsen je nach Modus als Ein- oder Ausgänge arbeiten (**Bild 2**). **Bild 3** zeigt den Bildschirm nach dem Einschalten. Kanal 1 ist rot, Kanal 2 (**Bild 4**) ist blau. Kanal 2 wird im Harmonic-Modus betrieben. Der Marker im Wobbelmodus erleichtert die Anwendung (**Bild 5**).

Der Sequenzbetrieb ermöglicht die Verkettung mehrerer Wellenformen im „Direkt“-Modus, wodurch die Akkumulation von Phasenfehlern verhindert wird (**Bild 6**). Die Simulation pseudo-zufälliger Sequenzen von seriellen Bits kann bei der Entwicklung digitaler Übertragungssysteme ziemlich hilfreich sein (**Bild 7**).

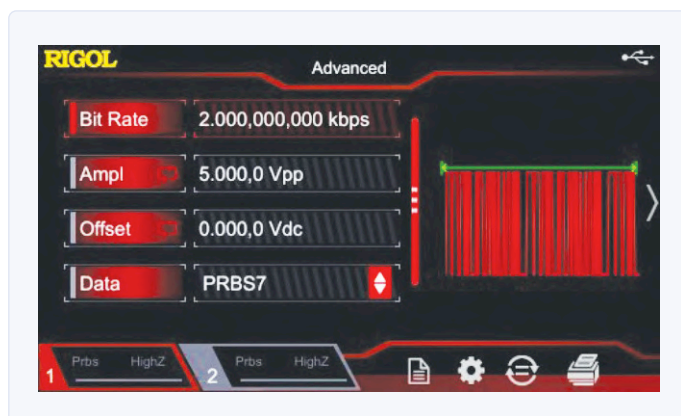


Bild 7. Die Simulation von Pseudozufallsfolgen serieller Bits kann bei der Entwicklung digitaler Übertragungssysteme helfen.

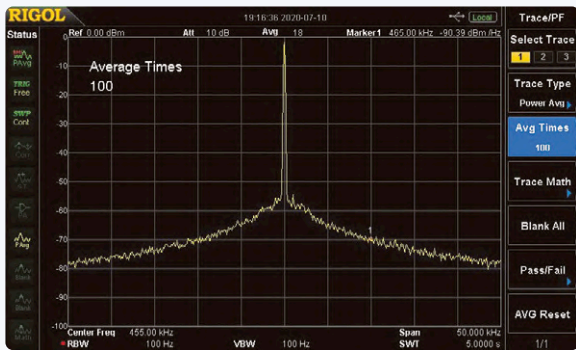


Bild 8. Spektralanalyse.

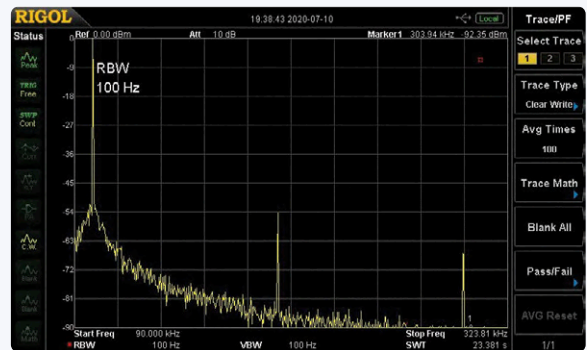


Bild 9. Spektrum eines Sinussignals von 100 kHz bei 0 dBm.

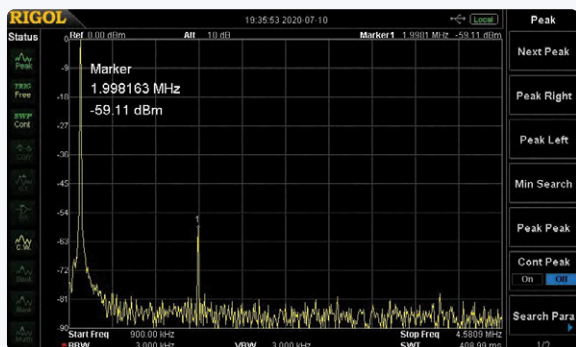


Bild 10. Spektrum eines Sinussignals von 1 MHz bei 0 dBm.

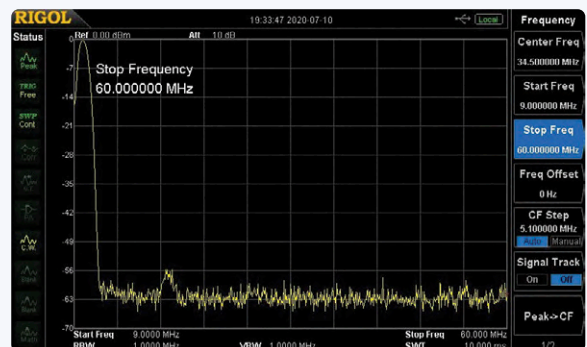


Bild 11. Spektrum eines Sinussignals von 10 MHz bei 0 dBm.

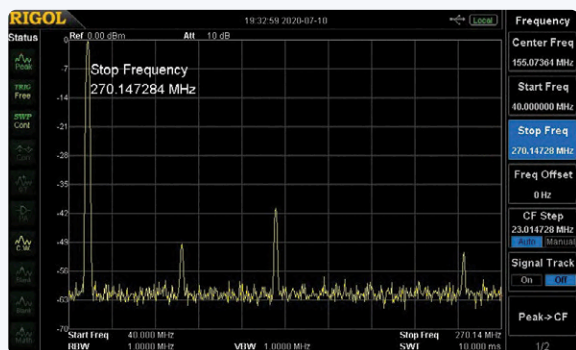


Bild 12. Spektrum eines 50-MHz-Sinussignals bei 0 dBm.

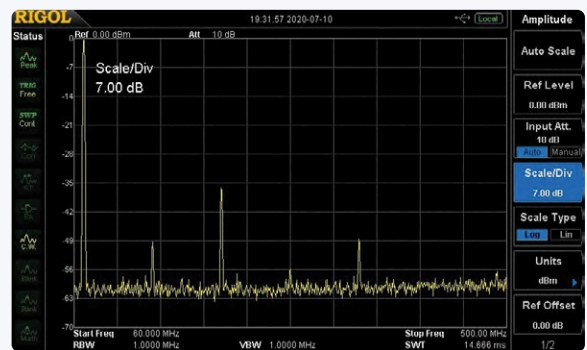


Bild 13. Spektrum eines 70-MHz-Sinussignals bei 0 dBm.

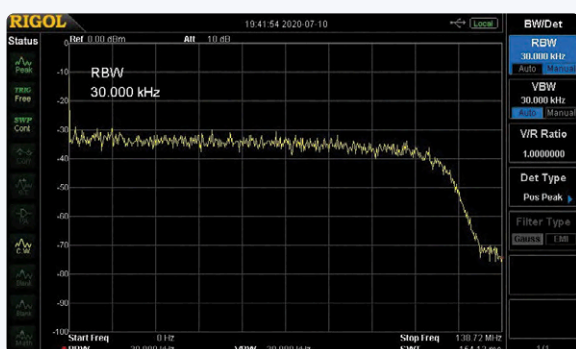


Bild 14. Im Modus „Weißes Rauschen“ ist der Pegel bis 100 MHz nahezu stabil.

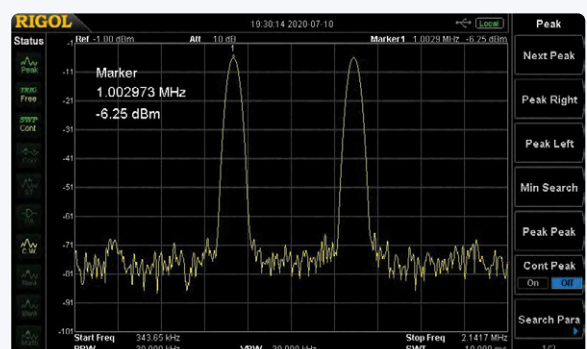


Bild 15. Der Dual-Tone-Modus und seine Kamel-Kurve (oder ist es ein Dromedar?).



Bild 16. Ich habe die Anschlüsse nach vorne verlegt.

Spektralanalyse

Ich habe einige Messungen mit dem Spektrumanalysator Rigol DG815TG durchgeführt. Ich wollte das auf 10 kHz normierte Phasenrauschen eines Trägersignals messen, aber mein Analysator war dafür nicht empfindlich genug (Bild 8).

Bild 9 zeigt das Ausgangsspektrum einer Sinuswelle von 100 kHz bei 0 dBm. Die erste Harmonische liegt bei -54 dB. Bild 10 zeigt das Spektrum eines Sinussignals von 1 MHz bei 0 dBm. Die erste Harmonische liegt bei -59 dB.

Bild 11 zeigt das Spektrum eines Sinussignals von 10 MHz bei 0 dBm. Die erste Harmonische liegt bei -56 dB. Bild 12 zeigt das Spektrum eines 50-MHz-Sinus bei 0 dBm. Die erste Harmonische liegt bei -49 dB, die dritthöchste bei -40 dB.

Bild 13 zeigt das Spektrum einer 70-MHz-Sinuswelle bei 0 dBm. Die erste Harmonische liegt bei -49 dB, die dritthöchste bei -37 dB. Im Modus „Weißes Rauschen“ ist der Pegel bis zu 100 MHz nahezu stabil, ein ausgezeichneter Wert (Bild 14)! Bild 15 zeigt den „Dual Tone“-Modus und seine „Trampeltier“-Kurve.

Modifikationen

Wie üblich stören mich Anschlüsse auf der Rückseite, und so brachte ich sie wieder in den Vordergrund (Bild 16)! Natürlich habe ich das Gerät nicht auseinandergenommen, sondern nur ein zusätzliches Blech mit den Buchsen an der Seite befestigt (Bild 17). Trotz der etwas seltsamen Form, die Sie in Bild 18 und Bild 19 erkennen können, war das Werkstück schnell gefertigt und die Verbindungskabel mit den bedruckten Schrumpfschlauchhülsen an den Buchsen auf der Rückseite angeschlossen (Bild 20). ◀

200453-02



Bild 17. Die neuen Anschlüsse befinden sich nun auf der Seite.



Bild 18. Sägen und Feilen: So sieht das Zusatzblech aus.



Bild 19. Das Zusatzblech ist mit den Gehäuseschrauben befestigt.



Bild 20. Der bedruckte Schrumpfschlauch hilft, Verwechslungen zu vermeiden.



PASSENDEN PRODUKTE

> Funktions-/Arbiträr-Generator Rigol DG2072:
www.elektor.de/dg2072

Sie haben Fragen oder Kommentare?

Haben Sie Fragen oder Anmerkungen zu diesem Artikel? Senden Sie eine E-Mail an den Autor unter philippe.demerliac@free.fr oder kontaktieren Sie Elektor unter editor@elektor.com.

Mehr Infos im Video

Möchten Sie mehr wissen? Sehen Sie Cyrob's Video über den Rigol DG2072 (in französischer Sprache) unter https://youtu.be/tm_VqCWMY34.

WEBLINK

[1] Review im Video: https://youtu.be/tm_VqCWMY34

Ein Beitrag von
 Autor: Philippe Demerliac

Redaktion: C. J. Abate
 Übersetzung: Rolf Gerstendorf

Layout: Giel Dols

KiCad

-Plugins und Add-Ons

Eines der tollen Merkmale des freien und quelloffenen Schaltplan- und PCB-Design-Tools KiCad ist, dass es erweiterbar ist. Jeder kann in Python Plugins und Add-ons dafür erstellen. Hier stellen wir einige davon vor.

KiCad ist ein freies und quelloffenes Werkzeug zur Erstellung von Schaltplänen und zum Entwurf von Platinen (PCBs). Seit seiner ersten Veröffentlichung im Jahr 1992 hat es stetig an Popularität gewonnen. Auch wir bei Elektor setzen es immer häufiger ein. Eine der coolen Eigenschaften von KiCad ist, dass es erweiterbar ist, so dass jeder, der sich für (Python-)Programmierung (und Elektronikdesign) interessiert, Plugins und Add-Ons dafür entwickeln kann. Dies hat dazu geführt, dass eine Vielzahl von Werkzeugen und Dienstprogrammen online zur Verfügung stehen. Nicht alle bringen große Vorteile und einige sind nützlicher als andere, aber es lohnt sich, ein paar dieser Zusatztools auszuprobieren.

Unsere Auswahl

Hier sind also einige Plugins und Tools, die wir für Sie handverlesen haben. Eine umfangreiche Übersicht der Plugins und Tools für KiCad wird unter <https://github.com/xesscorp/kicad-3rd-party-tools> gepflegt.

Stretch - Action-Plugin zum Erstellen „hübscher“ Platinenlayouts. Obwohl KiCad eigentlich ein Zeichenprogramm ist, ist es in dieser Hinsicht paradoxerweise nicht besonders. Das Zeichnen gerader Linien funktioniert gut, aber Kurven und komplexe Formen gehören nicht zu den Stärken von KiCAD. Stretch schafft hier Abhilfe, indem Platinenlayouts mit Inkscape getauscht werden können. Typische

Platinenentwürfe werden in Pcbnew erstellt, künstlerische Feinheiten und andere komplexe Operationen werden in Inkscape hinzugefügt. Dann wird der Entwurf zurück nach Pcbnew exportiert, um zum Beispiel Produktionsdateien zu erzeugen.

<https://github.com/JarrettR/Stretch>

KiKit - Python-Toolbox für die automatische Erstellung von Panels, den Export von Produktionsdateien und die Erstellung von Platinen-Präsentationsseiten. Um diese Toolbox gut nutzen zu können, sind einige Kenntnisse der Python-Programmierung nützlich. Beachten Sie, dass diese Toolbox aufgrund von Problemen von Python zusammen mit KiCad unter Windows nicht funktioniert (siehe Kasten).

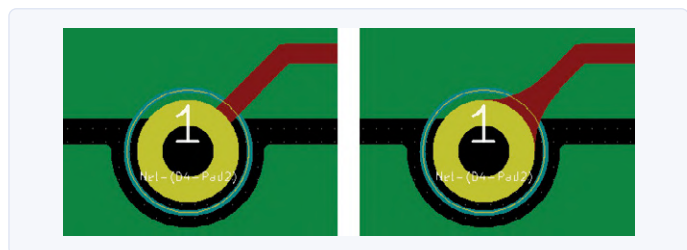
<https://github.com/yaqwsx/KiKit>

Interactive HTML BOM - Action-Plugin zur Erstellung nicht nur einer Stückliste (Bill of Materials, BoM), sondern auch eines interaktiven Dokuments zur Bauteilplatzierung, das grafisch hervorhebt, wo sich ein Bauteil aus der BoM auf der Platine befindet. Die Bauteile können nach Wert oder einzeln durchsucht werden. Dieses Plugin weiß nicht nur, wo die Bauteile platziert werden, sondern kennt auch alle Netze. Durch die schnelle Lokalisierung eines Bauteils oder eines Signals kann die Platine schneller montiert, getestet oder repariert werden.

<https://github.com/openscopeproject/InteractiveHtmlBom>

RF-Tools for KiCAD - Diese Sammlung von Action-Plugins erleichtert den Entwurf von Hochfrequenz-Platinen mit Funktionen wie Eckenabrundung und Längenberechnung von Leiterbahnen, Fencing von Durchkontaktierungen und Lötmasken-Expansionen. Sie enthält auch einige Tools, mit denen man RF-Footprints erstellen kann.

<https://github.com/easyw/RF-tools-KiCAD>



Lötage vor (links) und nach (rechts) der Bearbeitung mit dem Action Plugin Teardrops. Sie können es unter https://github.com/NilujePerchut/kicad_scripts herunterladen.

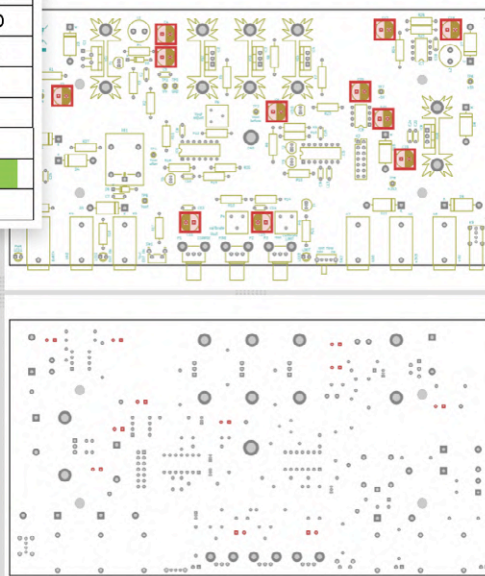
KiCad-Diff - Python3-Programm zur Verfolgung von Platinen-Revisionen mit Git, SVN oder Fossil. Es kann sowohl grafische als auch textbasierte Berichte als Webseite erzeugen, die die Unterschiede zwischen zwei Revisionen eines Board-Designs anzeigt.

motordrv

Rev:

2020-05-16 ...

</



Beispiel für einen interaktiven Bericht, der durch das Action Plugin Interactive HTML BOM generiert wird.

Funktioniert unter Linux, Mac-OS und Windows.

<https://github.com/Gasman2014/KiCad-Diff>

KiField - Ein Python-Tool, mit dem alle Bauteilfelder aus einem Schaltplan oder einer Bibliothek extrahiert und zur Massbearbeitung in eine Tabelle eingefügt werden können. Feldwerte können geändert und sogar neue Felder hinzugefügt werden. Anschließend können die aktualisierten Felder wieder in den Schaltplan oder die Bibliothek eingefügt werden.

<https://xesscorp.github.io/KiField> 

200429-02

Sie haben Fragen oder Kommentare?

Haben Sie Fragen oder Anmerkungen zum Artikel? Schicken Sie eine E-Mail an den Autor unter clemens.valens@elektor.com oder kontaktieren Sie Elektor unter editor@elektor.com.

Ein Beitrag von

Idee, Text und Bilder: **Clemens Valens**

Redaktion: **CJ Abate**

Layout: **Giel Dols**

WAS SIE VIELLEICHT NOCH WISSEN SOLLTEN

- › Bei korrekter Installation sind sogenannte Action Plugins im Pcbnew-Menü Tools -> Externe Plugins verfügbar. Sie beziehen sich stets auf den gerade in Pcbnew geladenen Entwurf. Stand-alone-Tools dagegen, deren Funktionen man nutzen möchte, um mit Daten aus den Dateien arbeiten zu können, benötigen Pcbnew nicht.
- › Unter Windows installiert KiCad seinen eigenen Python-Interpreter, was bedeutet, dass das pcbnew-Modul in einer systemweiten Python-Installation, die Sie vielleicht besitzen, nicht installiert ist. Bestimmte Add-ons und Toolboxes funktionieren daher unter Windows möglicherweise nicht, überprüfen Sie dies also, bevor Sie sich über eventuelle Probleme beschweren. Dieser Makel wird hoffentlich durch eine zukünftige Version von KiCad gelöst.
- › Einzelheiten zur Plugin-Entwicklung für KiCad finden Sie unter GitHub. Hier wird auch der Pfad zu den Ordnern beschrieben, in denen die Plugins installiert werden müssen: https://github.com/KiCad/kicad-source-mirror/blob/master/Documentation/development/pcbnew-plugins.md#typical-plugin-structure--ppi_pi_struct

Differentieller Stromtastkopf für Oszilloskope 2.0

Ströme mit dem Oszilloskop messen

Von **Alfred Rosenkränzer**

Hier kommt ein differentieller Stromtastkopf für Oszilloskope mit Single-Ended-Ausgang und hoher Bandbreite. Die zweifache Standardverstärkung kann durch einen einzigen Widerstand erhöht werden!

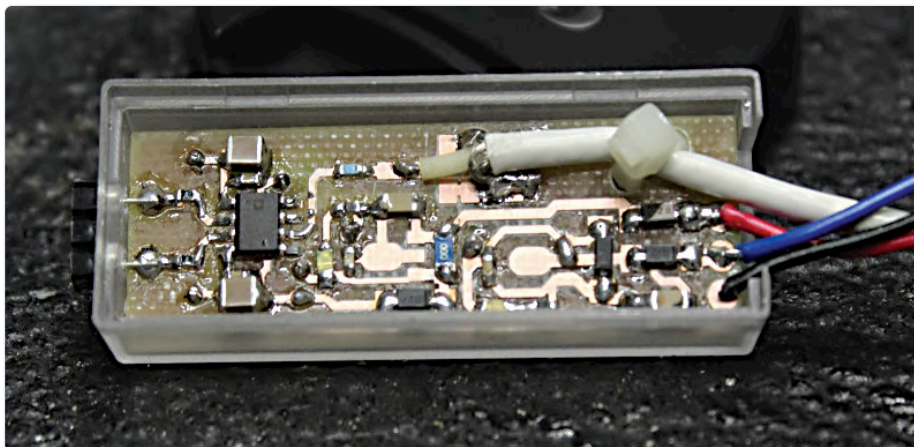


Bild 1. Prototyp der ersten Version des Tastkopfs von [1].

INFOS ZUM PROJEKT

Tags

Messen, Oszilloskop, Strom

Niveau

Einsteiger – Fortgeschrittene – Experte

Zeitbedarf

ca. 2 Stunden

Werkzeuge

Lötwerkzeug (SMD und bedrahtet), mechanisches Werkzeug

Kosten

ca. 30...40 €

Die „2.0“ im Titel indiziert: Das ist nicht der erste Tastkopf des Autors. **Bild 1** zeigt den Prototypen des schon vor vier Jahren in Elektor veröffentlichten [1] Tastkopfs von Alfred Rosenkränzer, der gegenüber dem hier beschriebenen Exemplar in Sachen Bandbreite aber weit hinterher hinkt. Damit ist schon die Motivation zur Entwicklung klar: Während die Version 1.0 nur mit maximal 130 kHz Bandbreite aufwarten konnte, legt der Tastkopf 2.0 in dieser Disziplin deutlich zu. Dieser Fortschritt wird mit einem reduzierten

Gleichtaktbereich der Eingänge bezahlt, doch lesen Sie selbst!

Stromtastkopf für Oszilloskope

Das Oszilloskop als unangefochtene Nr. 2 in der Messgerätehierarchie hat gegenüber dem Multimeter als Nr. 1 nicht nur den Vorteil, dass man damit die Dynamik eines Signals beurteilen kann, sondern auch einen entscheidenden Nachteil: Normalerweise kann ein Oszilloskop nicht direkt Ströme messen. Ein Stromtastkopf aber behebt diesen Nachteil.

Will man das dynamische Verhalten von Strömen in Schaltungen messen, steht man vor einem Problem: Klar könnte man irgendwo einen Shunt-Widerstand einfügen (oder aber einen vorhandenen Widerstand nutzen) und den Spannungsabfall daran mit einem normalen Tastkopf für Spannungen abgreifen. Der Horizontalverstärker selbst preiswerter Oszilloskope ist für solche Vorhaben mit mindestens 5 mV/Teilstrich empfindlich genug. Leider liegt dann aber die Masse des Oszilloskops an diesem Widerstand, was bezüglich Störungen nicht immer wünschenswert ist. Zudem besteht gerade bei einem Zweikanal-Oszilloskop die Gefahr, dass man Teile der Schaltung kurzschließt, wenn man mit dem zweiten

Kanal auch noch gleichzeitig eine Spannung messen will.

Die Lösung für das Problem ist ein kleiner differentieller Verstärker, der einfach potentialfrei eine kleine Spannungsdifferenz erfasst. Das ist nicht schwierig, denn Differenzverstärker haben einen sogenannten Gleichtaktbereich, innerhalb dessen sie ja nur Spannungsdifferenzen verstärken. Am Ausgang steht dann die auf den Massepegel des Oszilloskops bezogene Differenzspannung. Auf diese Weise lassen sich kleine Spannungsabfälle an Widerständen irgendwo in einer Schaltung sehr gut und (in Grenzen) unabhängig von einer gleichzeitigen Spannungsmessung erfassen.

Das Beste an diesem Konzept ist, dass es für diesen Zweck maßgeschneiderte ICs gibt. Es braucht noch ein paar zusätzliche Bauelemente und schon ist der Tastkopf fertig konzipiert.

Eigenschaften

Aufgrund der vom Hersteller ausgelobten großen Bandbreite von bis zu 6 MHz fiel die Wahl auf den Instrumentenverstärker AD8421 von Analog Devices. Dieses IC ist mit < 10 € nicht nur preiswert, sondern überzeugt mit geringem Rauschen, geringem Stromver-

brauch und hoher Gleichtakt-Unterdrückung. Die detaillierten Eigenschaften finden sich im Kasten „AD8421 – technische Daten“, noch mehr steht im Datenblatt des Herstellers [2]. Leider hat der AD8421 gegenüber dem in der ersten Version verwendeten IC AD8479 auch einen zentralen Nachteil: Während das IC von V 1.0 einen Gleichtaktbereich von immerhin $\pm 200 \dots 600$ V aufwies, müssen die Eingänge beim hier zum Einsatz kommenden AD8421 im Bereich seiner Versorgungsspannung bleiben. Bei einer Versorgung mit ± 15 V ergibt sich daher ein nutzbarer Gleichtaktbereich der Eingänge von etwa ± 12 V. Das scheint auf den ersten Blick wenig, aber leider mangelt es bei Halbleitern an eierlegenden Wollmilchsäuen, die alle idealen Eigenschaften kombinieren. Man nehme also die Version 1 von 2016, wenn man an Schaltungen mit hohen Versorgungsspannungen misst – und wähle die Version 2.0, wenn es um schnelle Signale in Schaltungen mit geringeren Betriebsspannungen geht. Man muss also lediglich zum zur Situation passenden Werkzeug greifen.

Tastkopf-Schaltung

Das Konzept von Alfred Rosenkränzer hat auch Elektor überzeugt, und so hat sich Ton Giesberts vom Elektor-Labor seiner Idee angenommen und sie leicht überarbeitet. Die resultierende Schaltung von **Bild 2** ist für eine Verstärkung mit dem Faktor 2 konfiguriert, die durch R5, einen einzelnen Widerstand, eingestellt wird. Die Formel für die Verstärkung lautet:

$$A = 1 + (9,9 \text{ k}\Omega / R5)$$

Lässt man R5 weg, ergibt sich laut Datenblatt eine einfache Verstärkung mit einer Bandbreite von 10 MHz, da R5 in diesem Fall ja quasi unendlich ist. Mit dem gewählten Wert von 10 k Ω für R5 erreicht man einen Verstärkungsfaktor von exakt 1,99, also einen Fehler von 0,5% gegenüber dem Faktor 2 – genauer als die verwendeten Widerstände. Die Bandbreite ist bei dieser Verstärkung > 6 MHz – gemessen lag sie sogar bei rund 8 MHz. Wenn Sie andere Verstärkungen bevorzugen: Selbst bei einer Verstärkung von 100 erreicht die Bandbreite des AD8421 noch 2 MHz!

Die Eingangsfilter (R1/C1 & R2/C2) bieten für kurze Überlastbedingungen einen einfachen Schutz gegen Überspannungen und Transienten. Zusätzlich korrigieren sie den leichten Verstärkungsanstieg des AD8421 in der Nähe der Grenzfrequenz, und linearisieren so den Frequenzgang des Tastkopfs. Obwohl der AD8421 über sehr robuste Eingänge verfügt, sollten längere Überspan-

nungen vermieden werden. Wie schon erwähnt ist der nutzbare Gleichtaktbereich der Eingangsspannungen durch die ± 15 -V-Versorgung auf etwas über ± 12 V begrenzt.

Der Tastkopf verfügt über eigene Spannungsregler sowie ein EMV-Filter mit C13, C14 und der Gleichtaktdrossel L1, um mögliche Störungen durch das für den Tastkopf verwendete Netzteil zu unterdrücken – wichtig, falls der Tastkopf durch ein Schaltnetzteil versorgt werden sollte. Er kann für gelegentliche Einsätze durch ein symmetrisches Labornetzteil oder besser durch ein kleines eigenes Netzteil versorgt werden. D3 und D4 sind übrigens nicht zum Gleichrichten einer Wechsellspannung gedacht, sondern schützen vor falscher Polarität!

LED1 und LED2 zeigen an, dass beide Versorgungsspannungen vorhanden sind und die Sonde aktiv ist. Ihre Vorwiderstände R7 und R8 sind so hochohmig, dass man von den LEDs nicht gleich geblendet wird. Die beiden 10-M Ω -Widerstände R3 und R4 bringen auch offene Eingänge auf Massepotential und sorgen so dafür, dass der IC-Ausgang dann auch 0 V liefert. Außerdem ergibt sich so eine definierte, hochohmige Eingangsimpedanz. Der 50- Ω -Widerstand R6 ist zwecks Anpassung an ein Koax-Kabel erforderlich, da der Ausgang von IC1 niederohmig ist.

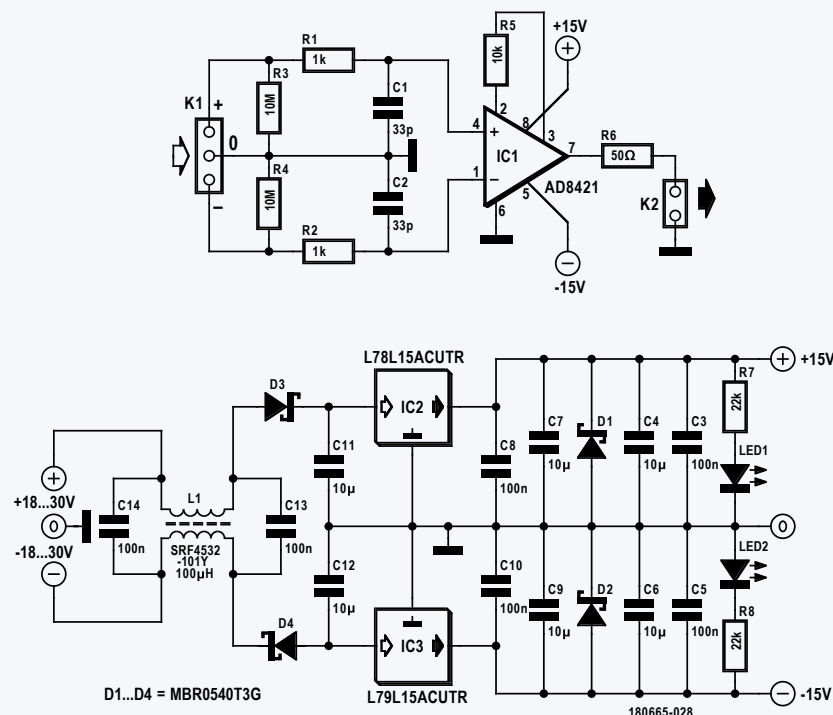


Bild 2. Komplette Schaltung des Tastkopfs 2.0. Man beachte den Aufwand bei der Genese der Versorgungsspannungen.

AD8421 – TECHNISCHE DATEN

Versorgung:

$\pm 2,5 \dots \pm 18$ V

Strombedarf:

max. 2,3 mA

Rauschen:

max. 3,2 nV/ $\sqrt{\text{Hz}}$ @ 1 kHz

max. 200 fA/ $\sqrt{\text{Hz}}$ @ 1 kHz

Bandbreite:

10 MHz @ Verstärkung = 1

2 MHz @ Verstärkung = 100

CMRR:

94 dB @ DC, Verstärkung = 1

80 dB @ 20 kHz, Verstärkung = 1

Slew Rate:

35 V/ μ s

Drift:

0,2 μ V/ $^{\circ}$ C

Eingangs-Offset-Spannung

1 ppm/ $^{\circ}$ C der Verstärkung @ Faktor 1

Eingangsschutz:

max. 40 V zwischen „+IN“ und „-VS“

max. 40 V zwischen „-IN“ und „+VS“

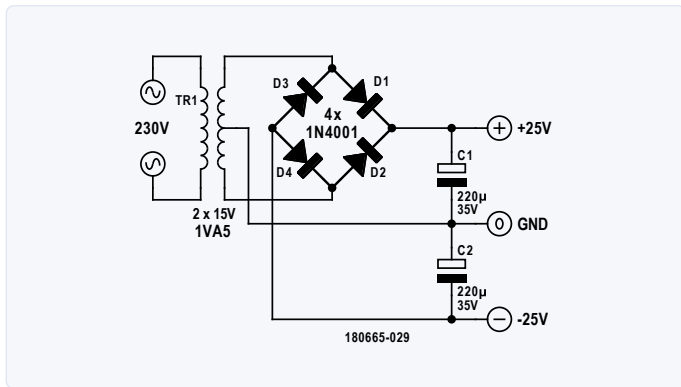


Bild 3. Schaltung eines einfachen Netzteils aus Trafo, Vollweg-Gleichrichtung und Siebelkos.

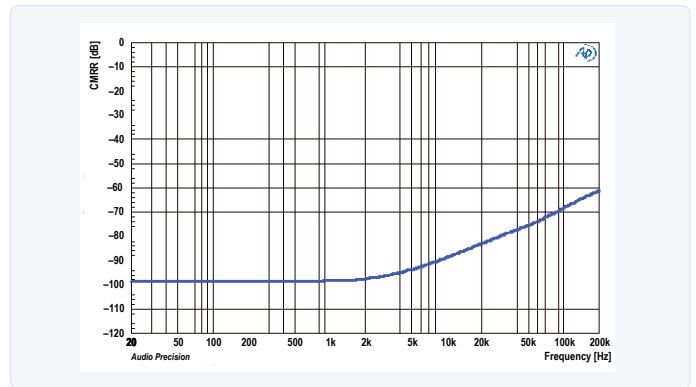


Bild 4. Die gemessene Gleichtaktunterdrückung in Abhängigkeit von der Frequenz.

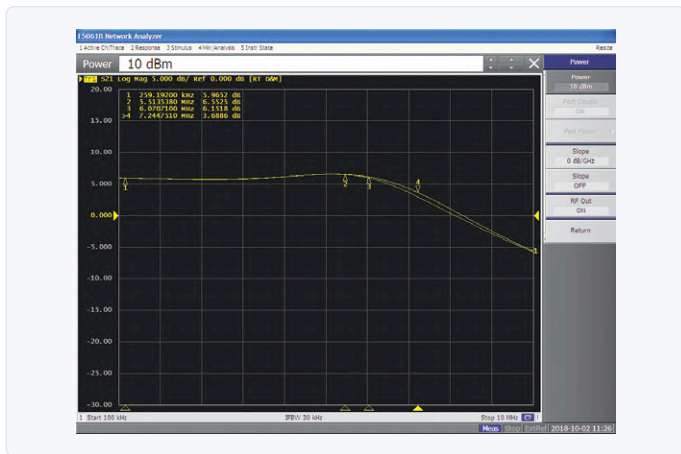


Bild 5. Der Frequenzgang des Tastkopfs reicht bis etwa 8 MHz.

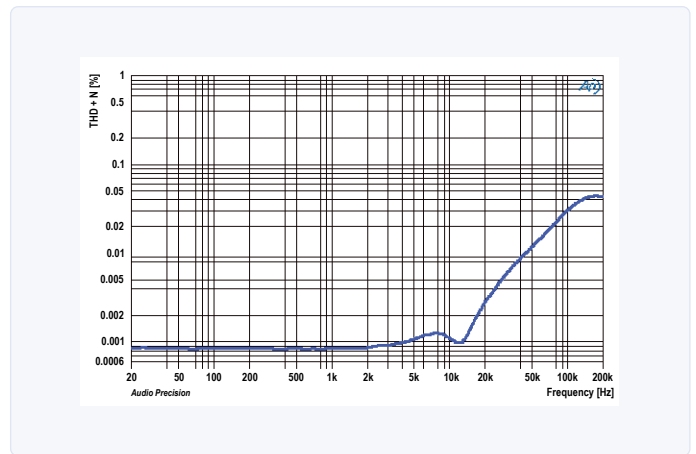


Bild 6. Die Verzerrungen steigen oberhalb von 10 kHz sprunghaft an, liegen aber auch noch bei 200 kHz auf sehr niedrigem Niveau.

Versorgung

Da nur sehr kleine Ströme fließen, reicht ein 1,5-VA-Trafo mit 2 x 12 V oder 2 x 15 V sekundär plus Vollweggleichrichter und zwei Siebkondensatoren von je 220 µF / 35 V wie in **Bild 3** völlig aus. Messungen ergaben bei der Versorgung mit so einem kleinen Netzteil gegenüber einem Labornetzgerät keine Unterschiede. Die Welligkeit der Versorgungsspannungen ist durch die geringe Last mit etwa 200 mV_{SS} recht gering.

Die verwendeten 15-V-Spannungsregler vertragen Eingangsspannungen von 17,5...35 V. Durch die beiden Schottky-Dioden D3 und D4 kann der Tastkopf daher mit Spannungen ab ±18 V versorgt werden. Ein kleiner 12-V-Trafo für TR1 kann hierfür schon reichen, da kleine Trafos hochohmig sind und bei kleiner Last deutlich erhöhte Leerlaufspannungen liefern. Dies führt auch abzüglich des Spannungsverlusts von ±0,7 V durch D1...D4 oft noch bis zu ±20 V am Netzteilaustritt.

Man sollte das aber unbedingt nachmessen. Ein kleiner 18-V-Trafo hingegen kann u.U. schnell über ±35 V am Ausgang liefern, was man tunlichst vermeiden sollte. Bei einem kleinen 15-V-Trafo ist mit etwa ±26 V am Ausgang zu rechnen.

Es wurden auch Tests zur Ermittlung des Verhaltens bei reduzierter Versorgungsspannung durchgeführt. Gemessen wurde ein 1-kHz-Signal mit einer Amplitude von 3 V_{SS}, was am Ausgang des Tastkopfs durch seine Verstärkung 6 V_{SS} ergibt. Das kleine Netzteil wurde an einen Regeltrafo angeschlossen und seine Eingangsspannung immer weiter verringert. Solange die vom Netzteil gelieferte Versorgungsspannung über ±12 V blieb, waren keine Veränderungen beobachtbar, aber darunter begann das Signal am Ausgang des Tastkopfs zu clippen. Kein Wunder, denn an D3/D4 und den Spannungsreglern fallen über 2 V ab und für den Ausgang von IC1 bleibt dann nicht mehr viel Reserve. Dass sich

selbst bei knapp über ±12 V keine Störungen zeigten, obwohl doch beide Spannungsregler nicht mehr regelten, belegt die hervorragende Unterdrückung der Effekte der Versorgungsspannungen beim AD8421 (≥100 dB @ 100 Hz).

Messungen

Bei einem 1-kHz-Signal mit 3 V_{SS} an beiden Eingängen beträgt die Gleichtaktunterdrückung > 98 dB. Sie nimmt ab 2 kHz stetig bis auf 61 dB bei 200 kHz ab (siehe **Bild 4**). Dies ist besser als erwartet, obwohl die Bauteile des Eingangsfilters aus R1/C2 und R2/C2 trotz enger Toleranz nicht völlig gleich sind. Eine Single-Ended-Messung an einem Funktionsgenerator mit einem Eingang (der andere Eingang lag auf Masse) brachte eine reale Bandbreite von etwa 8 MHz (bezogen auf 1 kHz). Bei niedrig eingestellten Verstärkungen durch R5 ergibt sich eine geringe Anhebung der Verstärkung nahe der Grenzfrequenz mit



Bild 7. Koax-Kabel RG316 mit abisoliertem Ende und aufgedrimptem Stecker auf dem anderen Ende.

dem Höchstwert bei 7 MHz (siehe **Bild 5**). Die Eingangsfilter haben eine Grenzfrequenz von 4,8 MHz (ohne Berücksichtigung der Quellimpedanz), was diese Anhebung kompensiert. Diese Anhebung ist dafür verantwortlich, dass die Bandbreite des Tastkopfs höher ausfällt als erwartet. Bei 8 MHz beträgt die Phasenverschiebung des Ausgangssignals jedoch mehr als 180°. Sinuswellen im MHz-Bereich sehen eher dreieckförmig aus. Die Verzerrungen nehmen bei Signalen > 10 kHz stetig zu (siehe **Bild 6**).

Kabel

Zum Anschluss des Ausgangs kann ein koaxiales RG316-Kabel verwendet werden, das als Meterware erhältlich ist. Der Anschluss eines BNC-Steckers am anderen Ende erfordert jedoch das richtige Crimp-Werkzeug. Falls nicht verfügbar, kann man einfach ein komplettes BNC-Koax-Kabel kaufen, einen Stecker an einem Ende abschneiden, dort die Isolation entsprechend entfernen (siehe **Bild 7**) und es direkt an die Platine anlöten. Kauft man ein längeres BNC-Kabel und teilt es in der Mitte, hat man gleich zwei Kabel für zwei Tastköpfe.

Zum direkten Verlöten des RG316-Kabels hat die Platine in der Mitte große Pads (siehe **Bild 8**), an die sowohl der Mittelleiter als auch der Schirm angelötet werden können. In die drei Pads rechts oben können Drähte für die Stromversorgung eingelötet werden. Alle fünf Anschlüsse haben Löcher, in die auch 1,3-mm-Lötstifte passen würden. Verwendet man Lötstifte, müssen sie wohl etwas gekürzt werden, damit das Board noch in das empfohlene Gehäuse passt. Mit einem kleinen Kabelbinder durch die 2,5-mm-Löcher ganz rechts kann man alle die Sonde verlassenden Leitungen fixieren (**Bild 9**). Auch Heißkleber kann hier gute Dienste leisten. Lesen Sie aber

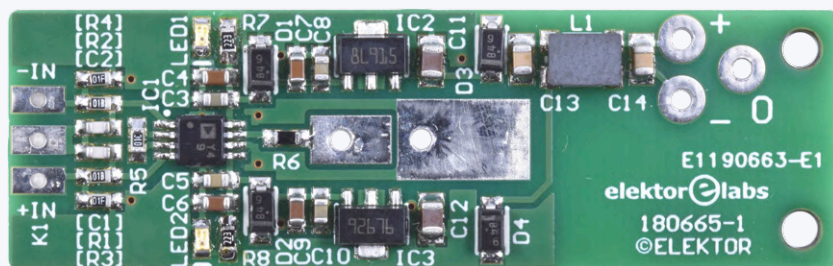


Bild 8. Die voll bestückte Platine des Prototypen. Aufgrund der SMDs im Format 0603 ist eine ruhige Hand erforderlich.

vor dem Kabelanschluss zuerst den folgenden Abschnitt mit den Aufbauhinweisen.

Als „Buchse“ für die Messleitungen lötet man am einfachsten ein Teilstück aus 3 Pins eines SIL-Sockels (wegen der besseren Qualität mit „gedrehten“ Kontakten) flach auf die drei Pads links auf die Platine (Bild 8). In Bild 9 ist zu sehen, wie das fertig bestückt aussieht.

Aufbauhinweise

In der Stückliste ist ein geeignetes Gehäuse angegeben. Zunächst bringt man links ein

passendes, rechteckiges Langloch für die dreipolige SIL-Steckverbindung am Gehäuse an. Nun legt man die fertig bestückte und geprüfte Platine ins Gehäuse, führt die dreipolige SIL-Buchse links durch das Langloch auf die Platine und lötet sie fest.

Nun kommt das Koax-Kabel an die Reihe. Unser Vorschlag: Verwenden Sie RG316-Kabel. Der BNC-Stecker muss für RG316 geeignet sein! Es gibt viele BNC-Varianten für verschiedene Kabel. Beachten Sie die Stückliste. Das beim Prototyp verwendete Kabel

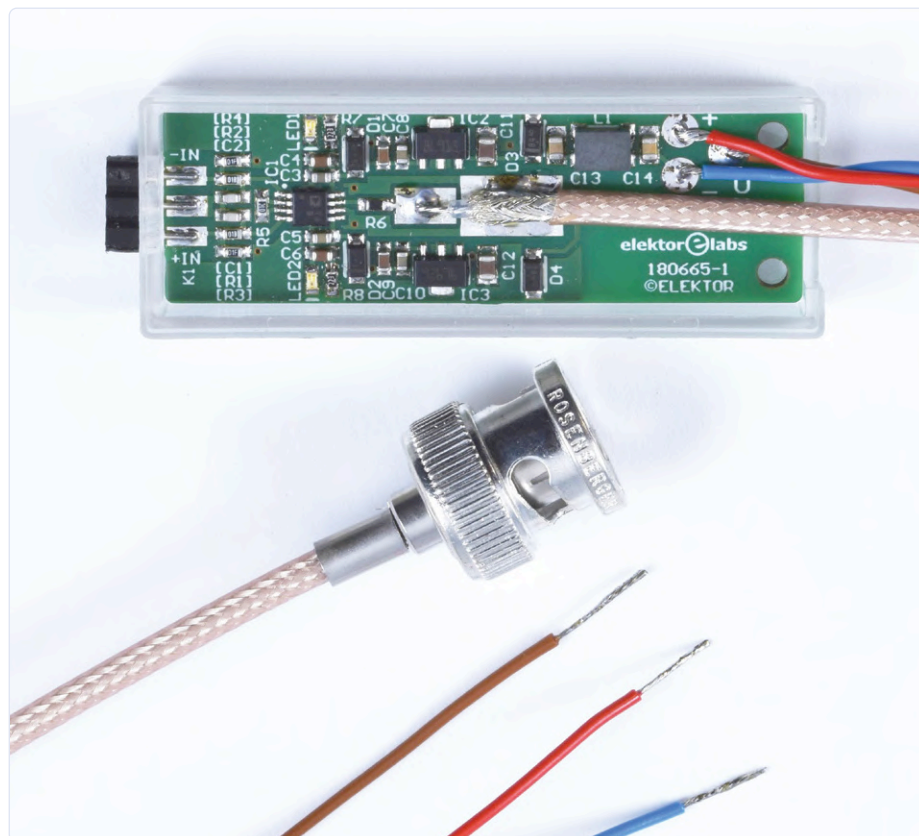


Bild 9. Der Prototyp eingebaut im Gehäuse mit angelöteten Kabeln und SIL-Buchse.



STÜCKLISTE

Widerstände:

(alle 1%, 100 mW, SMD 0603)

R1,R2 = 1 k

R3,R4 = 10 M

R5 = 10 k

R6 = 50 Ω

R7,R8 = 22 k

Kondensatoren:

C1,C2 = 33 p / 50 V, 1%, C0G/NP0, SMD 0603

C3,C5,C8,C10 = 100 n / 50 V, 10%, X7R,
SMD 0603

C4,C6,C7,C9 = 10 μ / 25 V, 20%, X5R,
SMD 0603

C11,C12 = 10 μ / 35 V, 10%, X5R, SMD 0805

C13,C14 = 100 n / 100 V, 10%, X7R, SMD 0805

Spule:

L1 = 100 μ H @ 100 kHz, Gleichtaktrossel,
200 mA, SMD, z.B. SRF4532-101Y, Bourns

Halbleiter:

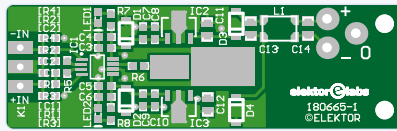
IC1 = AD8421ARMZ, MSOP 8

IC2 = L78L15ACUTR, SOT-89

IC3 = L79L15ACUTR, SOT-89

LED1,LED2 = LED, rot, SMD 0603

D1..D4 = MBR0540T3G, SOD-123



Die Bestückungsseite der im Elektor-Labor konzipierten Platine.

Außerdem:

K1 = dreipol. SIL-Buchse, stehend, RM 1/10"

Dreipol. SIL-Buchse als Stecker für Messkabel*

Koaxialkabel RG316, 50 Ω , $\varnothing$ 2,9 mm, 1 m,

z.B. Conrad 606450*

BNC-Crimp-Stecker für RG316, 50 Ω ,

z.B. Conrad 1490723*

Gehäuse USB1KL, 56x20x12 mm,

z.B. Conrad 531276*

Alternatives Gehäuse: TC-USB1 KL203,

56x20x12 mm

Platine # 180665-1 v1.0*

* siehe Text

hat einen Außendurchmesser von 2,9 mm. Falls Sie noch keine Erfahrung mit solchen Steckern haben, sollten Sie sich auf der Hersteller-Webseite informieren, wie man das Kabel korrekt abisoliert, um den BNC-Stecker anzubringen.

Um das Koax-Kabel an die Pads auf der Platine zu löten, müssen zuerst 12 mm der Ummantelung abisoliert werden. Dann kürzt man das Abschirmgeflecht auf 6 mm. Nun entfernt man 4 mm der Isolation des Mittelleiters. Die Litze des Mittelleiters ist dann vom Abschirmgeflecht 2 mm entfernt. Zuerst den Mittelleiter anlöten, dann die Abschirmung. Bei Letzterer sollte man zügig arbeiten. Bild 9 zeigt, wie das fertig angelötet aussehen sollte. Nun kommen die drei Litzen für die Stromversorgung an die Reihe. Was noch fehlt sind die Messleitungen. Hierzu lötet man drei dünne Litzen an einen dreipoligen SIL-Stecker. Wir haben am Prototypen testweise zwei dünne Kupferlackdrähte an eine SIL-Buchse gelötet (Bild 10). Jetzt kann der Deckel drauf und der Tastkopf ist einsatzbereit.

Messen

Die Stromversorgung des Tastkopfs sollte von der zu untersuchenden Schaltung galvanisch getrennt sein – daher am besten ein eigenes kleines Netzteil wie das von Bild 3 verwenden. Beim Messen gilt zwingend: Verbinden Sie die Masse des Eingangs mit der Masse der Schaltung! Die Messpunkte dürfen gegenüber Masse eine Gleichtaktspannung von bis zu ± 12 V aufweisen – aber nicht mehr. Zur Steckverbindung für die Messstrippen: Die mechanische Lebensdauer gedrehter SIL-Buchsen beträgt in der Regel ein paar hundert Steckvorgänge. Sie sollten daher jahrelang funktionieren. Falls erforderlich, kann die auf die Platine gelötete Buchse leicht ersetzt werden, da ihre Pins ja nicht durch Bohrungen gesteckt, sondern flach auf die Pads gelötet sind.

Statt den mittleren Pin dieser Buchse als Masseanschluss zu nutzen, kann man auch

MESSUNGEN AM PROTOTYPEN

Versorgungsspannung mit Netzteil nach Bild 3 (12-V-Trafo): $\pm 18,9$ V.

Die Stromaufnahme lag pro Zweig bei etwa 5...6 mA.

Der Löwenanteil entfällt hierbei auf die Spannungsregler und die LEDs.

Maximalwerte bei einem 1-kHz-Signal von 4,8 V_{SS}:

U_{out} : 9,52 V @ Last = 100 k Ω

THD+N: 0,1 %,

Typische Werte bei einem 1-kHz-Signal von 3 V_{SS}:

U_{out} : 5,97 V @ Last = Oszilloskop-Eingang

THD+N 0,00025 % @ B = 22 kHz

0,00083 % @ B > 500 kHz

Gleichtaktunterdrückung bei 3 V_{SS} an beiden Eingängen

CMRR: >98 dB @ 20...1.000 Hz

>60 dB @ 200 kHz

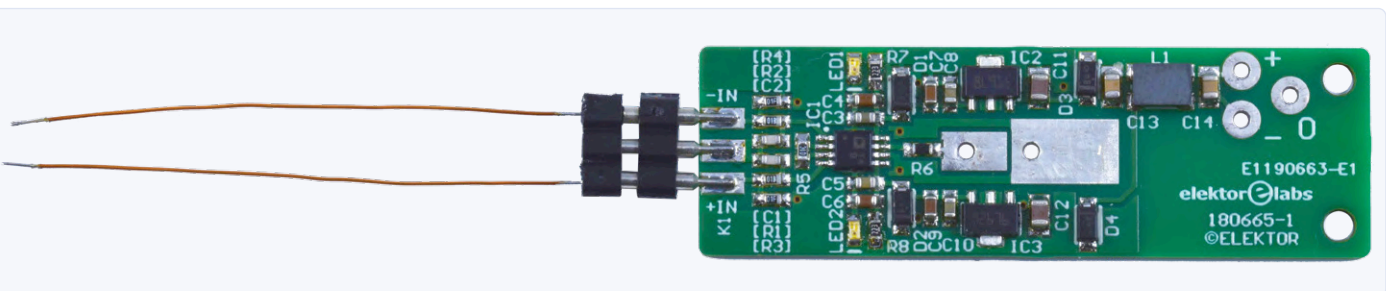


Bild 10. Testweise wurden als „Messstrippen“ zwei dünne Kupferlackdrähte an eine dreipolige SIL-Buchse gelötet und eingesteckt. Die Masse muss dann anderweitig angeschlossen werden.

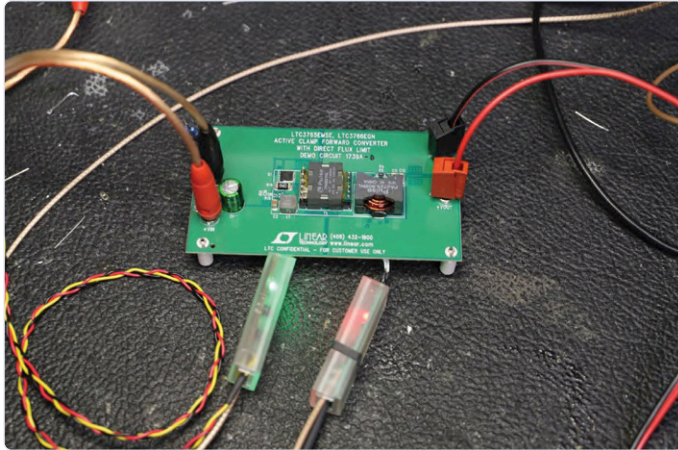


Bild 11. Alfred Rosenkränzer hat hier gleich zwei Tastköpfe an einer Schaltregler-Platine im Einsatz.

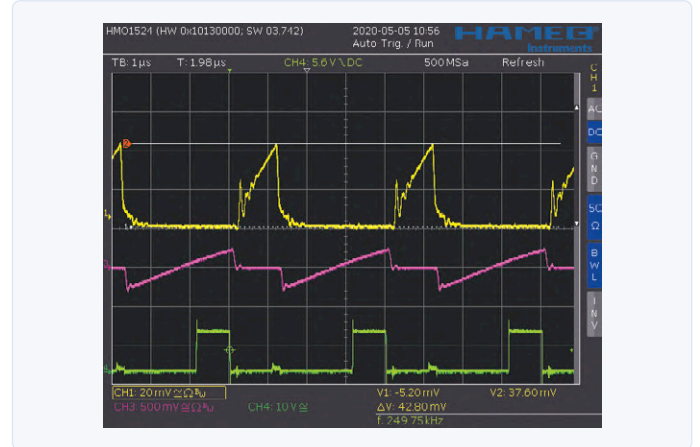


Bild 12. Der Tastkopf 2.0 kann auch schnelle Stromänderungen erfassen.

einfach das Pad „0“ der Platine mit der Masse der zu messenden Schaltung verbinden. Dank der hohen Gleichtaktunterdrückung des Tastkopfs ist es unwahrscheinlich, dass hierdurch Störungen, Interferenzen oder Rauschen entstehen.

Bei der Verwendung mehrerer, vom selben Netzteil gespeisten Tastköpfe darf man natürlich nur eine Masseverbindung zur zu messenden Schaltung herstellen – sonst

entsteht eine unschöne Masseschleife. Außerdem wichtig: Ist die Versorgungsspannung der untersuchten Schaltung $> 12\text{ V}$, sollte man vor dem Anlegen der Messleitungen an einen Shunt-Widerstand zunächst mit einem Multimeter prüfen, ob die Spannung an beiden Punkten innerhalb des Gleichtaktbereichs von $\pm 12\text{ V}$ bleibt.

Bild 11 zeigt den Einsatz zweier Tastköpfe an einer kleinen Schaltregler-Platine. **Bild 12**

beweist, dass sich mit diesem Tastkopf wirklich schnelle Stromänderungen erfassen lassen.

Last not least: Wie immer können Sie die Layout-Dateien der zugehörigen Platine kostenlos von der Elektor-Webseite zu diesem Artikel [3] herunterladen. 

180665-01

Sie haben Fragen oder Kommentare?

Gerne können Sie sich an den Autor wenden unter der E-Mail-Adresse: alfred_rosenkraenzer@gmx.de.



PASSENDE PRODUKTE

- **PeakTech Stromzange 4350**
www.elektor.de/peaktech-4350-clamp-meter
- **Potentialfreie Strommessung für Oszilloskope**
www.elektor.de/potentialfreie-strommessung-fur-oszilloskope
- **OWON SDS1102 2-Kanal-DSO (100 MHz)**
www.elektor.de/owon-sds1102-2-ch-digital-oscilloscope-100-mhz

Ein Beitrag von

Idee und Schaltung: **Alfred Rosenkränzer**
Überarbeitung und Text: **Ton Giesberts**

Schaltpläne: **Patrick Wielders**
Redaktion: **Dr. Thomas Scherer**

Layout: **Giel Dols**

WEBLINKS

- [1] „Stromtastkopf für Oszilloskope“, A. Rosenkränzer, *Elektor* 7-8/2016: www.elektormagazine.de/magazine/elektor-201607/29062
- [2] **Datenblatt AD8421**: www.analog.com/media/en/technical-documentation/data-sheets/AD8421.pdf
- [3] **Elektor-Webseite zu diesem Artikel**: www.elektormagazine.de/180665-01
- [4] **Projekt auf Labs**: www.elektormagazine.com/labs/differential-current-probe-180665

Die Hewlett-Packard Interface-Loop

Verbinde die Welt!



Blue Globe by Torty3, Wikimedia Commons under Creative Commons Attribution-Share Alike 3.0 Unported (https://commons.wikimedia.org/wiki/File:Blue_globe_icon.svg).

Von **Dipl.-Inf. Karl-Ludwig Butte**

Klein, portabel, batteriebetrieben und – ganz wichtig – alles über ein Netzwerk miteinander verbunden: So stellen wir uns heute im Zeitalter des Internets der Dinge elektronische Systeme vor, die vollautomatisch unsere Umwelt mit ihren Sensoren überwachen und regelnd über ihre Aktuatoren eingreifen, wenn es die Situation erfordert. Aber gibt es das alles wirklich erst, nachdem Mark Weiser seine Vision vom „allgegenwärtigen Computing“ in einem viel beachteten Artikel in der September 1991 Ausgabe von Scientific American [1] dargelegt hatte? Nein, und wenn Sie jetzt neugierig geworden sind, dann folgen Sie mir ins Jahr 1981; in die Welt der Hewlett-Packard Interface-Loop.

Das faszinierendste Feature des Hewlett-Packard Taschenrechners HP-41C, der am 1. Juli 1979 [2] der Weltöffentlichkeit vorgestellt wurde, waren die vier Steckplätze für Peripheriegeräte, wie z.B. ein Barcode-Lesestift, Thermodrucker, Magnetkartenleser und RAM- und ROM-Erweiterungsmodule. Das hatte es so noch nicht gegeben! Die Anwender kamen sehr schnell auf den Geschmack und wünschten sich weit darüber hinaus reichende Möglichkeiten, den HP-41C mit der elektronischen Außenwelt zu verbinden. Hewlett-Packard hatte bereits einschlägige Erfahrungen bei der Verbindung und automatischen Steuerung ihrer Messgeräte mit dem Hewlett-Packard Interface-Bus (HP-IB) gesammelt. Der HP-IB wurde zum Industrie-Standard IEEE-488. Doch die dicken, mindestens 18 Adern umfassenden Kabel und die klobigen 24-poligen Amphenol-Stecker des HP-IB, der einen 8-bit-Parallelbus realisierte, standen in krassem Gegensatz zu den kleinen batteriebetriebenen Geräten der HP-41C-Familie. Um diese beiden so unterschiedlichen Welten zusammenzubringen, serialisierten die HP-Ingenieure den Datenverkehr. Statt acht Bits parallel zu übertragen, wurden die Bits eins nach dem anderen auf die Reise geschickt. Der Hardware-Handshake wurde zusätzlich durch ein Software-Handshaking ersetzt. Mit diesen Maßnahmen konnte das Kabel auf das absolute Minimum von nur zwei Adern reduziert werden. Aber es fehlte noch ein Detail, um das neue Verbindungssystem so zuverlässig und robust zu machen, wie es die Kunden von der sprichwörtlichen Hewlett-Packard-Qualität gewohnt waren. Statt

einer Bus-Topologie entschieden sich die HP-Ingenieure für eine Loop-Topologie. Das Kabel ging vom Loop-Controller (z.B. dem HP-41C) zum ersten Peripheriegerät. Von dort wurde das Kabel zum zweiten Peripheriegerät gezogen und so weiter, bis es schließlich vom letzten Peripheriegerät wieder zum Loop-Controller zurückgeführt wurde. Somit konnte der Loop-Controller selbst per Software kontrollieren, ob die ausgesendeten Bits den wieder empfangenen Bits zu 100% entsprachen und gegebenenfalls Übertragungsfehler sofort korrigieren.

Am 14.12.1981 [3] war es dann soweit: HP stellte die Hewlett-Packard Interface-Loop (kurz HP-IL) in Form des HP-IL-Moduls 82160A (**Bild 1**) für den HP-41C vor. Sie wurde einfach in einen der vier zuvor erwähnten Steckplätze für Peripheriegeräte des HP-41C eingesteckt. Zusammen mit dem HP-IL-Modul wurden das IL-Kassettenlaufwerk HP-82161A und der IL-Thermodrucker HP-82162A (**Bild 2**) eingeführt. Das übertraf die kühnsten Erwartungen der Kunden. Die Konkurrenz dagegen wurde mit diesem ausgefuchsten System in Angst und Schrecken versetzt, denn niemand hatte etwas Vergleichbares zu bieten. Im Laufe der Zeit wuchs nicht nur die Liste der Peripheriegeräte, sondern auch die Liste der Computer, für die ein HP-IL-Modul verfügbar war. So unterstützten die BASIC-Taschen-Computer HP-71B [4] und HP-75C / D die Interface-Loop und konnten so dieselben Peripheriegeräte wie der HP-41C nutzen. Für die Desktopgeräte der Serie 80 (HP-85, 86, 87) sowie den HP-110 und den HP-150 gab es entsprechende Schnittstellenkarten. 1986 kam dann mit dem HP-82973A HP-IL-Interface eine ISA-Einsteckkarte für IBM-PC und kompatible Rechner auf den Markt und das HP-82166C HP-IL Development Kit ermöglichte es anderen Firmen, HP-IL-kompatible Geräte zu entwickeln.

Peripheriegeräte von HP

Als Kassettenlaufwerk diente nicht irgendein Standard-Musik-Kassettenrekorder wie bei vielen Heimcomputer-Herstellern. Nein, HP hatte ein Präzisions-Laufwerk konstruiert, welches von der Funktion, der Größe und des Gehäuse-Designs perfekt zum HP-41C-System passte (**Bild 3**). Die Kassetten ähnelten denen von Micro-Diktiergeräten. **Bild 4** zeigt eine solche Kassette im Vergleich zu einer normalen Compact-Kassette. Der Platz links neben dem Kassettenschacht des Laufwerks bot Aufbewahrungsplatz für zwei Kassetten (Bild 3). Auf einer Kassette hatten 131.072 Bytes Platz, das entsprach in etwa der Kapazität einer 5-1/4-Zoll-Diskette beim Commodore PET 2001 [5], die 176.640 Bytes speichern konnte. Der Thermodrucker HP-82162A war äußerlich kaum von seinem älteren Bruder HP-82143A zu unterscheiden, der direkt an einen der vier I/O-Ports des HP-41C angeschlossen werden konnte. Aber der IL-Thermodrucker brachte ein von den Anwendern lang gewünschtes Feature mit: Mit ihm konnte man HP-41C-Barcodes drucken! Mit dem HP-82153A Barcodeleser, der ebenfalls an einem der vier I/O-Ports des HP-41C direkt angeschlossen wurde, konnte man Programme und Daten aus Büchern und Zeitschriften in den HP-41C sehr viel schneller übernehmen, als sie mühsam abzutippen. Aber wie schon gesagt, diese beiden Geräte waren nur der Anfang. **Tabelle 1** gibt einen Überblick über die von Hewlett-Packard angebotenen HP-IL kompatiblen Geräte. Es gab wirklich alles, was das Herz eines HP-Fans höherschlagen ließ: Von Speichergeräten über Drucker bis zu den verschiedensten Schnittstellen und einen Akustikkoppler! Der Clou aber waren die Messgeräte. Mit einem System, bestehend aus dem HP-3468 Multimeter, einem Time-Modul HP-82182A und einem HP-41C, ließen sich Messreihen über



Bild 1. Das HP-Interface-Loop-Modul HP-82160A.



Bild 2. Ein HP-IL-System bestehend aus HP-41C, Kassettenlaufwerk und Thermodrucker.



Bild 3. Kassettenlaufwerk HP-82161A.



Bild 4. Eine Kassette für das Kassettenlaufwerk im Vergleich zu einer normalen Musik-Kassette.

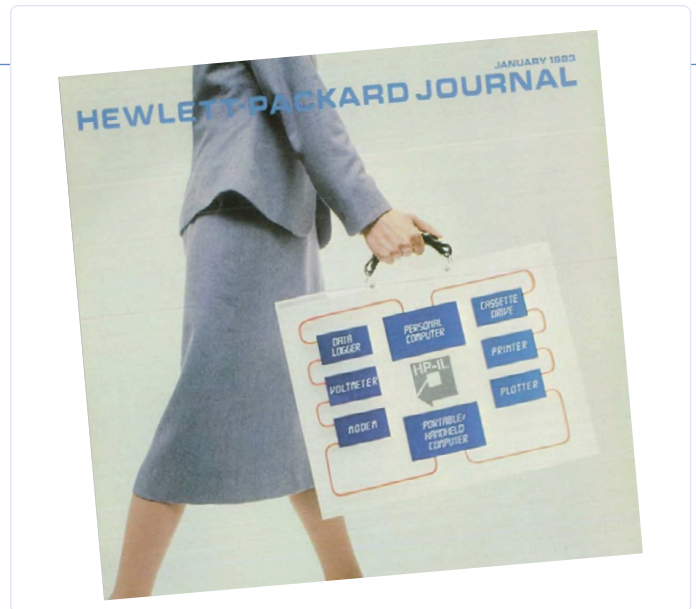


Bild 5. Titelbild des Hewlett-Packard Journals vom Januar 1983.

Tabelle 1. Übersicht über die von HP angebotenen HP-IL kompatiblen Peripherie-Geräte [6].

Gerät	Bezeichnung
HP 9114A/B	Diskettenlaufwerk
HP 82161A	Kassettenlaufwerk
HP 82162A	Thermo-Drucker (HP-IL Version des HP 82143A Druckers für den HP-41C)
HP 82163A	Video-Schnittstelle
HP 92198A	80-Zeilen-Videoschnittstelle (Mountain Computer)
HP 82168A, 92205M	Akustikkoppler (Modems)
HP 82905A/B	Drucker
HP 2225B	ThinkJet Drucker
HP 7470A	Graphischer Plotter (Opt. 003 HP-IL Schnittstelle)
HP 2671A/G	Alphanumerischer, graphischer Thermo-Drucker
HP 1630 und HP1631	Logikanalysatoren (kann Schleifen-Controller sein)
HP 3421A	Datenerfassungs-/Steuereinheit
HP 3468A/B	Digital-Multimeter (HP-IL-Version des HP 3478A mit GPIB)
HP 5384A und HP 5385A	Frequenzzähler, mit Option 003 (HP-IL)
HP 82164A	HP-IL-/RS-232C-Schnittstelle (seriell)
HP 82165A	HP-IL-/GPIO-Schnittstelle (generischer Parallelausgang)
HP 82166A	HP-IL-Konverter (kleinere Version der GPIO-Schnittstelle für Embedded-Anwendungen)
HP 82166C	HP-IL Converter Prototyping Kit
HP 82169A	HP-IL-/HPIB-Schnittstelle (bidirektional)
HP 82985A	HP-IL/NTSC-Video (monochrom)

einen längeren Zeitraum vollautomatisch erstellen. Fügt man noch Drucker und/oder Kassettenlaufwerk in die Interface-Loop dazu, konnten die Messwerte gleich protokolliert und gespeichert werden. Und mit dem HP-82168A Akustikkoppler ließen sich die Daten per Telefon an einen Zentralrechner zur Weiterverarbeitung übertragen. Aus rein funktionaler Sicht unterscheiden sich viele heutige IoT-Anwendungen in keinsten Weise von diesem Setup mit der HP-Interface-Loop.

Funktionsweise

Die Hewlett-Packard Interface-Loop verband die angeschlossenen Geräte in einem Ring, vergleichbar mit der Token-Ring-Technologie, welche 1974 an der Universität von Cambridge (UK) entwickelt und von IBM Mitte der 1980er Jahre weiterentwickelt und vertrieben wurde. Der Ring-Character dieses Netzwerks ist sehr schön auf dem Cover des Hewlett-Packard Journals vom Januar 1983 zu sehen (**Bild 5**), in dem die Interface-Loop ausführlich vorgestellt wurde. Jedes HP-IL-System wird von einem sogenannten Loop-Controller gesteuert und kann maximal dreißig Geräte miteinander vernetzen. Jedem Gerät wird vom Loop-Controller automatisch eine Geräte-Adresse zugeteilt. Das funktioniert analog zur automatischen IP-Adressenvergabe mittels DHCP in einem Ethernet-Netzwerk. Der Loop-Controller sorgt für einen strikten Protokoll-Ablauf, wofür ihm eine ganze Reihe von Kommando-Gruppen zur Verfügung stehen. Mit den Reset-Messages für die Loop und für einzelne Geräte sorgt der Loop-Controller für klare Ausgangsbedingungen. Mit einer weiteren Message-Gruppe bestimmt der Loop-Controller den aktiven Listener oder Talker – er spricht also direkt ein bestimmtes Gerät an bzw. erteilt ihm das Wort. Status-Abfragen, Datenaustausch-Nachrichten und weitere Messages, die der Verwaltung und Synchronisierung dienen, komplettieren die Kommando-Gruppen. Wie jedes einzelne Gerät auf eine bestimmte Nachricht reagiert, kann ganz unterschiedlich sein und wird von dem Entwickler des Peripherie-Gerätes bestimmt. Der Thermodrucker wird auf eine Device-Clear-Message (DCL) seinen internen Druckbuffer leeren, den Druckkopf an den linken Rand positionieren und das Papier eine Zeile vorschieben. Die Video-Schnittstelle hingegen wird den internen Video-Speicher löschen und den Cursor in die linke obere



Bild 6. HP-IL Interface Kit HP-82166C.



Bild 7. Die Spezial-Bauteile des HP-IL Interface Kits.

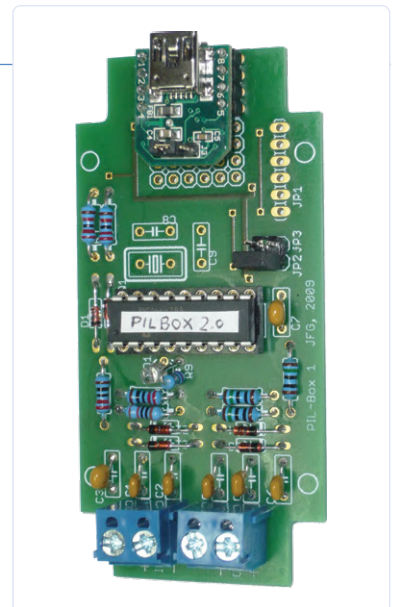


Bild 8. Das PIL-Box-Interface (Abdruck mit freundlicher Genehmigung von J.-F. Garnier, der das volle Urheberrecht behält).

Ecke des Bildschirms positionieren. Aber wie integrierte man ein HP-IL-Interface in die eigene Geräte-Entwicklung? Dafür gab es das HP-IL Interface Kit HP-82166C.

Das HP-IL Interface Kit

HP ermutigte die Hardware-Entwickler-Gemeinde durchaus, Geräte mit HP-IL-Schnittstelle zu entwickeln und veröffentlichte detaillierte technische Beschreibungen dafür. Darüber hinaus wurde ein Interface-Kit (**Bild 6**) auf den Markt gebracht. **Bild 7** zeigt die darin enthaltenen Spezial-Bausteine. Neben drei ASICs beinhaltete das Kit zwei Stecker-Module für die Spezial-Stecker der Loop-Verbindungskabel, kleine Trenntrafos im DIL-Gehäuse zur galvanischen Trennung der Geräte und ein Einsteck-Modul für den HP-41C für das Debugging der Software.

HP-IL lebt!

Wann genau Hewlett-Packard die Interface-Loop abgekündigt hat, ließ sich nicht mehr genau ermitteln. Der Nachfolger des HP-41C, der 1988 auf den Markt gekommene HP-42S, besaß jedenfalls weder I/O-Ports geschweige denn ein HP-IL-Interface. Aber die Interface-Loop hatte mit ihrer Flexibilität, Vielseitigkeit und technischen Überlegenheit schnell die Herzen der Entwickler weltweit

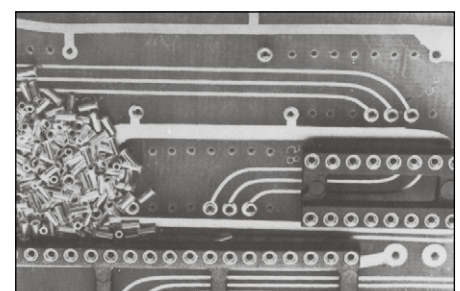
erobert und diese Begeisterung hält auch heute noch an. Es gibt etliche Community-Projekte, die die Interface-Loop mit heutiger Technik kombinieren. Eines der bekanntesten Projekte ist die PIL-Box von Jean-Francois Garnier [7]. **Bild 8** zeigt die Platine der PIL-Box, die ein HP-IL-zu-USB-Interface realisiert. Was das für neue, ungeahnte Möglichkeiten eröffnet, kann sich sicherlich jeder hier in der Elektor-Community leicht vorstellen.

Wie so oft in der damaligen Zeit war Hewlett-Packard mit der HP-Interface-Loop dem State-of-the-Art weit voraus. Doch durch die besondere Sorgfalt bei der Entwicklung, Qualität in der Produktion und ausführliche Dokumentation können wir auch heute noch, lange nachdem diese Produkte eingestellt wurden, viel Spaß mit dieser Technik haben. ◀

200329-01

WEBLINKS

- [1] Weiser, Mark: „The Computer for the 21st Century“. Scientific American, September 1991: www.lri.fr/~mbl/Stanford/CS477/papers/Weiser-SciAm.pdf
- [2] Hewlett-Packard Taschenrechner HP-41C: www.finseth.com/hpdata/hp41c.php
- [3] HP-Taschenrechner mit Datum der Einführung: www.vcalc.net/hp-date.htm
- [4] Butte, Karl-Ludwig: „Der Zahlenfresser HP-71B (1984)“, Elektor 9/2014 S. 82: www.elektormagazine.de/magazine/elektor-201409/27018
- [5] Butte, Karl-Ludwig: „Alles Gute zum 40., PET!“, Elektor 5/2017 S. 90: www.elektormagazine.de/magazine/elektor-201705/40281
- [6] Quelle Wikipedia: https://de.wikipedia.org/wiki/Hewlett-Packard_Interface_Loop
- [7] HP-IL Resource Page: www.jeffcalc.hp41.eu/hpil/



Durchkontaktieren

ohne Spezialwerkzeug mit
versilberten Kupferhohlknoten

Durchmesser:									
außen	0,6	0,8	1,0	1,2	1,5	1,8	2,0	2,5	mm
innen	0,4	0,6	0,7	0,9	1,0	1,2	1,5	2,0	mm
Kragen	0,9	1,1	1,4	1,6	2,0	2,3	2,5	3,2	mm
Menge:									
1000	43,-	37,-	28,-	28,-	28,-	29,-	31,-	32,-	€
3000	115,-	99,-	77,-	77,-	77,-	78,-	84,-	86,-	€
10000	350,-	305,-	230,-	230,-	230,-	240,-	255,-	265,-	€

Lieferung zzgl. Versandkosten

Elmar Wienecke, Dipl.-Ing.

Wasserstr. 18, D 32602 Vlotho, Tel. 05733/95610

elmar-wienecke@t-online.de

Die Zukunft des maschinellen Lernens

Ein Interview mit Daniel Situnayake

Fragen von **C. J. Abate** (Elektor)

Daniel Situnayake ist ein kreativer Ingenieur, der glaubt, dass Embedded Machine Learning (ML) eine „Technologie ist, die es nur einmal in einer Generation gibt“. Hier spricht er über das Potenzial von ML und stellt TinyML sowie einige ideale Anwendungen vor. Wir gehen auch auf seine Erfahrungen als Entwickler bei Google ein.



Bild 1. Daniel Situnayake (als Ingenieur Gründungsmitglied von TinyML, Edge Impulse).

Der Wechsel aus der akademischen Welt

Abate: Lassen Sie uns mit Ihrem Hintergrund beginnen. Wann haben Sie sich zum ersten Mal für Embedded ML interessiert? War es an der Birmingham City University (BCU), wo Sie einen Bachelor of Science (BSc) in Computer-Netzwerke und -Sicherheit erwarben? Oder geschah es, nachdem Sie die Universität verlassen hatten?

Situnayake: Ich hatte nicht vor, mich mit Embedded Machine Learning zu beschäftigen. Tatsächlich habe ich einen langen und kurvenreichen Pfad zurückgelegt, auf dem mich viele unterschiedliche Erfahrungen auf dieses Gebiet gestoßen haben! Nach meinem Abschluss an der BCU wurde ich gebeten, Fakultätsmitglied zu bleiben, um zu unterrichten und im Auftrag der Universität zu beraten. Ich legte meinen Schwerpunkt auf das Thema Automatische Identifikations- und Datenerfassungstechnologien, das alle Möglichkeiten betrachtet, wie ein Computer physische Objekte erkennen und verarbeiten kann - von der Biometrie, wie Gesichtserkennung, bis hin zu RFID, Erfassung von Strichcodes und Smartcards. Dies war meine erste Begegnung mit den beiden Themen Algorithmen für Computer Vision, die damals noch nicht auf maschinellem Lernen basierten, und eingebettete Systeme. Ein paar Jobs später in meiner Karriere, nach einer Übernahme, fand ich mich in dem entstehenden Data-Science-Team eines Bankunternehmens für Privatkundengeschäfte wieder. Dort begegnete ich zum ersten Mal Data-Science-Werkzeugen und -Arbeitsabläufen, und ich lernte die Arbeit mit großen Datensätzen kennen. Es war

kein Thema, mit dem ich vorher schon einmal zu tun gehabt hatte, aber ich fand es faszinierend.

Als wir Tiny Farms gründeten, sah ich zum ersten Mal die Gelegenheit, diese Bereiche miteinander zu verknüpfen, indem ich Data-Science-Tools benutzte, um Daten eingebetteter Sensoren in einem agrartechnologischen Umfeld zu interpretieren und darauf zu reagieren. Aber es war bei Google, wo ich Pete Warden (**Bild 2**) und Andy Selle vom TensorFlow-Lite-Team kennenlernte und wo ich das Potenzial des Deep Learnings auf winzigen Chips wirklich begriff und es zu meinem Schwerpunkt Nummer eins wurde.

Insektenzucht-Technologie

Abate: Sie waren Mitbegründer von Tiny Farms, das Sie auf Linked-In als „Amerikas erstes Technologieunternehmen für Insektenzucht“ bezeichnen. Wie kam es zur Gründung dieser Firma? Und wie geht es dem Unternehmen jetzt?

Situnayake: Wie alle meine besten Geschichten beginnt auch diese mit einer Reise quer durch die USA. Zwei enge Freunde von mir hatten darüber nachgedacht, ein Projekt zur Ernährungssicherung zu starten. Während einer Reise quer durchs Land wohnten sie in einer Hütte an einem See, der von großen, lauten Heuschrecken umgeben war. Aus Spaß fragten sie sich, ob die Heuschrecken vielleicht essbar wären. Nach ein paar Nachforschungen erfuhren sie, dass essbare Insekten für die globale Nahrungssicherheit tatsächlich äußerst wichtig sind - aber dass Techniken für ihre

Aufzucht in Gefangenschaft kaum erforscht worden waren. Nach einigen ersten Experimenten fragten sie mich, ob ich mit ihnen zusammen ein Unternehmen gründen wolle. Ich sah ein großes Potenzial für die Anwendung der Techniken, die ich im Zusammenhang mit der automatischen Identifizierung und Datenerfassung gelernt hatte, um die Aufzucht von Insekten zu automatisieren, und in der Nutzung der Data Science, um die iterative Entwicklung neuer Systeme zu steuern.

Ich bin ohne mit der Wimper zu zucken eingestiegen! Zuerst war es ein Teilzeitprojekt, aber schließlich brachten wir Geld auf und bauten eine Insektenfarm im industriellen Maßstab im kalifornischen San Leandro. Es war intensiv, aufregend und mehr als nur ein wenig stressig, zu versuchen, Millionen von Grillen am Leben zu erhalten, während wir mit Kunden, Investoren und der Forschung zu tun hatten. Nach etwa fünf Jahren beschloss ich, dass es an der Zeit wäre, das Handtuch zu werfen. Meine Mitbegründer machten ein paar Jahre lang weiter, aber letztendlich war es zu schwierig, ein marktfähiges Produkt herzustellen, und sie hörten auf. Tiny Farms gibt es weiterhin als Open-Source-Repository der Entwürfe und Daten, die wir in den sieben Jahren unserer Tätigkeit gesammelt haben [1].

Arbeiten bei Google

Abate: Sie haben fast drei Jahre bei Google verbracht. Bevor wir auf die Einzelheiten Ihrer dortigen Positionen eingehen, können Sie uns ein wenig darüber erzählen, wie es war, in ein so beeindruckendes Unternehmen einzusteigen?



Bild 2. Daniel Situnayke und Pete Warden signieren ihr TinyML-Buch.

damit man nicht von all den „Nebengeräuschen“ überfahren wird. Es ist ein weitverbreiteter Irrglaube, dass jeder, der bei Google arbeitet, eine Art Genie ist. Bei Google gibt es definitiv das gleiche Spektrum an Fähigkeiten wie in der übrigen Technologiebranche. Ich habe Personenverehrung noch nie gemocht, und es ist ein schönes Gefühl, wenn man einige der Top-Leute in seinem



Google hat eine erstaunliche Kultur der Freiheit: Man kann an so ziemlich allem arbeiten, woran man arbeiten möchte, solange es einen Einfluss auf das Produkt hat



Situnayake: Nachdem ich fünf Jahre lang ein landwirtschaftliches Unternehmen aufgebaut hatte - wobei ich mehr Zeit mit Bauarbeiten als mit der Programmierung verbracht hatte - wollte ich wieder ins kalte Wasser springen, und es gab kein Unternehmen, das ich technisch mehr bewunderte als Google.

Nach meinen ersten Tagen war ich ziemlich überwältigt. Ich hatte noch nie in einem großen Technologieunternehmen gearbeitet. Mit einem Google-Dienstausweis kann man die Räumlichkeiten jeder Google-Geschäftsstelle weltweit betreten, so dass es sich anfühlte, als wäre ich Bürger eines seltsamen, neuartigen Landes mit eigenen öffentlichen Diensten, eigener Infrastruktur und eigener Regierung geworden.

Google hat eine erstaunliche Kultur der Freiheit: Man kann so ziemlich an allem arbeiten, woran man arbeiten möchte, solange es sich auf das Produkt auswirkt. Wenn man eine coole Idee hat, wird man ermutigt, diese weiterzuverfolgen und andere Leute an Bord zu holen, die einem dabei helfen. Das führt zu einer riesigen Anzahl spannender Projekte, kann aber auch ein bisschen überwältigend sein: Wie entscheidet man bei all dieser Auswahl, wo man sich einbringt?

An dieser Stelle wird es ein bisschen verzwickter. Weil immer so viel los ist, gibt es potenziell endlose Besprechungen, Ablenkungen und Dinge, über die man nachdenken muss. Im ersten Jahr der Arbeit bei Google geht es darum, einen inneren Filter zu entwickeln,

Fachgebiet trifft und dabei erfährt, dass sie normale Menschen sind, genau wie man selbst.

Abate: Sie haben bei Google als Trainer für Entwickler von TensorFlow-Lite gearbeitet. Was ist TensorFlow-Lite? Und wie haben Sie Entwickler ausgebildet? Durch Meetups? Durch Online-Kurse?

Situnayake: TensorFlow ist der Name für Googles digitales Ökosystem von Open-Source-Tools für die Schulung, Bewertung und den Einsatz von Deep-Learning-Modellen. Es umfasst alles vom High-Level-Python-Code, der zur Definition von Modellarchitekturen verwendet wird, bis hin zum Low-Level-Code, der zur Ausführung dieser Modelle auf verschiedenen Prozessortypen verwendet wird.

TensorFlow-Lite ist die Untergruppe der TensorFlow-Werkzeuge, die sich mit der Anwendung von Modellen auf so genannte „Edge Devices“ befasst, zu denen alles gehört, was kleiner als ein Personal Computer ist: Telefone, Mikrocontroller und mehr. Die Tools von TensorFlow-Lite können ein Deep-Learning-Modell nutzen und es so optimieren, dass es besser für den Einsatz auf diesen Gerätetypen geeignet ist. Es umfasst auch den hoch optimierten Code, der auf den Geräten selbst läuft.

Mein Lieblingsbereich von TensorFlow-Lite ist die abgespeckte, superschnelle Variante, die für die Ausführung von Deep-Learning-Modellen auf winzigen, kostengünstigen Mikrocontrollern mit geringem Stromverbrauch konzipiert ist. Vor der Einführung

von TensorFlow-Lite für Mikrocontroller mussten die Entwickler ihren eigenen Low-Level-Code schreiben, um Modelle für maschinelles Lernen auf eingebetteten Geräten laufen zu lassen. Dadurch war die Hürde für den Einstieg in On-Device-Machine-Learning unglaublich hoch. Doch die Einführung von TensorFlow in diesem Bereich riss eine Menge Mauern ein, so dass plötzlich jeder mit etwas Erfahrung mit Embedded-Geräten Modelle ausführen konnte. Es war sehr aufregend zu sehen, wie der Bereich von TinyML zum Leben erwachte.

Meine Aufgabe bestand darin, dem Team von TensorFlow-Lite zu helfen, das Entwickler-Ökosystem zu verstehen und mit ihm in Verbindung zu treten. Alle Entwickler sind unterschiedlich, daher ist es wichtig, ihnen mehrere Möglichkeiten zu schaffen, sich zu beteiligen. Wir haben alles getan, was man sich vorstellen kann, von persönlichen Meetups und Google-finanzierten Konferenzen bis hin zur Erstellung von leicht verständlichem Beispielcode und dem Schreiben des TinyML-Buchs!

Abate: Warum haben Sie Google schließlich verlassen? War es eine schwere Entscheidung?

Situnayake: Bei der Zusammenarbeit mit den TensorFlow-Entwicklern habe ich festgestellt, dass die Nutzung selbst der besten ML-Tools und -Bibliotheken einen enormen Schulungsaufwand erfordert. Trotz einiger ausgezeichneten Kurse und Inhalte ist das maschinelle Lernen aufgrund des erforderlichen Zeitaufwands für eine große Anzahl vielbeschäftigter Ingenieure unerreichbar. Modelle zu trainieren ist mehr eine Kunst als eine Wissenschaft; instinktives Vorgehen und „best practice“ brauchen Jahre, damit sie sich entwickeln können. Mit TinyML ist es sogar noch schwieriger, da es sich um ein so neues Gebiet handelt und die Bedingungen und Grenzen nicht hinreichend bekannt sind.

Als ich von Edge Impulse hörte, dem Unternehmen, für das ich schließlich Google verlassen habe, war ich begeistert. Schon vor der offiziellen Markteinführung war es den Gründern gelungen, ein Produkt zu konstruieren, das jedem Entwickler ermöglichte, Deep-Learning-Modelle zu trainieren, die sich dazu eigneten, auf eingebetteter Hardware zu laufen. Durch meine Arbeit mit Entwicklern konnte ich erkennen, dass sie einige der schwierigsten Probleme in diesem Bereich gelöst und die Knackpunkte beseitigt hatten, die die Arbeit mit ML-Tools erschweren.

Ich glaube, dass Embedded Machine Learning eine Technologie ist, die es nur einmal in einer Generation gibt. Sie wird die Welt um uns herum auf eine Weise verändern, die uns erst später klar werden wird. Edge Impulse wurde gegründet, um diese Technologie in die Hände aller Ingenieure zu bringen, damit diese ihre vielfältigen Perspektiven und ihr tiefes Fachwissen nutzen können, um komplexe Probleme in jedem Teilgebiet unserer Welt zu lösen. Sobald ich dieses Potenzial erkannte, beschloss ich, an Bord zu springen. Es war traurig, meine Freunde bei Google zu verlassen, aber TinyML ist eine kleine Welt, und ich arbeite immer noch eng mit vielen von ihnen zusammen!

Tiny Machine Learning (TinyML)

Abate: Was sollten professionelle Ingenieure und seriöse Hersteller über TinyML wissen? Und können Sie ein oder zwei Beispiele für eine extrem energiesparende ML-Anwendung nennen?

Situnayake: Unsere Welt ist angefüllt mit Milliarden von Sensoren an allen möglichen Orten, von Haushaltsgeräten über intelligente Fabriken bis hin zu den Fahrzeugen, mit denen wir fahren. All diese

Sensordaten stellen für die Entwickler eine einmalige Gelegenheit dar, Geräte zu bauen, die die Welt um sie herum verstehen und intelligente Entscheidungen treffen, die den Menschen helfen.

Bisher bestand die einzige Möglichkeit, diese Daten zu nutzen, darin, sie über das Internet an einen leistungsfähigen Computer zu senden, der in der Lage ist, die Zahlen zu analysieren, die Daten zu interpretieren und auf der Grundlage der Ergebnisse Aktionen auszulösen. Dies funktioniert bei einigen Anwendungen sehr gut. Es ist zum Beispiel in Ordnung, wenn Daten von einer Wetterstation nur einige Sekunden brauchen, um zu einem Server zu gelangen. Aber es gibt viele Anwendungen, bei denen dies nicht funktioniert. Eines meiner Lieblingsbeispiele dafür ist die Kamerafalle. Wissenschaftler verwenden solche Fallen, um Tiere in freier Wildbahn zu beobachten, oft an abgelegenen Orten, an denen keine Internetverbindung zur Verfügung steht. Wenn ein Tier vorbeiläuft, löst es den Bewegungssensor der Kamerafalle aus und es wird ein Foto gemacht. Die Forscher kommen in regelmäßigen Abständen vorbei und entnehmen eine Speicherkarte voller Tierfotos.

Das Problem ist, dass der Bewegungssensor durch alle möglichen Dinge ausgelöst werden kann, wie zum Beispiel Blätter, die sich im Wind bewegen oder Tierarten, an denen die Forscher nicht interessiert sind. Es dauert lange, alle Fotos zu sichten, und die Speicherkarte wird schnell voll, was bedeutet, dass die Leute sie häufig austauschen müssen. Selbst wenn eine Internetverbindung zur Verfügung stünde, würde der zusätzliche Stromverbrauch bedeuten, dass sich die Batterien der Kamera schneller erschöpfen würden und gewechselt werden müssten.

Durch die Anwendung eines TinyML-Modells, das darauf trainiert ist, die richtige Tierart zu erkennen, kann die Falle das Fotografieren von sich im Wind bewegenden Blättern vermeiden. Das bedeutet, dass die Fallen länger an Ort und Stelle verbleiben können, wodurch die Forscher Zeit und Geld sparen.

Es gibt vier Hauptbedingungen, bei denen der Einsatz von TinyML von großem Vorteil ist. Wenn eine Anwendung eine begrenzte Konnektivität, eine geringe Latenztoleranz, eine begrenzte Stromquelle oder den Anspruch an eine starke Privatsphäre hat, kann TinyML potenziell helfen, indem es die Notwendigkeit verringert oder meidet, Daten vom Gerät zu senden.

Es ist erwähnenswert, dass nur bestimmte Probleme durch maschinelles Lernen gelöst werden können. Es müssen Probleme sein, bei denen ein bisschen Ungenauigkeit hingenommen werden kann, da ML selten vollkommene Ergebnisse liefern kann. Es eignet sich perfekt für die Aufnahme von Tierfotos, wo ein gelegentlicher Fehler akzeptabel ist, aber es könnte in einigen sicherheitskritischen Anwendungen nicht zuverlässig genug sein.

Abate: Sie haben das Buch „Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers“ mitverfasst. Da Sie den Arduino behandeln, bin ich geneigt zu glauben, dass die TinyML-Technologie sowohl für Nicht-Experten als auch für professionelle Ingenieure zugänglich ist. Abgesehen von der Lektüre Ihres Buches, was können Ingenieure und Innovatoren, die mit TinyML nicht vertraut sind, für einen schnellen Start tun?

Situnayake: Wie immer ist der beste Weg zu lernen, einige Projekte zu bauen! Ein guter Ausgangspunkt sind einige der Tutorials, die wir in der Edge-Impulse-Dokumentation anbieten: Bewegungserkennung mit einem Beschleunigungssensor [2], Erkennung von Sounds in einem Audiosignal [3] und die Hinzufügung von „Optik“ zu Ihren Sensoren [4].

Wir haben Edge Impulse so aufgebaut, dass Sie mit Ihrem Smartphone Daten erfassen und Modelle ausprobieren können, so dass Sie ohne spezielle Hardware loslegen können. Sobald Sie mit dem Ablauf des maschinellen Lernens vertraut sind, können Sie sich Ihr bevorzugtes Entwicklungsboard schnappen und mit dem Hacken beginnen!

Ich würde den *Arduino Nano 33 BLE Sense* als ein gutes Einstiegsboard für TinyML empfehlen. Es hat einen schnellen, aber stromsparenden Cortex-M4F-Prozessor und genügend RAM für einige interessante Modelle. Es wird von Edge Impulse vollständig unterstützt.

Edge Impulse

Abate: Sie sind im Januar 2020 zu Edge Impulse mit Sitz in San José gekommen. Wie kam es zu dieser Gelegenheit?

Situnayake: Ich traf den Geschäftsführer Zach Shelby während meiner Zeit bei Google. Er war zuvor stellvertretender Geschäftsführer bei ARM, die sehr eng mit dem TensorFlow-Lite-Team zusammenarbeiten. Als Zach ankündigte, dass er ARM verlassen würde, um ein TinyML-Unternehmen zu gründen, erregte das meine Aufmerksamkeit, und als sie ihr erstes Jobangebot veröffentlichten, wollte ich mich unbedingt bewerben!

Abate: Wie sieht Ihr Arbeitstag aus? Schreiben Sie den ganzen Tag Programme? Bilden Sie aus? Unternehmensentwicklung?

Situnayake: Eines der Dinge, die ich am meisten an meiner Arbeit bei Edge Impulse genieße, ist, dass ich wieder hauptsächlich Ingenieur bin. Ich verbringe die meiste Zeit damit, zu hacken. Ich treffe mich immer noch mit Kunden, was ziemlich lohnend ist und wichtig, damit ich weiß, worauf ich meinen Schwerpunkt bei der Arbeit als Ingenieur legen kann. Ich bin auch in der TinyML-Community sehr aktiv: Ich halte regelmäßig Vorträge und helfe dabei, TinyML-Meetups zu organisieren. Abgesehen davon habe ich normalerweise ein paar Meetings am Morgen, danach kann ich mich auf den Hosenboden setzen und für den Rest des Tages Code schreiben. Das ist großartig!

Abate: Wie ich sehe, hat das Edge-Impulse-Team im Mai mit einem COVID-19-bezogenen Projekt begonnen - „Hustenerkennung mit TinyML auf Arduino“ (Bild 3). Das Projekt wurde mehr als 5.400 Mal angesehen. Wie war die Reaktion aus der Community der Ingenieure? Wie geht es dem Projekt mittlerweile?

Situnayake: Auch wenn es Edge Impulse erst seit wenigen Monaten gibt, haben wir bereits eine großartige Community von Menschen mit den unterschiedlichsten Backgrounds. Als es mit COVID-19 losging, war eines unserer Communitymitglieder - Kartik Thakore, ein Datenwissenschaftler und biomedizinischer Ingenieur - daran interessiert, einige Projekte zur Unterstützung zu bauen. Wir dachten, dies wäre eine großartige Gelegenheit, um zu zeigen, was

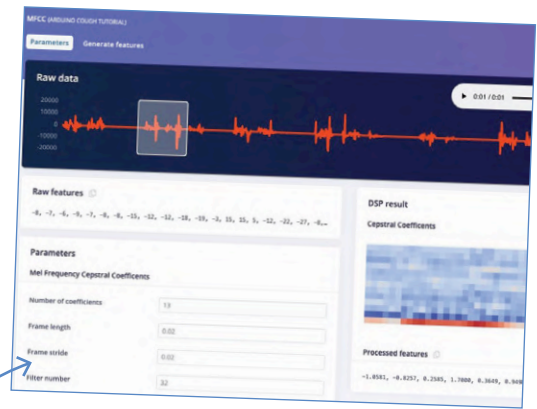


Bild 3. Hustenerkennung mit TinyML auf Arduino (Quelle: Edge Impulse).

mit TinyML möglich ist, also schickten wir ihm ein wenig Hardware, und er stellte ein Projekt zur Erkennung von Husten anhand eines TinyML-Modells zusammen. Das Ergebnis war dieses Projekt [5], von dem wir hoffen, dass es Hersteller inspiriert, ähnliche Herausforderungen anzugehen!

TinyML-Projekte

Abate: Gibt es weitere TinyML-Projekte, auf die Sie die Elektor-Community aufmerksam machen möchten?

Situnayake: Bei Edge Impulse glauben wir fest an die Technologie als eine Kraft des Guten, und einige meiner Lieblingsprojekte gehören in diese Kategorie. Ein Beispiel sind Smart Parks TinyML-Modelle, die trainiert werden, um wilde Elefanten aufzuspüren, so dass die Dorfbewohner gewarnt werden können, wenn die Dickhäuter auf dem Weg zum Ort sind.

Eine andere Lieblingskategorie von mir sind Menschen, die es mit Computern ermöglichen, verrückte Dinge zu tun, die man nie für möglich gehalten hätte. Zum Beispiel hat Benjamin Cabé ein erstaunliche Projekt geschaffen, das einen Gassensor verwendet, um verschiedene Marken alkoholischer Getränke zu erkennen! Wenn Sie es nachbauen möchten: Es gibt es ein fantastisches Tutorial von Seeed Studio. ◀

200423-02

Sie haben Fragen oder Kommentare?

Haben Sie Fragen oder Anmerkungen zu seinem Artikel? Kontaktieren Sie Elektor unter editor@elektor.com.

Ein Beitrag von

Interviewer: **C. J. Abate**

Übersetzung: **Katharina Sophia Gerstendorf**

Layout: **Giel Dols**

WEBLINKS

- [1] **Open Tiny Farms:** www.opentinyfarms.com/
- [2] **Edge Impulse, „Continuous Motion Recognition“:** <https://docs.edgeimpulse.com/docs/continuous-motion-recognition>
- [3] **Edge Impulse, „Recognize Sounds from Audio“:** <https://docs.edgeimpulse.com/docs/audio-classification>
- [4] **Edge Impulse, „Adding Sight to Your Sensors“:** <https://docs.edgeimpulse.com/docs/image-classification>
- [5] **Edge Impulse, „Cough Detection with TinyML on Arduino“:** <http://bit.ly/edge-cough-detection-project>

Mobile Apps für Android und iOS

Aus einem Guss programmiert



Von **Dr. Veikko Krypczyk**

Für immer mehr Anwendungen (zum Beispiel rund um das Thema Internet of Things) wird eine App für mobile Geräte benötigt. Dabei muss man üblicherweise sowohl Android als auch iOS bedienen. Eine Vielzahl von unterschiedlichen Ansätzen rangelt sich um die Gunst des Entwicklers. Gefragt sind plattformübergreifende Ansätze und ein möglichst intuitives Vorgehen zum Erstellen der Oberfläche. Die Entwicklungsumgebung RAD Studio – besser bekannt als Delphi – kann ein passender Ansatz sein.

Immer mehr Anwendungen aus dem Bereich der Elektronik setzen auf ein ausgeklügeltes Zusammenspiel von Hard- und Software. Dabei kommen vermehrt Apps für mobile Geräte wie Tablets und Smartphones zum Einsatz, auf denen die Betriebssysteme Android und iOS laufen. Die Herausforderung besteht darin eine individuelle App für beide Systeme zu erstellen, welche sich genau an den Anforderungen des aktuellen Projektes orientiert. Ist man nicht primär mit der Entwicklung von mobilen Apps beschäftigt, dann kann der Einstieg schwerfallen. Eine Vielzahl von Ansätzen konkurrieren untereinander, oft sind die Lernkurven steil. Auf diese Weise wird die Programmierung für die kleinen Geräte zu einer großen Sache,

vielleicht sogar zu einem Stolperstein für das gesamte Projekt. Abhilfe können plattformübergreifende Ansätze bieten, welche ein weitgehend intuitives Vorgehen bei der Erstellung der Benutzeroberfläche unterstützen. In diesem Artikel geben wir einen kompakten Überblick über die Möglichkeiten der Entwicklung von mobilen Apps, bevor wir uns dann einen interessanten Ansatz genauer ansehen. Mit Hilfe von RAD Studio – vielen Praktikern auch unter dem Namen Delphi als effizientes Tool zur Entwicklung von Desktop-Applikationen bekannt – kann man heute mit einem deutlich reduzierten Aufwand Apps für Android und iOS aus einer Quellcodebasis erstellen. Programmiert wird mit Delphi (Objekt Pascal) und der umfassenden Unter-

stützung eines grafischen Designers. Ein Beispiel wird zeigen, wie es geht.

Apps für mobile Geräte

Aus technischer Perspektive unterscheidet man folgende Arten von Apps:

- > native Apps
- > Web-Apps
- > hybride Apps.

Native Apps werden exklusiv für Android oder iOS erstellt. Man nutzt direkt die API der Systeme. Die Benutzeroberfläche fügt sich bestmöglich in die Plattform ein. Ein weiterer Vorteil besteht darin, dass diese keine Einschränkungen beim Zugriff auf mögliche

spezifische Gerätehardware des Smartphones oder Tablets aufweisen. Alle Sensoren dieser Geräte können direkt angesprochen werden. Die Bereitstellung der App erfolgt über die App-Stores. Ist eine native App installiert, kann diese – sofern sinnvoll – auch ohne Internetzugriff (offline) ausgeführt werden. Eine notwendige Datensynchronisation kann beim Herstellen der nächsten Online-Verbindung automatisch stattfinden. Apps für iOS (Apple) werden mit Xcode (Entwicklungsumgebung) und Swift (Programmiersprache) erstellt. Für Android kommt Android Studio und Kotlin bzw. Java zum Einsatz.

Web Apps sind dagegen auf die mobilen Devices ausgerichtete Web-Applikationen. Das betrifft primär die Gestaltung des User-Interfaces. Der Zugriff auf die Systemhardware ist eingeschränkt. Einige Basisfunktionen, wie zum Beispiel die Ortung via GPS, sind jedoch möglich. Man nutzt dafür entsprechende JavaScript-APIs. Eine Bereitstellung der Apps über die Stores ist jedoch nicht möglich. Sie laufen auf einem Server und benötigen daher eine stetige Internetverbindung. Jedoch kann man ein Icon auf dem Home-Screen einrichten, so dass die Nutzer die App schnell starten können. Die fehlende Offline-Fähigkeit kann durch den Ansatz der Progressive Web App (PWA)-Technologie teilweise beseitigt werden. Eine PWA ist eine Symbiose aus einer Webseite und einer App. Mittels eines Service Workers kann eine Caching-Funktion umgesetzt werden. Dieser Service Worker ist zwischen dem Webserver und der App auf dem mobilen Gerät geschaltet. Um eine Web-Applikation für die mobilen Systeme zu erstellen, steht Ihnen die gesamte Bandbreite der Web-Technologien zur Auswahl. Insbesondere die gängigen Bibliotheken und Frameworks können verwendet werden. Letztendlich basieren diese Apps final auf den Elementen HTML (Struktur), CSS (Design) und JavaScript (Interaktion, Logik), da der Browser nur diese Sprachen interpretieren kann.

Hybride Apps basieren intern auf Web-Technologien. Die Web-App wird in einem eingebetteten Webbrowser ausgeführt, d.h. sie läuft in einer Art „Sandkasten“, worüber die Plattformunabhängigkeit erreicht wird. Das System glaubt daher eine native App (den Browser) vor sich zu haben. Es stehen mehrere Ansätze für hybride Apps zur Wahl, zum Beispiel Cordova der Apache Software Foundation. Das Prinzip ist stets ähnlich. Die App öffnet beim Starten ein Browserfenster im Vollbildmodus, so dass der Browser als solcher nicht

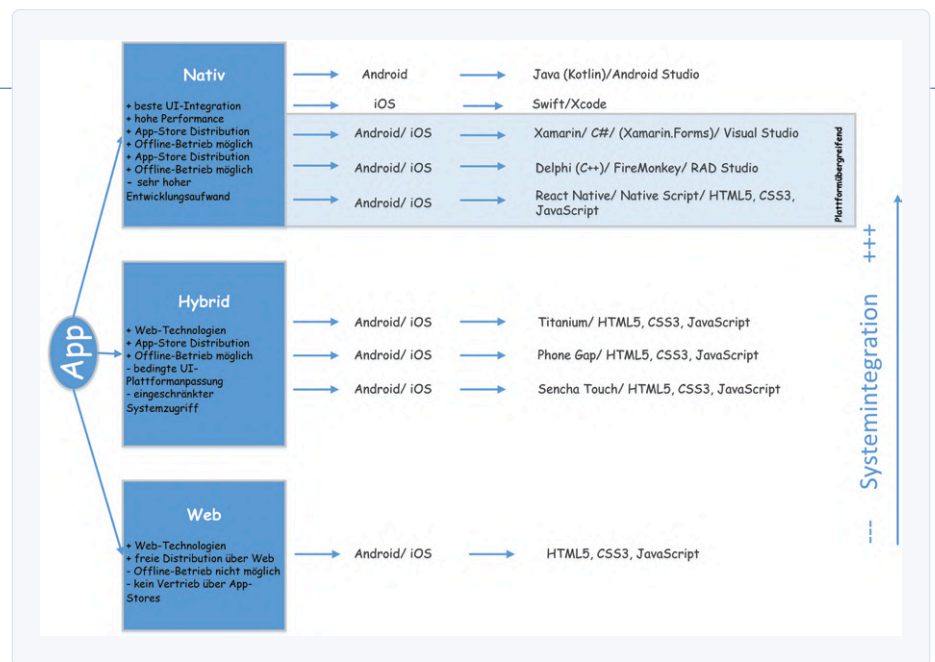


Bild 1. App-Arten und Wege zur App.

identifizierbar ist und die Web-Adresse nicht geändert werden kann. Im Browser wird die mit HTML, CSS und JavaScript erstellte Web-App angezeigt. Über das Framework bekommt die App Zugriff auf die Systemfunktionen, wie die Kamera oder das Adressbuch. Das geschieht mit Hilfe von Plug-Ins. Sowohl Web-Apps als auch hybride Apps zielen simultan auf Android und iOS. Sie weisen jedoch Nachteile wie eine schlechtere Performance und einen eingeschränkten Hardwarezugriff gegenüber den nativen Apps auf. Daher versucht man beide Arten miteinander über **Cross-Plattform-Ansätze** zu verknüpfen. Das Ziel ist es, dem nativen Vorbild (auch beim Aussehen der Benutzeroberfläche) möglichst nahe zu kommen und dabei die App für Android und iOS aus einer gemeinsamen Quellcodebasis zu erstellen. Es gibt eine große Anzahl von Cross-Plattform-Ansätzen, welche sehr unterschiedlich in der Herangehensweise sind, dazu zählen unter anderem Xamarin, RAD Studio, Native-

Script und React Native. Die vorherstehenden Aspekte sind in **Bild 1** zusammengefasst.

In diesem Artikel sehen wir uns die Vorgehensweise mit **RAD Studio (Delphi)** näher an. Dieser Ansatz scheint gerade im Bereich von Elektronik- und IoT-Projekten gut geeignet. Das Vorgehen ist vielen Praktikern vertraut. Mit Hilfe eines grafischen Designers kann – ähnlich wie bei einer Entwicklung für den Windows Desktop – das User Interface erstellt werden. Die Logik wird in der Sprache Delphi (Object Pascal) geschrieben, welche ebenfalls intuitiv und wenig komplex ist. Manche Elektroniker werden mit Pascal auch vertraut sein, zum Beispiel aus der Programmierung von Embedded-Software für Mikrocontroller.

Installation und Systemkonfiguration

Wir installieren jetzt Delphi unter Microsoft Windows und richten es für die mobile Entwicklung ein. Laden Sie die Community

RAD STUDIO, DELPHI UND C++-BUILDER

Delphi und C++-Builder sind integrierte Entwicklungsumgebungen von Embarcadero, um Anwendungen für unterschiedliche Betriebssysteme zu erstellen. Bei Delphi arbeitet man mit der gleichnamigen Programmiersprache Delphi (eine Weiterentwicklung von Object Pascal), beim C++-Builder mit C++. In RAD Studio werden beide Umgebungen zusammengefasst. Es stehen mehrere Editionen zur Verfügung. Für den professionellen Einsatz sind es die kommerziellen Produkte Professional, Enterprise und Architect. Für eine eingeschränkte kommerzielle Nutzung, zum Testen, Lernen, für private und Open-Source-Projekte gibt es die kostenfreie Community Edition. Diese verwenden wir auch hier. Aktuell ist Version 10.4 von RAD Studio.

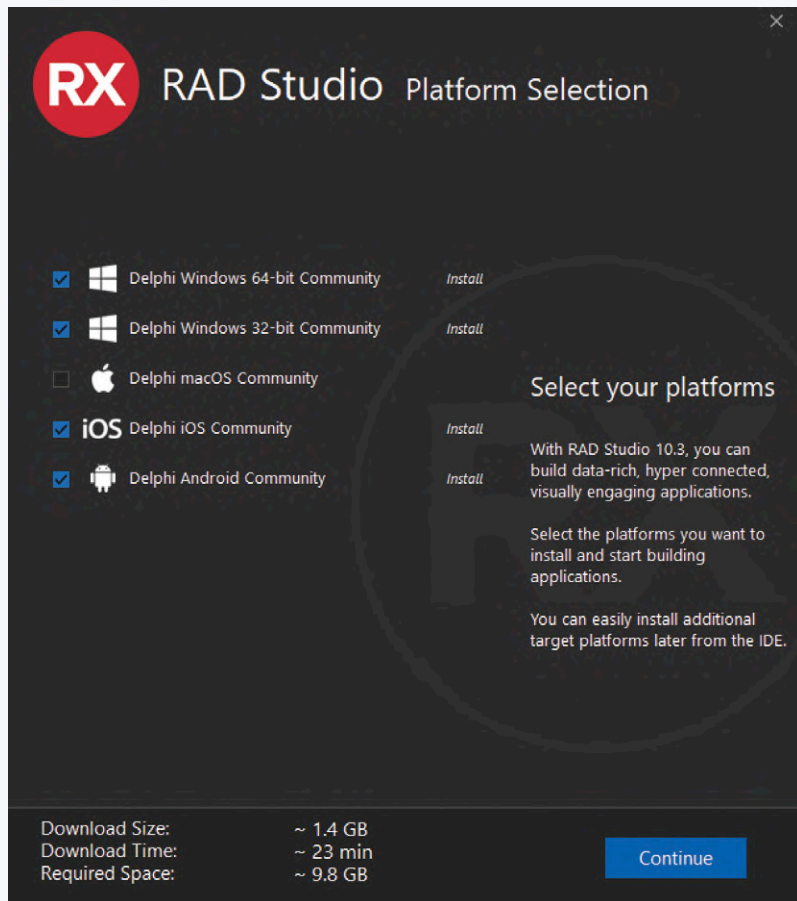


Bild 2. Auswahl der Plattformen.

Edition von [1] nach Registrierung herunter und führen Sie das Installationsprogramm aus. Während der Installation erhalten Sie ein Dialogfeld mit einer Auswahloption der Plattformen (Bild 2).

Wählen Sie hier neben *Windows* auch *Android*

und *iOS*. Als Nächstes werden Sie aufgefordert, zusätzliche Optionen auszuwählen. Das sollten Sie für das *Android SDK/NDK* machen. Nach der Installation starten Sie die Entwicklungsumgebung (Bild 3).

Für die mobile Entwicklung sind noch ein

paar Nacharbeiten notwendig. Möchten Sie die App auf iOS testen, dann benötigen Sie über das Netzwerk den Zugriff auf einen Apple PC mit Xcode. Ein Simulator für iOS kann nur unter macOS ausgeführt werden. Auch ein physisches iPhone wird direkt mit dem Mac-PC gekoppelt. Über das Netzwerk wird dann von Windows aus über das RAD Studio auf den Mac-PC zugegriffen (Bild 4).

Dazu muss auf dem Mac die Software *PASever* installiert werden, damit der Fernzugriff möglich ist. Arbeiten Sie grundsätzlich auf einem Mac-PC, dann können Sie Windows in einer virtuellen Maschine (Parallels, Virtual Box usw.) ausführen und in dieser virtuellen Maschine RAD Studio installieren. Aus dieser wird dann auf den Mac-PC zugegriffen, d.h. alle Komponenten zur Entwicklung sind auf einem Rechner installiert. Die Notwendigkeit eines Mac-PCs, um Apps für iOS zu bauen, ist keine Besonderheit von RAD Studio, sondern trifft bei allen Vorgehensweisen zu.

Apps für Android können Sie direkt aus Windows erstellen. Zum Testen können Sie ein Smartphone/ Tablet mit Ihrem Rechner über USB anschließen oder (alternativ) einen Emulator installieren. Ein „echtes“ Gerät ist jedoch deutlich schneller. Schließen Sie Ihr Smartphone mit dem USB-Kabel an den Computer an und konfigurieren Sie es ordnungsgemäß, d.h. schalten Sie unter den Einstellungen die Entwicklerberechtigungen inklusive USB-Debugging auf dem Smartphone frei. Überprüfen Sie nun im Gerätemanager, ob Ihr mobiles Gerät korrekt erkannt und eingerichtet wurde. Die offizielle Dokumentation zum Einrichten von Delphi für die mobile Entwicklung finden Sie unter [3] und [4]. Damit sind die Konfigurationsarbeiten auch schon abgeschlossen. Ein erster Test („Hello World“) verdeutlicht die Vorgehensweise zur Entwicklung einer mobilen App.

Hello Elektor

In diesem Abschnitt geht es um das Erstellen eines ersten Projektes. Wählen Sie in der Entwicklungsumgebung von Delphi im Hauptmenü *File | New | Multi-Device-Application - Delphi* und klicken Sie im Fenster auf *Blank Application* (Bild 5) und dann auf *OK*.

Speichern Sie das Projekt. Über die Tool-Palette können wir mit Hilfe von visuellen Steuerelementen die Benutzeroberfläche erstellen. Ziehen Sie dazu testweise eine Komponente vom Typ *TLabel* auf das Dialogfeld. Sie können die Vorschau für die unterschiedlichen Systeme (Windows, Android, iOS) umschalten. Im *Objektsinspektor* können die Eigenschaf-

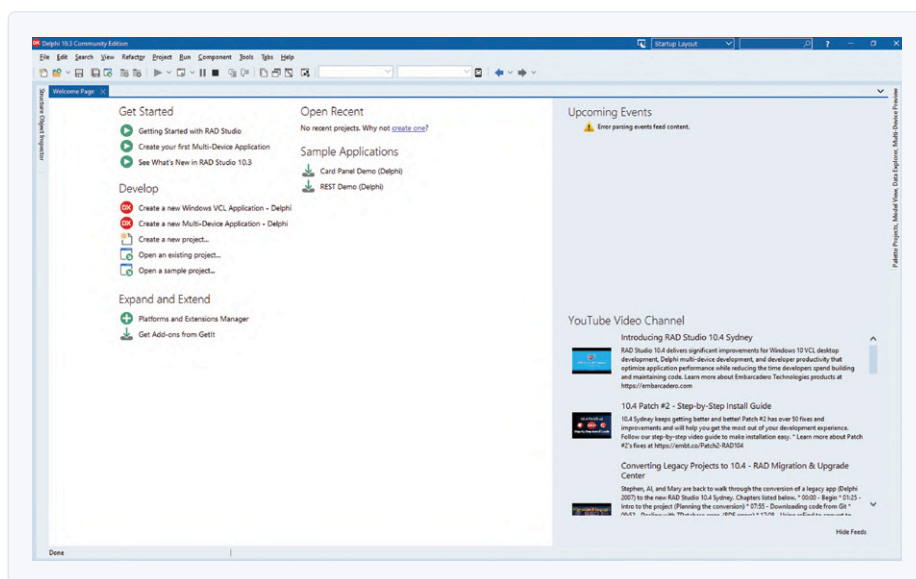


Bild 3. Startbildschirm der integrierten Entwicklungsumgebung von Delphi.

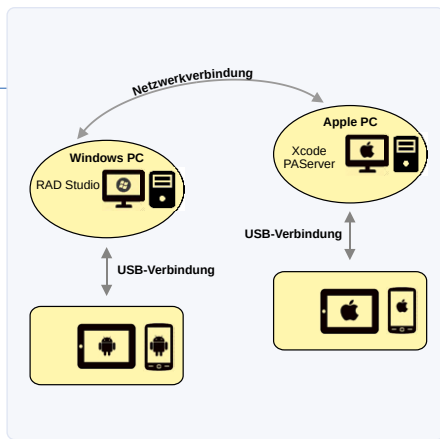


Bild 4. Systemkonfiguration zur Entwicklung von Apps für iOS.

KEIN IOS OHNE MACOS UND XCODE

iOS ist ein geschlossenes System, d.h. für das Erstellen der App Packages und die Verteilung auf die mobilen Devices wird Xcode (integrierte Entwicklungsumgebung) und damit ein Mac-PC mit macOS benötigt. Ebenso lässt sich der iOS-Simulator nur unter macOS betreiben. Alternativ kann man einen Cloudservice nutzen, d.h. einen gehosteten Mac-PC mit allen notwendigen Entwicklungswerkzeugen „mieten“ und diesen über eine Remote-Verbindung steuern. Ein solches Angebot bietet der Dienst MacInCloud [2]. Der Bildschirm wird dann auf den eigenen Entwicklungsrechner übertragen. Das löst das Problem, so dass man das App-Package erstellen und den iOS-Simulator während der Programmierung verwenden kann. Ein echtes Device kann man natürlich über die Cloud nicht nutzen. Der Autor hat gute Erfahrungen mit dem genannten Dienst gemacht. Die Software PAserver zur Fernsteuerung des Mac ist bereits vorinstalliert, d.h. man kann nach Registrierung und Anmeldung direkt loslegen. Für den Einstieg gibt den flexiblen Pay-as-You-Go-Plan, d.h. man zahlt nur die Zeiteinheiten für die Nutzung des Mac in der Cloud.

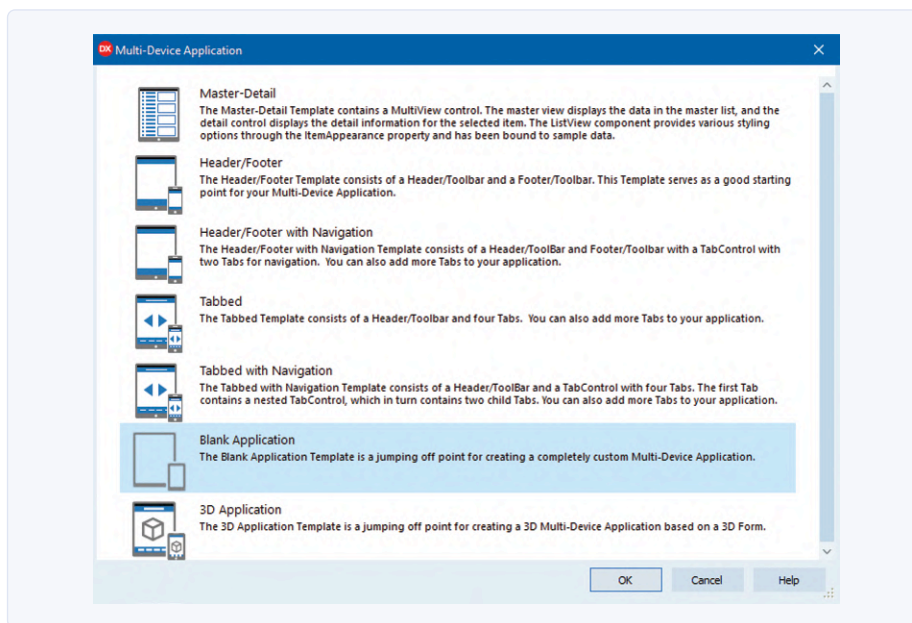


Bild 5. Eine neue geräteübergreifende Anwendung erstellen.

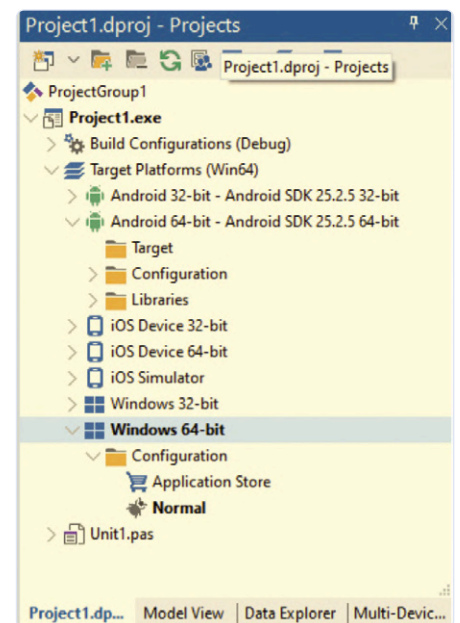


Bild 6. Zielplattformen werden direkt in Delphi ausgewählt.

ten der Komponenten konfiguriert werden. Wir setzen die *Text*-Eigenschaft auf den Wert „Hello Elektor“. Speichern Sie das Projekt und die zugehörigen Dateien über das Menü *File*. Nun geht es bereits um einen ersten Test auf den unterschiedlichen Plattformen. Öffnen Sie im *Projektmanager* den Knoten *Zielplattformen* und wählen Sie die Plattform *32-Bit-Windows* durch Doppelklick aus (Bild 6). Starten Sie die Anwendung (grüner Pfeil). Diese wird als „normale“ Windows-Applikation ausgeführt (Bild 7).

Schließen Sie nun die Anwendung und gehen Sie zur Entwicklungsumgebung zurück. Wählen Sie nun als Zielplattform *Android 64-Bit*. Sofern die Konfiguration des Smartphones/ Tablets oben korrekt erfolgt ist, erscheint jetzt hier das angeschlossene Gerät. Aktivieren Sie dieses und starten Sie die App erneut. Die App wird auf dem mobilen Gerät

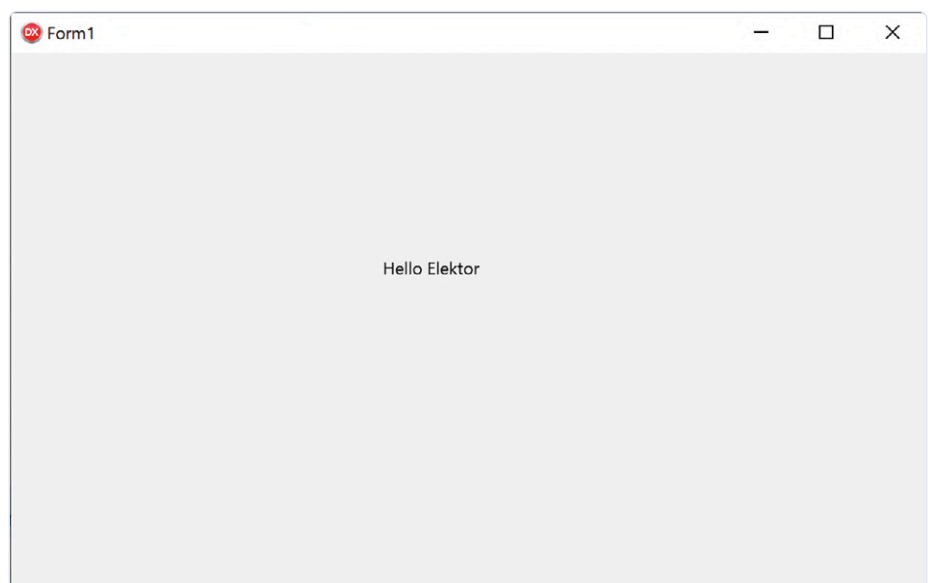


Bild 7. Die Test-App als Windows-Applikation.

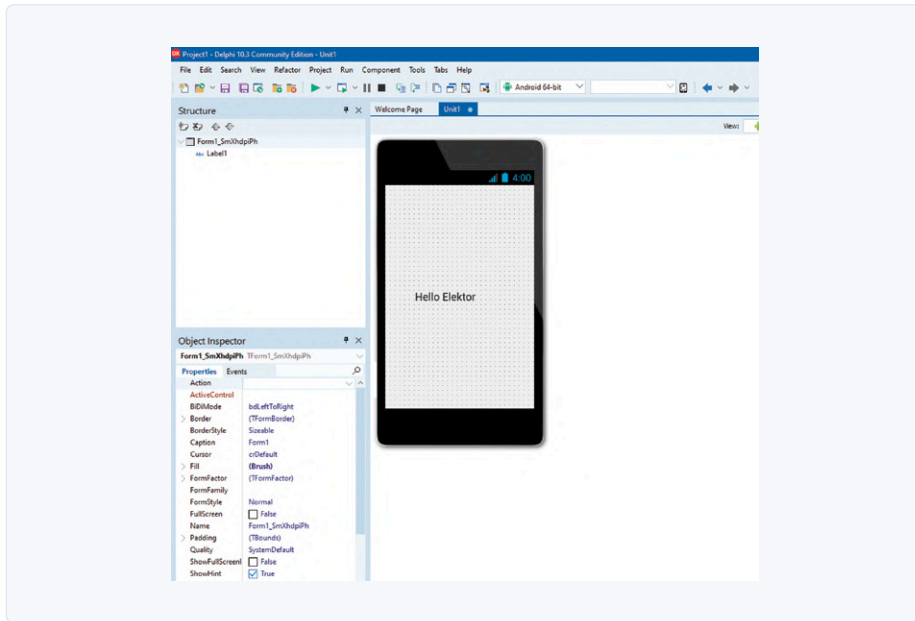


Bild 8. Die Vorschau der App für Android im Designer von RAD Studio.

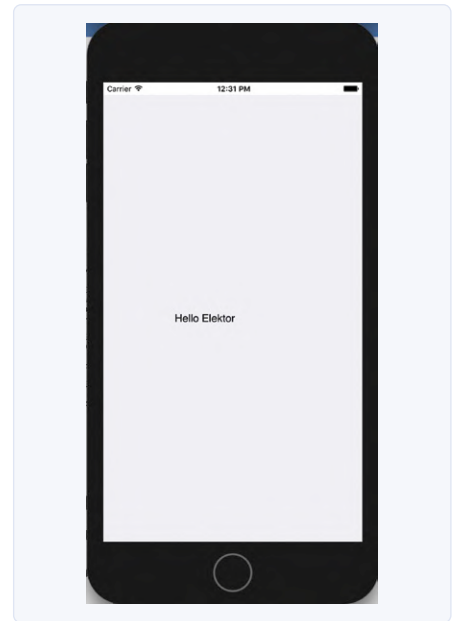


Bild 9. „Hello Elektor“ auf dem iPhone-Simulator.

installiert und ausgeführt. Auch im Designer der Entwicklungsumgebung können Sie die die App für ein Android-Gerät in der Vorschau anzeigen lassen (Bild 8).

Sofern Sie das System auch für iOS konfiguriert haben, können Sie die Verbindung zum Mac-PC und über den PAServer (siehe oben) zu Xcode, den Simulator und einem iPhone herstellen und die App dort ausführen (Bild 9). Nach dieser minimalen Einführung wollen wir nun eine etwas umfassendere App erstellen

und uns mit der Arbeitsweise weiter vertraut machen.

Eine App für Elektroniker

In diesem Abschnitt beschreiben wir die Entwicklung einer App zur Bestimmung des Widerstandswertes aus den Farbringen. Als Erstes gestalten wir die Oberfläche im grafischen Designer. Wählen Sie als Vorlage wiederum **File | New | Multi-Device-Application – Delphi** und jetzt die Auswahl **Tabbed**.

Wir erhalten eine Ansicht mit Registerkarten (in Android oben, in iOS unten). Die Inhalte dieser Registerkarten müssen wir nun entsprechend befüllen. Die Vorlage enthält vier Registerkarten (Typ *TTabItem*), welche wir auf zwei reduzieren, d.h. wir löschen zwei Registerkarten über die Ansicht **Structure / Struktur**. In Registerkarte Eins kommen die Elemente zur Auswahl des Farbcodes und in Registerkarte Zwei platzieren wir allgemeine Informationen zur App. Es hat sich bewährt, vor der Umsetzung der Benutzerschnittstelle eine Skizze anzufertigen und damit zu klären, an welcher Stelle die Steuerelemente platziert werden sollen (Papier und Bleistift genügen). Die Anordnung der Steuerelemente erfolgt in der Regel nicht mit absoluten Positionsangaben, denn man muss berücksichtigen, dass wir die App auf mobilen Geräten mit unterschiedlichen Bildschirmgrößen und -auflösungen starten. Eine Tabellenstruktur kann ein guter Ansatz für die grobe Einteilung der Platzverhältnisse sein. In Zeilen und Spalten werden dann die einzelnen Elemente angeordnet. Für jede Zeile und Spalte können wir wiederum die Größe (absolut, besser relativ) vorgeben. Für unsere App wählen wir ein Raster von fünf Zeilen und vier Spalten. Verwenden Sie dazu ein Steuerelement des Typs *TGridPanelLayout*. Wählen Sie dieses aus der Palette aus und ziehen Sie es auf die erste Registerkarte. Damit es den gesamten Platz einnimmt, setzen Sie die Eigenschaft **Align** auf den Wert **Client**. Fügen Sie über die **Struktur**-Ansicht insgesamt fünf Zeilen

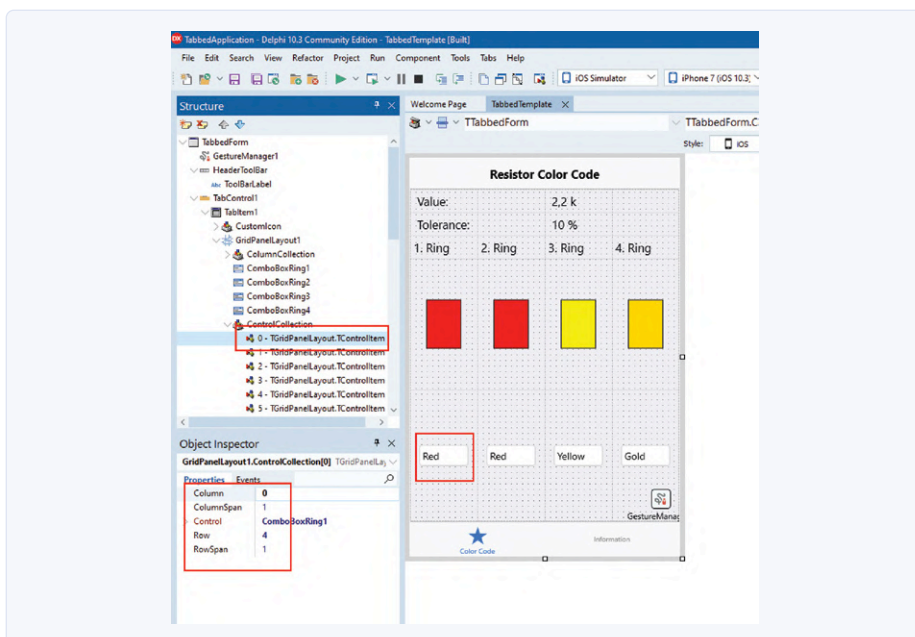


Bild 10. Die Positionierung der Steuerelemente erfolgt hier in einem Tabellenraster.

Farbe	Ring 1 Stelle 1	Index in TComboBox	Ring 2 Stelle 2	Index in TComboBox	Ring 3 Multiplikator	Index in TComboBox	Ring 4 Toleranz	Index in TComboBox
Schwarz	0	0	0	0	1	0	-	-
Braun	1	1	1	1	10	1	1%	0
Rot	2	2	2	2	100	2	2%	1
Orange	3	3	3	3	1.000	3	-	-
Gelb	4	4	4	4	10.000	4	-	-
Grün	5	5	5	5	100.000	5	0,5%	2
Blau	6	6	6	6	1.000.000	6	0,25%	3
Violett	7	7	7	7	10.000.000	7	0,1%	4
Grau	8	8	8	8	-	-	0,05%	5
Weiß	9	9	9	9	-	-	-	-
Gold					0,1	8	5%	6
Silber					0,01	9	10%	7

Tabelle 1. Bedeutung der Farbringe und Position (Index) in der jeweiligen Auswahlbox (TComboBox).

und vier Spalten zu diesem *TGridPanelLayout* hinzu. Die Größe der Zeilen können Sie prozentual zum Beispiel wie folgt aufteilen: 7%, 7%, 7%, 40% und 39% (insgesamt = 100%). Die vier Spalten teilen Sie gleichmäßig, d.h. jede mit einer Größe von 25%, auf. Nun geht es darum, die Steuerelemente in diesem Tabellenlayout zu platzieren. Wählen Sie ein gewünschtes Element aus der Palette und ziehen Sie es auf die Registerkarte. Für die Farbauswahl verwenden wir Steuerelemente vom Typ *TComboBox*. In der Ansicht *Struktur*-Ansicht können wir dann unterhalb des Knotens *TabItem1 | Control Collection* die exakte Position (Zeile und Spalte) für jedes Element einstellen. In **Bild 10** sehen Sie diese für die Farbauswahl des ersten Rings.

Es werden die Positionen *Column: 0* und *Row: 4* ausgewählt, da die Auswahlbox auch an dieser Stelle platziert werden soll. Für Textanzeigen verwenden wir Steuerelemente vom Typ *TLabel*, zum Beispiel für Beschriftungen oder Hinweise. Ebenso verwenden wir *TLabel* für Werte, welche wir später einblenden möchten, zum Beispiel für den berechneten Widerstandswert. Die Farbringe simulieren wir über farbig befüllte Rechtecke, d.h. Steuerelemente des Typs *TRectangle*. Deren Farbe wollen wir später aus dem Quellcode heraus anpassen. In den Auswahlfeldern für die Farbringe (*TComboBox*) fügen wir die zur Auswahl stehenden Farben nach dem in **Tabelle 1** bekannten Muster hinzu.

Das Layout der Anwendung im grafischen Designer ist in **Bild 11** zu sehen. Nach einem

Start der App (Simulator oder physisches Gerät) sehen wir das gestaltete Layout der Registerkarte (**Bild 12**).

Die Registerkarte *Information* haben wir zunächst nur mit einem Logo (Steuerelement des Typs *TImage*) und einem Text befüllt (**Bild 13**).

Ist man so weit gekommen, besteht der nächste Schritt darin, die Logik zur Ermittlung des Widerstandswertes zu ermitteln.

Eine Neuberechnung und Aktualisierung der Anzeige muss immer dann durchgeführt werden, wenn wir einen neuen Farbwert auswählen. Damit wir die Steuerelemente der Oberfläche im Quellcode einfacher identifizieren können, geben wir den Elementen eindeutige Namen. Es wird dazu jeweils die Eigenschaft *Name* des Elements angepasst, also zum Beispiel *LabelValue*, *RectangleRing1*, *ComboBoxRing1* usw.

Wir selektieren die Auswahlboxen für die

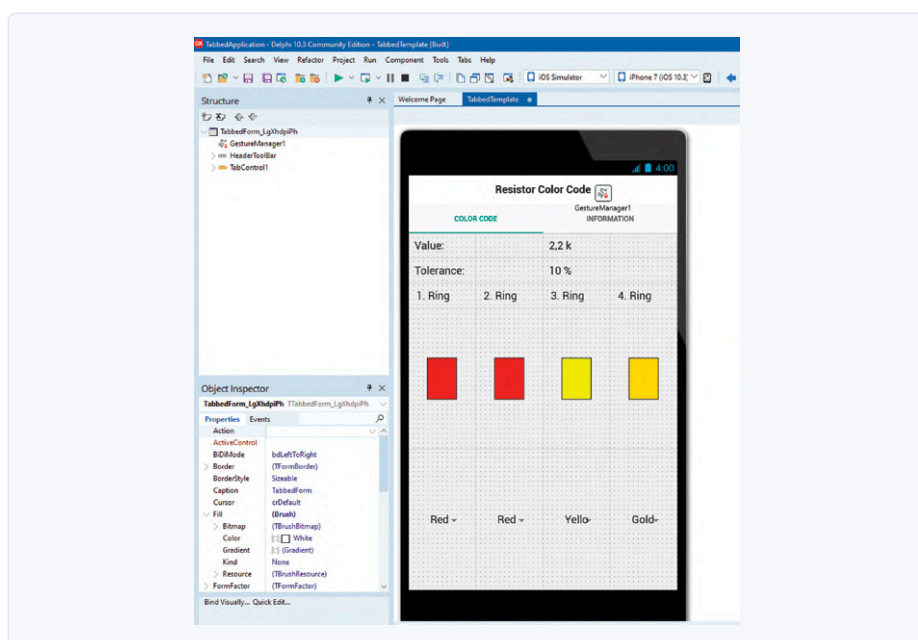


Bild 11. Das Layout der App in Delphi – Ansicht Android.

Listing 1. Anpassung der Farbanzeige je nach Auswahl.

```
case digit1 of
  0:
  begin
    RectangleRing1.Fill.Color := TAlphaColors.Black;
  end;
  1:
  begin
    RectangleRing1.Fill.Color := TAlphaColors.Brown;
  end;
  2:
  begin
    RectangleRing1.Fill.Color := TAlphaColors.Red;
  end;
  ...
  ...
  9:
  begin
    RectangleRing1.Fill.Color := TAlphaColors.White;
  end;
end;
```

Listing 2. Berechnung des Widerstandswertes und Formatierung der Anzeige.

```
if digit3 < 8 then
  value := Power10((digit1 * 10 + digit2), digit3)
else if digit3 = 8 then
  value := (digit1 * 10 + digit2) * 0.1
else if digit3 = 9 then
  value := (digit1 * 10 + digit2) * 0.01;

result := FloatToStr(value) + ' Ohm';

if value > 1000 then
  result := FloatToStr((value / 1000)) + ' k Ohm';

if value > 1000000 then
  result := FloatToStr((value / 1000000)) + ' M Ohm';

LabelValue.Text := result;
```

Farben der Ringe alle gemeinsam und wählen das Ereignis *OnChange*. Mit einem Doppelklick im Objektinspektor veranlassen wir die Entwicklungsumgebung eine Prozedur für dieses Ereignis anzulegen. Dieser Quellcode wird also immer dann aufgerufen, wenn wir die Farbauswahl ändern. Zunächst passen wir die Farbe der Rechtecke gemäß der Auswahl des Rings an. Das geschieht zum Beispiel über:

```
RectangleRing1.Fill.Color :=
  TAlphaColors.Red
```

Damit wird das Rechteck rot gezeichnet. Für die erste Auswahlbox sieht dieses dann wie in **Listing 1** dargestellt aus.

Für die anderen Farbringe wird die Auswahl entsprechend dem obengenannten Index (siehe **Tabelle 1**) angepasst. Es folgt die Berechnung des Widerstandswertes, das heißt:

$(\text{Wert von Stelle 1} * 10 + \text{Wert von Stelle 2}) * 10^{\text{Multiplikator}}$

Die Berechnung erfolgt in der Einheit Ohm. Für Werte größer 1.000 wird eine Umrechnung in kΩ und für Werte > 1.000.000 in MΩ vorgenommen. Für die Ringfarben Gold und Silber erfolgt eine Multiplikation mit den Werten 0,1 bzw. 0,01. Der Wert für die Toleranz muss nur aus der obigen Tabelle abgebildet werden (**Listing 2**).

Damit ist unsere App funktionsfähig: Sie können eine beliebige Farbkombination der Ringe auswählen und der korrekte Wert des Widerstandes wird angezeigt. Sie können nun die App-Packages für Android und/oder iOS erstellen und die App bereitstellen. Auch eine Bereitstellung im Google Play oder Apple Store ist aus Delphi möglich. Den Quellcode zum Programm-Beispiel können Sie von der Elektor-Webseite zu diesem Artikel [5] herunterladen.

Weitere Möglichkeiten

Im Beispiel haben wir bereits einige Möglichkeiten der App-Entwicklung mit Delphi gezeigt. Doch hier geht noch viel mehr. Dazu einige Beispiele:

➤ **Visuelle Komponenten:** Um ansprechende Oberflächen zu erstellen, stehen eine Vielzahl von Komponenten (Buttons, Textfelder, Auswahlfelder, Registerkarten usw.) zur Verfügung. Diese Steuerelemente können Sie mit Hilfe des grafischen Designers platzieren und über die Eigenschaften konfigurieren. Das System erstellt automatisch beim Kompilieren der Anwendung die passenden Elemente für die Zielplattform (Android, iOS, Windows).

➤ **Nicht visuelle Komponenten:** Diese stellen wichtige Bausteine der Programmierung bereit, welche immer wieder benötigt werden. Beispiele sind eine Timer-Komponente (Zeitgeber) oder Komponenten, um einen Zugriff auf Datenbanken zu erleichtern.

➤ **Mobile App-Spezifika:** Es werden visuelle und nicht visuelle Komponenten angeboten, welche es ermöglichen, typische Aufgaben bei der Programmierung von mobilen Apps schneller zu lösen, zum Beispiel der Zugriff auf die Ortungsfunktionen oder die Kamera des Smartphones oder die Abwicklung des Datenaustausches mit Servern (Backends) in der Cloud.

Diese Möglichkeiten der Entwicklung benötigt man sukzessive, wenn man eine mobile App bis zu der produktiven Verwendung erstellen möchte. Ebenso wird der Entwickler dabei unterstützt, direkt aus der Entwicklungsumgebung heraus die App-Packages zu generieren, welche man für die Verteilung der Apps über den Google- bzw. Apple Store benötigt.

Fazit und Ausblick

Apps sind zwar nur Applikationen für die „kleinen“ Geräte in der Hosentasche, ihre Programmierung macht aber nicht minder Arbeit. Mit einem passenden Werkzeug kann man sich den Aufwand ersparen und simultan für Android und iOS aus einem gemeinsamen Quellcode entwickeln. Mit Delphi wird ein intuitiver Ansatz geboten, welcher vielen Elektronikern vertraut vorkommen könnte (Visual Basic, Windows Forms usw.). Die Oberfläche wird direkt und recht schnell im grafischen Editor entworfen. Das Vorgehen bietet sich ebenso dann an, wenn man eine Anwendung für Windows oder Linux schreiben muss. Man kann auf diese Weise den Quellcode mehrfach verwenden. Mit einem Klick schaltet man zwischen den Plattformen um. Die Entwicklungsumgebung erzeugt die notwendigen App-Packages automatisiert. Im Idealfall braucht man keine weiteren Anpassungen vornehmen. ◀

200265-01

Sie haben Fragen oder Kommentare?

Gerne können Sie sich an den Autor des Artikels wenden unter der E-Mail-Adresse v.krypczyk@larinet.com.



PASSENDE PRODUKTE

- > **Buch „Android App Development for Electronics Designers“**
www.elektor.de/android-app-development-for-electronics-designers
- > **Buch „C# Programming for Windows and Android“**
www.elektor.de/c-programming-for-windows-and-android

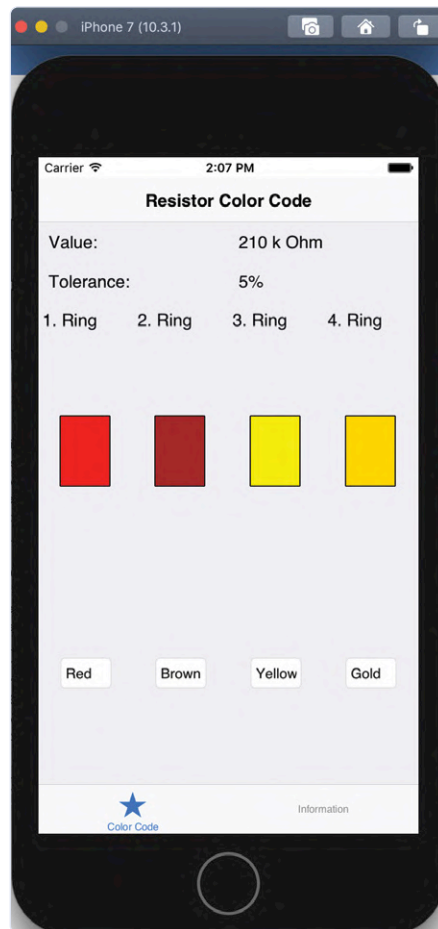


Bild 12. Die gestartete App – noch ohne Funktion – im Simulator (iOS).

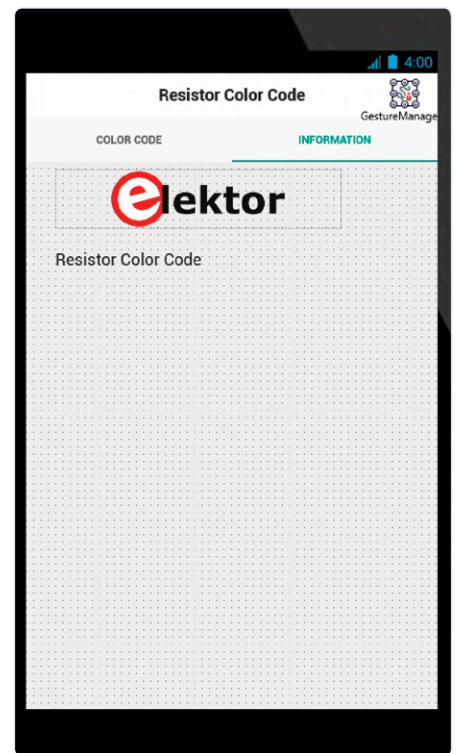


Bild 13. Die Registerkarte Information.

Ein Beitrag von

Autor: **Dr. Veikko Krypczyk**

Redaktion: **Jens Nickel**

Layout: **Giel Dols**

WEBLINKS

- [1] www.embarcadero.com/products/delphi/starter/free-download
- [2] www.macincloud.com/
- [3] http://docwiki.embarcadero.com/RADStudio/Rio/en/Android_Mobile_Application_Development
- [4] http://docwiki.embarcadero.com/RADStudio/Rio/en/IOS_Mobile_Application_Development
- [5] **Projektseite zu diesem Beitrag:** www.elektormagazine.de/200265-01

Aus dem Leben gegriffen

Der Zusammenprall von Alphas, Betas und Gammas

Von **Ilse Joostens** (Belgien)

Vor langer Zeit befand sich der Französischlehrer der technischen Hochschule, ein begeisterter Museumsbesucher und Opernliebhaber, an einem nieselnden Nachmittag im Klassenzimmer, als er über eine Ausstellung der Werke des surrealistischen belgischen Malers René Magritte in Brüssel berichtete. Eines seiner berühmtesten Gemälde, sagte er, zeigt eine Pfeife mit der Bildunterschrift „Ceci n'est pas une pipe“. „Aber das verstehen Sie nicht“, setzte er seine Argumentation trocken fort und fuhr dann mit der Lektion fort. Das machte (nicht sehr subtil) deutlich, dass er tatsächlich auf uns herabsah - schließlich waren wir dumme Wissenschaftsstudenten und vor allem Kulturbarbaren, denen all diese Schönheit nicht zuteil geworden war...

Das Wasserfallsystem

Trotz der zahlreichen Bildungsreformen und der jüngsten Konzentration auf die MINT-Fächer ist die belgische Sekundarschulbildung traditionell der Schauplatz des besseren Wettrennens um Abfälle. Wenn Sie sich zu der Zeit, als ich etwa zwölf Jahre alt war, zu den besseren Schülern in der Grundschule zählen konnten, dann war es selbstverständlich, dass Sie in die allgemeine Sekundarstufe wechseln würden. Die Mehrheit meiner Klassenkameraden ging in diese Richtung und es waren vor allem die leistungsschwachen Schüler, sozusagen die Dummköpfe, die in die technische und berufliche Sekundarausbildung gingen. Aber dann begann das Trash-Rennen erst richtig. Es war schon schmerzhaft, wenn man nicht mehr gut zurechtkam und aus dem Lateinischen oder den



modernen Sprachen in die Wirtschaftswissenschaft abspringen musste, aber es wurde völlig katastrophal, wenn man in die technische Ausbildung verbannt wurde. Glücklicherweise blieb mir dieses grausame Schicksal erspart - veni, vidi, vici!

Dieses „Wasserfallsystem“ setzt einen Teufelskreis fort, weil der Zustrom weniger guter Schüler das Bildungsniveau in der technischen Ausbildung senkt, wodurch die allgemeine Wertschätzung für Technik sinkt und die Eltern ihre begabteren Kinder eher in die Allgemeinbildung schicken, obwohl sie sich für Technik interessieren. Im Gegenzug wandern die Abbrecher der technischen Ausbildung in die Berufsausbildung ab. Es ist davon auszugehen, dass Schüler einer technischen Fachrichtung geringere Erfolgchancen in der Hochschulausbildung haben, was bedeutet, dass vor allem allgemein gebildete Menschen, die nicht durch praktische technische Vorkenntnisse behindert sind, in höhere technische Studiengänge an Hochschulen und Universitäten eintreten. Dort wurde uns ein Berg von Theorie eingetrichtert. Abgesehen von all den ganzen Doppel-, Dreifach- und Kreisintegralen, Differentialgleichungen und anderen Unbekannten konnten wir zwar auch eine Verteilerdose berechnen, aber wir haben so etwas nie in seinem natürlichen Lebensraum zu sehen bekommen. Und nein, Magnefix ist keine Figur aus einem Asterix-Album! Trotz meiner hohen Erwartungen waren dies die langweiligsten Jahre meines Lebens.

Mehr als Problemlöser: die Techies

Es ist ein schöner Moment in einem Leben, wenn man cum laude abschließt oder sine und sein brandneues Diplom in einer feierlichen Proklamation zur Schau stellt. Die sehr harte Realität ist jedoch, dass man, wenn man kein Glück hat, am



Tag nach der Proklamation nicht mehr ein erfolgreicher Student, sondern nur noch ein arbeitsloser Verlierer ist.

Die Geschäftswelt, die sich heutzutage mehr und mehr um Buchhalter, Ökonomen, Opportunisten, Berater und andere Wirtschaftskriminelle zu drehen scheint, ist auch nicht wirklich freundlich zu den technisch orientierten Mitmenschen. Wir mögen uns an den schwammigen und teuren Worten oder den verächtlichen Ausdrücken weiden, die bereits von der Führungsspitze verwendet werden, aber Aussagen wie „das wird das Problem lösen“ oder „das machen unsere Techies schon“ zeugen nicht von viel Respekt [1]. Diese Unterbewertung von Technikern wird in der britischen Sitcom „The IT Crowd“ des irischen Regisseurs Graham Linehan auf humorvolle Weise thematisiert. Als ehemaliger Systemadministrator in einem gewissen Handelsunternehmen muss ich leider gestehen, dass viel Wahrheit in den verrückten Erfahrungen von Roy Trenneman und Maurice Moss steckt - sehr zu empfehlen!

Das Märchen vom MINT-Mangel

Jahrelang wurden wir mit der Litanei getäuscht, dass es einen Mangel an technischen Profilen und Ingenieuren geben würde [2]. In Belgien stehen diese Positionen seit Jahren auf der Liste der sogenannten Engpassberufe.

Wenn es wirklich einen solchen Mangel gäbe, würde man erwarten, dass im Rahmen der Marktkräfte die Löhne und Leistungen für Techniker entsprechend hoch wären. Nichts könnte weiter von der Wahrheit entfernt sein: Viele technische Jobs in Belgien werden unterdurchschnittlich bezahlt. Mit einem Standard-Neun-bis-Fünf-Bürojob, inklusive Anfertigen von Kopien, Kaffee kochen und Pflanzen gießen, kann man leicht viel mehr verdienen als mit einem Job als Servicetechniker. Und man muss nicht in einen dumpf riechenden Raum kriechen, ohne Tageslicht und mit einer dröhnenden Leuchtstoffröhre über dem Kopf. Auch die besseren technischen Arbeitsstellen kann es treffen. Ein Bekannter von mir, leitender Elektronik-Hardware-Ingenieur in einem ähnlich gewissen Handelsunternehmen, musste nach jahrelangem treuen Dienst anlässlich einer Mitarbeiterfeier eine gerahmte Urkunde und einen Blumenstrauß als Dank für sein Engagement in Kauf nehmen. Das ist nicht wirklich etwas, worüber man zu Hause stolz berichten kann, gar nicht zu reden von den Boni und den Gehaltserhöhungen, die regelmäßig an Manager und Vertriebsmitarbeiter vergeben wurden und werden.

Ich schließe also mit dem bekannten Sprichwort „Warum ein Ingenieur/Wissenschaftler sein, wenn man sein Chef sein kann?“ ❏

200455-02

Sie haben Fragen oder Kommentare?

Haben Sie Fragen oder Anmerkungen zu diesem Artikel? Senden Sie eine E-Mail an den Autor oder an die Redaktion von Elektor über redactie@elektor.com.

Ein Beitrag von

Text: **Ilse Joostens (Belgien)**

Redaktion: **Eric Bogers**

Übersetzung: **Rolf Gerstendorf**

Layout: **Giel Dols**

WEBLINKS

[1] **Phrasendrescher:** <https://www.webwriting.ch/spiel/heissluft.php>

[2] **Das Märchen vom Ingenieursmangel:** <http://www.beteronderwijsnederland.nl/blogs/2009/02/het-sprookje-van-het-beta-tekort/>



Wie Infrastruktur unsere Gesellschaft formt

Von Tessel Renzenbrink

„Eine Wissensgesellschaft, die auf privatisiertem Wissen basiert, ist natürlich völliger Unsinn“, sagt Dr. Niels ten Oever, der mit der Universität Amsterdam und der Texas A&M University verbunden ist. Er erforscht die Infrastruktur von Informationsnetzwerken und deren Auswirkungen auf die Gesellschaft. In diesem Interview spricht er über die Einführung von 5G, dem Mobilfunknetz der fünften Generation.

„Normalerweise denken wir nicht über die Infrastruktur nach“, sagt Ten Oever. „Erst wenn sie zusammenbricht - der Strom fällt aus oder eine Straße wird gesperrt - fällt sie auf. Dennoch sollten wir nicht unterschätzen, wie entscheidend die Gestaltung der Infrastruktur ist. Nehmen Sie zum Beispiel das Straßennetz. Wie das angelegt wird, hat Einfluss auf Ihr tägliches Leben. Eine vierspurige Straße, flankiert von schmalen Bürgersteigen, fördert eine andere Nutzung des öffentlichen Raums als breite Geh- und Radwege. Dasselbe gilt für Informationsinfrastrukturen wie 5G. Die Architektur des Netzwerks bringt Möglichkeiten und Einschränkungen mit sich. Die Frage ist deshalb: Wofür soll ein Netzwerk optimiert werden?“

Virtualisierung des Netzwerks

„Mit der Einführung von 5G wird eine neue Infrastruktur geschaffen“, fährt Ten Oever fort. „Es wird auf anderen Frequenzen betrieben als 4G. Es wird also neue Antennen, aber auch neue Router und Switches geben. Dies bedeutet auch, dass neue *Computingparadigmen* eingeführt werden können. Denn diese Router und Switches sind eher *Universal-Computer*, die also viel allgemeineren Zwecken dienen: Geräte, die einfach auf Linux laufen können. Dies steht im Gegensatz zur aktuellen Generation von Netzwerkgeräten mit anwendungsorientierter Hardware und spezifischer Firmware. Weil man viel mehr mit Software löst, können Netzwerkfunktionen virtualisiert werden.“ Die Netzwerkvirtualisierung ermöglicht es, das Netzwerk immer intelligenter zu machen. Dies bedeutet den Abschied von einem der zentralen Gestaltungsprinzipien des frühen Internets: dem End-to-End-Prinzip. End-to-End bedeutet, dass jeder Computer im Netzwerk eine

Verbindung zu jedem anderen angeschlossenen Computer herstellen kann. Die Idee war, dass das Netzwerk selbst so wenig wie möglich beiträgt, sondern nur aus „dummen Pipelines“ besteht, die Daten transportieren. Die Intelligenz - Rechenleistung und Datenspeicherung - findet an den Endpunkten des Netzes statt: auf den Computern der Nutzer. Der Trend der Verlagerung von Intelligenz von den Endgeräten in das Netzwerk selbst findet aber schon seit Jahren statt. Denken Sie an das Cloud-Computing, bei dem Datenspeicherung und Softwareanwendungen auf Servern Dritter ausgeführt werden. Die Einführung von 5G ermöglicht es, den Trend zu mehr Intelligenz im Netzwerk fortzusetzen, zum Beispiel beim *Information Centric Networking*, bei dem die Daten im Netzwerk selbst gespeichert werden. Ten Oever gibt ein Beispiel dafür, wie das aussehen könnte.

Game of Thrones in der Badewanne

„Es ist bereits jetzt so, dass bestimmte Anwendungen wie Netflix auf der Grundlage Ihres Verhaltens Empfehlungen geben. Aber es kann auch vorhergesagt werden, wo Sie Netflix nutzen möchten. Ich zum Beispiel sehe mir Game of Thrones nämlich immer in der Badewanne an. Und ich höre meine Podcasts immer beim Radfahren. Der Podcast kann in dem intelligenten Laternenpfahl, an dem ich immer vorbeifahre, zwischengespeichert werden. Wenn man auf der Arbeit ankommt, ist der Computer bereits eingeschaltet und die bevorzugten Programme laufen. Super praktisch. Aber wie behält man die Kontrolle darüber? *Will* man es kontrollieren? Woher weiß man, wer es tut und auf welcher Grundlage?“

„So weit sind wir noch nicht. Aber die Existenzbedingung, nämlich die General-purpose-Infrastruktur, wird mit 5G geschaffen“, fährt Ten

Niels ten Oever untersucht die Informationsinfrastruktur (Foto mit freundlicher Genehmigung von Niels ten Oever).



Oever fort. „Dadurch wird es möglich sein, ein solches Netzwerk immer weiter zu optimieren. Das ist der Traum eines jeden Netzbetreibers, denn so kann das Netzwerk so effizient wie möglich eingerichtet werden. Die Effizienz ist ein Parameter, der optimiert werden kann. Das bedeutet aber auch, dass eine Kontrollmöglichkeit für den Verbraucher verschwindet. Denn es wird immer unsichtbarer, welches Gerät womit verbunden ist und welche Informationen es sendet. Sie können sich vorstellen, wie dies zu stärkeren Filterblasen führt. Oder zur Extraktion von noch mehr den Datenschutz berührenden Daten und selbst zur Entwicklung von persönlichen Zensurprofilen, so dass es einer bestimmten Person nicht mehr erlaubt ist, einen gewissen Inhalt zu sehen oder mit jener anderen Person in Kontakt zu kommen.“

5G-Standards und Patente drängen

Ten Oever untersucht, wie die Entscheidungsprozesse bei der Standardisierung von 5G funktionieren. 5G ist ein Bündel verschiedener Technologien, das in drei Phasen standardisiert werden soll. Der Abschluss der zweiten Phase ist für 2020 geplant. Die Spezifikationen für 5G werden vom 3GPP-Verband definiert, einem Zusammenschluss von sechs Standardisierungsorganisationen, darunter das europäische ETSI. Die endgültigen Normen werden schließlich in der Internationalen Fernmeldeunion (ITU) verabschiedet, in der die Mitgliedstaaten Stimmrecht haben.

Ten Oever: „Es ist ziemlich schwer einzuschätzen, warum bestimmte Entscheidungen getroffen werden. Aber einer der Gründe für die Entwicklung bestimmter Technologien sind Patente. Dies ist etwas, das in der Internet-Welt eine viel geringere Rolle spielt. Die Unternehmen dort haben zwar Patente, aber sie nutzen sie defensiv. Eigentlich ist es eine Art Waffenstillstand: Wir haben Patente, ihr habt Patente, aber wir benutzen sie nicht. Erst wenn ihr anfangt, zahlen wir es euch heim. Aber in der Mobilfunkbranche funktioniert es anders. Im 3GPP basiert vieles auf proprietärer Hard- und Software. Das ist auch das, was am meisten Geld einbringt. Ein Unternehmen wie Qualcomm verdient mehr mit Patenten als mit dem Verkauf von Hardware. Die Unternehmen versuchen daher, ihre patentierte Technologie bei der Entwicklung der Normen voranzutreiben. Wir erwarten in den kommenden Jahren eine gewaltige Explosion neuer Geräte. Nehmen wir an, ein Unternehmen erhält 10 Cent pro Gerät für Lizenzen. Das ist eine Menge Geld, und das Unternehmen wird nie wieder etwas dafür tun müssen.“

„Wir haben nicht die leiseste Ahnung, wie diese Netzwerke funktionieren“

Diese Patente machen die Funktionsweise von 5G undurchsichtiger als das traditionelle Internet. Ten Oever: „Sie und ich können lernen, wie das Internet funktioniert, weil das Wissen darüber öffentlich ist. Wir können zu Hause ein Mini-Internet aufbauen. Aber wir können kein Mini-5G-Netzwerk einrichten, weil ein Großteil des Wissens hinter Patenten und Lizenzen im privaten Sektor liegt“. Der Mangel

an Transparenz stehe auch der Forschung und Entwicklung im Wege, sagt Ten Oever. „Jetzt arbeiten wir immer noch mit Protokollen. Dies sind relativ einfache Regeln, aber das wird bald alles algorithmisch optimiert werden. Das wirft eine Menge komplexer Fragen auf. Es wäre gut, wenn viele Leute zusammen darüber nachdächten. Es müssten Universitäten und Forschungsinstitute einbezogen und der Markt für Neueinsteiger mit neuen Ideen geöffnet werden. Aber das können wir jetzt nicht tun, weil wir nicht die leiseste Ahnung haben, wie diese Netzwerke funktionieren. Patente sind eine Barriere, die den Austausch von Wissen verhindert.“

„Statt die Interessen etablierter Unternehmen zu schützen, sollten wir uns für offene Hardware und offene Software einsetzen“, sagt Ten Oever. „Denn dann schaffen wir ein offenes digitales Ökosystem, in dem Innovation stattfinden kann. Und hier haben Regierungen eine Rolle zu spielen. Sie könnten Forderungen dieser Art stellen. Bei Infrastrukturen wie der Wasser- und Stromversorgung haben wir viele sehr strenge Richtlinien, die Abläufe, Kontrollmaßnahmen und Anforderungen genau festlegen. Bei digitalen Infrastrukturen ist dies viel weniger der Fall. Aber in dem Moment, in dem eine wichtige Infrastruktur entsteht, ergibt sich auch eine damit zusammenhängende Verantwortung. Dieses Bewusstsein ist in der Internet- und Telekommunikationswelt noch zu selten. Und die Regierung ist unzureichend informiert und verkennet die Notwendigkeit, solche Anforderungen zu stellen.“

„Das Menschenrecht auf Wissenschaft sollte auch unsere Informationsinfrastruktur umfassen“, so Ten Oever abschließend. „Das hat Vorrang vor dem Schutz von Unternehmensinteressen. Wenn man sagt, dass wir eine Informationsgesellschaft oder eine Wissensgesellschaft sind, dann muss dieses Wissen verfügbar sein. Eine Wissensgesellschaft, die auf privatisiertem Wissen basiert, ist natürlich völliger Unsinn.“

200468-02

Sie haben Fragen oder Kommentare?

Haben Sie Fragen oder Anmerkungen zu diesem Artikel? Senden Sie eine E-Mail an den Autor oder an die Elektor-Redaktion über redactie@elektor.com.

Ein Beitrag von

Text: **Tessel Renzenbrink** (Niederlande)

Redaktion: **Eric Bogers**

Layout: **Giel Dols**

Übersetzung:

Katharina Sophia Gerstendorf

Praktisches ESP32-Multitasking (5)

Task-Ereignisbenachrichtigung

Von **Warren Gay** (Kanada)

Obwohl FreeRTOS Queue-, Semaphore- und Mutex-Funktionen für die Synchronisierung bietet, können diese bei einfachen Anforderungen manchmal übertrieben kompliziert sein. Die FreeRTOS-Version V8.2.0 führte das Konzept der *direct to task event notifications* (Task-Ereignisbenachrichtigungen) ein, wodurch dem Programmierer eine leichtgewichtige Methode zur Synchronisierung von Tasks zur Verfügung steht.

Jeder Task enthält einen internen 32-Bit-Ereignisbenachrichtigungswert, der bei der Erstellung des Tasks auf Null gesetzt wird. Da dieser Wert in den Task eingebaut ist, wird kein zusätzliches RAM benötigt. Diese Vorgehensweise erfordert also wesentlich weniger Speicher als beispielsweise die Verwendung von Semaphore-Objekten.

Einschränkungen

Es gibt jedoch einige Einschränkungen bei der Verwendung direkter Task-Benachrichtigungen:

- Sie können keine Benachrichtigung an eine ISR senden, da eine ISR kein Task ist. Eine ISR ihrerseits kann jedoch einen Task benachrichtigen.
- Es kann nur ein Task durch einen Task-Benachrichtigungs-Call angesprochen werden. Um mehrere Tasks zu benachrichtigen, müssen Event-Gruppen verwendet werden.
- Benachrichtigungsereignisse können nicht wie Queues gepuffert werden.
- Der benachrichtigende Task (oder ISR) blockiert seine Ausführung nicht, um auf den Empfang des Ereignisses durch den empfangenden Task zu warten. Die Ereignisbenachrichtigung wird lediglich verschickt und der Call kehrt sofort und ungehindert zurück.

Wenn eine dieser Einschränkungen mit Ihren Anforderungen in Konflikt steht, müssen Sie stattdessen einen anderen FreeRTOS-Mechanismus wählen.

Warten auf ein Ereignis

Der erste Schritt bei einer Task-Ereignisbenachrichtigung besteht darin, den empfangenden Task zu informieren, auf eine Benachrichtigung zu warten. Mit anderen Worten, der Task muss so eingerichtet werden, dass er pausiert, bis ein Ereignis ausgelöst wird. Dies wird durch eine der folgenden Funktionen realisiert:

- `ulTaskNotifyTake()`
- `xTaskNotifyWait()`

Die Funktion `ulTaskNotifyTake()` ist die einfachere der beiden Funktionen und wird hier in diesem Artikel untersucht. Die fortgeschrittenere Funktion `xTaskNotifyWait()` wird im Elektor-Buch „FreeRTOS für ESP32-Arduino“ ausführlich behandelt.

ulTaskNotifyTake()

Der eine Benachrichtigung empfangende Task blockiert seine Ausführung durch den Aufruf von `ulTaskNotifyTake()`. Dieser Call erfordert zwei Argumente:

```
uint32_t ulTaskNotifyTake(
    BaseType_t xClearCountOnExit, // pdFALSE or pdTRUE
    TickType_t xTicksToWait
);
```

Das erste Argument arbeitet mit dem 32-Bit-Task-Benachrichtigungswort (auch Task-Event-Word) und ist in **Tabelle 1** definiert. Der zweite Parameter ist die Schwelle für die Zeitüberschreitung, den bekannten *Timeout* in Ticks (verwenden Sie das Makro `pdMAX_DELAY`, wenn Sie kein Timeout wünschen). Bei allen Aufrufen von `ulTaskNotifyTake()` wird die Ausführung des aufrufenden Tasks blockiert, solange das Task-Event-Word den Wert 0 besitzt. Wenn der Wert ungleich Null wird, bestimmt das Argument `xClearCountOnExit`, wie das Task-Ereigniswort vor der Rückkehr aktualisiert wird.

Tabelle 1. Die Bedeutung der Argumente `xClearCountOnExit` für `ulTaskNotifyTake()`.

xClearCountOnExit Wert	Auswirkung auf das 32-Bit-Task-Notification-Word
<code>pdFALSE</code>	Wert dekrementieren, weitere Ausführung blockieren, wenn der Wert vor der Dekrementierung Null war, bevor der Wert vor der Dekrementierung zurückgegeben wird.
<code>pdTRUE</code>	Wert auf 0 zurücksetzen, weitere Ausführung blockieren, wenn der Wert bereits Null war, bevor der Wert vor der Dekrementierung zurückgegeben wird.

Der vom Funktionsaufruf zurückgegebene Wert ist der Wert des Event-Notification-Words, bevor es gelöscht oder dekrementiert wurde. Wenn ein *Timeout* auftritt, ist der zurückgegebene Wert ebenfalls Null.

Binäre Benachrichtigung

Wenn der Wert des Arguments `xClearCountOnExit` gleich `pdTRUE` ist, funktioniert der Aufruf von `ulTaskNotifyTake()` wie die binäre

semaphore *take*-Operation. Wenn das Task-Notify-Ereigniswort bereits ungleich Null ist, kehrt der Aufruf sofort zurück und setzt das Ereigniswort auf Null (sein früherer, von Null verschiedener Wert wird jedoch zurückgegeben). Wenn das Task-Benachrichtigungswort zum Zeitpunkt des Aufrufs Null war, wird die weitere Taskausführung blockiert, bis der Task (vorbehaltlich einer Zeitüberschreitung) benachrichtigt wird. Wenn ein Timeout eintritt, ist der Rückgabewert immer Null (was den Wert des Ereigniswortes zum Zeitpunkt der Rückkehr widerspiegelt).

In diesem Modus arbeitet der Aufruf wie ein binäres Semaphore. Wenn der Task zwei Mal benachrichtigt werden sollte, bevor der empfangende Task `ulTaskNotifyTake()` aufruft, wird der Rückgabewert gleich 2 sein. Das Task-Ereigniswort wird jedoch bei der Rückkehr auf Null gesetzt. Damit steht in diesem speziellen Fall also nur ein Benachrichtigungsereignis zur Verfügung, aber die tatsächliche Anzahl der Benachrichtigungen lässt sich leicht durch den Rückgabewert ermitteln.

Zählende Benachrichtigung

Wenn mehrere Task-Benachrichtigungen mehrere Task-Wakeups verursachen *müssen*, dann sollte das `xClearCountOnExit`-Argument mit dem Wert `pdFALSE` versehen werden. In einem solchen Anwendungsfall blockiert der empfangende Task, wenn der Wert des Task-Ereignisses Null ist (genau wie zuvor). Jede Benachrichtigung des Tasks inkrementiert den Task-Ereigniswert, aber er wird bei jedem Aufruf von `ulTaskNotifyTake()` nur um eins dekrementiert. Wenn beispielsweise vier Benachrichtigungen erfolgen, bevor der empfangende Task `ulTaskNotifyTake()` aufruft, kehrt dieser Aufruf unmittelbar ohne Blockierung zurück, bis der Wert des Task-Ereignisses wieder auf Null dekrementiert wurde. Mit anderen Worten, der empfan-

gende Task muss vier Mal `ulTaskNotifyTake()` aufrufen, bevor das Ereigniswort wieder auf Null dekrementiert ist.

Give Notify

Während der empfangende Task die Funktion `ulTaskNotifyTake()` verwendet, wird die Benachrichtigung durch `xTaskNotifyGive()` bereitgestellt:

```
BaseType_t xTaskNotifyGive(TaskHandle_t xTaskToNotify);
```

Diese Funktion erfordert nur, dass der Handle des Tasks benachrichtigt wird und gibt immer `pdTRUE` zurück (das FreeRTOS-Handbuch gibt an, dass `xTaskNotifyGive()` als Makro definiert ist).

Demonstrationsprogramm

Für eine sehr einfache Demonstration sorgt der Code in **Listing 1** [1]. Dieses Programm zeigt im Serial Monitor den zurückgegebenen Wert des gemeldeten Ereigniswortes. In **Bild 1** ist der Schaltplan für den

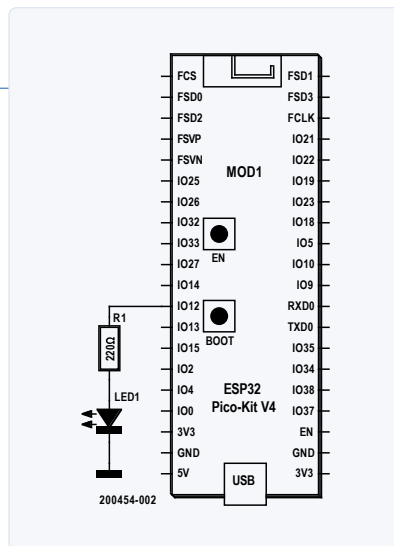


Bild 1. Schaltung für den `tasknfy.ino`-Code.

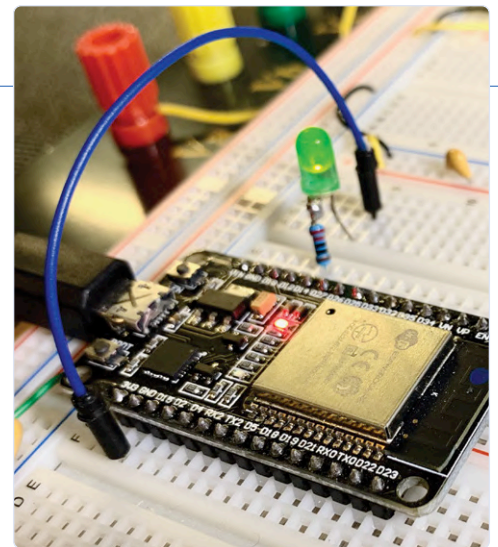


Bild 2. Aufbau der Demonstrationsschaltung mit einer LED an GPIO 12.

Listing 1. Das Demonstrationsprogramm `tasknfy1.ino` für `xTaskNotifyGive()` und `ulTaskNotifyTake()`.

```
0001: // tasknfy1.ino
0002:
0003: #define GPIO_LED      12
0004:
0005: static TaskHandle_t htask1;
0006:
0007: static void task1(void *arg) {
0008:     uint32_t rv;
0009:
0010:     for (;;) {
0011:         rv = ulTaskNotifyTake(pdTRUE,
portMAX_DELAY);
0012:         digitalWrite(GPIO_LED,
digitalRead(GPIO_LED)^HIGH);
0013:         printf("Task notified: rv=%u\n",
unsigned(rv));
0014:     }
0015: }
0016:
0017: void setup() {
0018:     int app_cpu = 0;
0019:     BaseType_t rc;
0020:
0021:     app_cpu = xPortGetCoreID();
0022:     pinMode(GPIO_LED, OUTPUT);
0023:     digitalWrite(GPIO_LED, LOW);
0024:
0025:     delay(2000); // Allow USB to connect
0026:     printf("tasknfy1.ino:\n");
0027:
0028:     rc = xTaskCreatePinnedToCore(
0029:         task1,          // Task function
0030:         "task1",        // Name
0031:         3000,           // Stack size
0032:         nullptr,        // Parameters
0033:         1,              // Priority
0034:         &htask1,        // handle
0035:         app_cpu         // CPU
0036:     );
0037:     assert(rc == pdPASS);
0038: }
0039:
0040: void loop() {
0041:     delay(1000);
0042:     xTaskNotifyGive(htask1);
0043: }
```


Anschluss einer optionalen LED zu sehen; **Bild 2** zeigt den Aufbau der Schaltung auf dem Breadboard. Es kann so gut wie jedes ESP32-Dev-Board verwendet werden, bei dem GPIO 12 verfügbar ist.

Das Programm *tasknfy1.ino* verwendet die vom Arduino bereitgestellte `loop`-Task, um `task1()` (Zeile 42) wiederholt in Intervallen von einer Sekunde zu benachrichtigen. Die `task1()`-Funktion blockiert dann (Zeile 11) bis zur Benachrichtigung. Danach schaltet sie die LED um (Zeile 12). Die Funktion `task1()` wird mit Hilfe des Handles `htask1` (Zeile 42) benachrichtigt, der in der Funktion `setup()` in Zeile 34 erstellt wurde.

```
0040: void loop() {
0041:   delay(1000);
0042:   xTaskNotifyGive(htask1);
0043: }
```

Der `task1`-Code ist fast genauso einfach und arbeitet in einer Endlosschleife. Er blockiert seine Ausführung beim Aufruf von `ulTaskNotifyTake()` in Zeile 11, um auf ein Notify-Ereignis zu warten:

```
0007: static void task1(void *arg) {
0008:   uint32_t rv;
0009:
0010:   for (;;) {
0011:     rv = ulTaskNotifyTake(pdTRUE, portMAX_DELAY);
0012:     digitalWrite(GPIO_LED,
0013:                  digitalRead(GPIO_LED)^HIGH);
0013:     printf("Task notified: rv=%u\n",
0014:           unsigned(rv));
0014:   }
0015: }
```

Die weitere Ausführung von `task1` bleibt blockiert, bis die Task-Ereignisbenachrichtigung eintrifft. Der zugewiesene Wert `rv` (Zeile 11) ist der Wert des Ereignisbenachrichtigungswortes, bevor es aufgrund des `pdTRUE`-Arguments gelöscht wird. Das sieht alles sehr sauber und einfach aus - es sind keine anderen Semaphor-Objekte beteiligt und somit keine zusätzlichen Handles.

Die resultierende Ausgabe im seriellen Monitor sollte wie folgt aussehen:

```
tasknfy1.ino:
Task notified: rv=1
Task notified: rv=1
...
```

Wenn Sie einen Task aus einer ISR benachrichtigen möchten, müssen Sie stattdessen die Funktion `xTaskNotifyGiveFromISR()` verwenden. Diese „FromISR“-Suffix-Konvention von FreeRTOS ermöglicht eine sichere Operation aus einer ISR heraus.

```
void vTaskNotifyGiveFromISR(
    TaskHandle_t xTaskToNotify,
    BaseType_t *pxHigherPriorityTaskWoken
);
```

Das zweite Argument ist die Adresse des ISR-Wakeup-Flags, das angibt, ob der Scheduler aufgerufen werden soll oder nicht (dann `NULL/nullptr`). Diese Task-Benachrichtigungsfunktion stellt eine sehr bequeme Möglichkeit für eine ISR-Routine dar, einen Task zu benachrichtigen.

Erweiterte Benachrichtigung

In diesem Artikel haben wir nur die beiden einfachsten Funktionen `xTaskNotifyGive()` (oder `xTaskNotifyGiveFromISR()`) und `ulTaskNotifyTake()` besprochen.

Es sind zusätzliche, leistungsfähigere Funktionen verfügbar, darunter:

- > `xTaskNotify()` (or `xTaskNotifyFromISR()`)
- > `xTaskNotifyAndQuery()` (or `xTaskNotifyAndQueryFromISR()`)
- > `xTaskNotifyWait()`
- > `xTaskNotifyStateClear()`
- > `xTaskNotifyValueClear()`

Diese Funktionen arbeiten auf ähnliche Weise, unterstützen jedoch die Ereignisverarbeitung auf Bit-Ebene. Sie bieten auch selektivere Formen der Ereignisprüfung und -löschung. Über diese Funktionen gibt das FreeRTOS-Referenzhandbuch ([2] oder online [3]) Auskunft.

Fazit

Die Task-Benachrichtigungs-API von FreeRTOS bietet dem Programmentwickler einen schlanken Mechanismus zur Ereignisbenachrichtigung. Dies ist besonders nützlich zur Implementierung einer Task-Benachrichtigung von ISRs, da es die Interruptverarbeitung kurz hält, während die „harte“ Arbeit im benachrichtigten Task ausgeführt werden kann. Selbst wenn keine Interrupts verwendet werden, spart diese Methode RAM und vereinfacht die Anwendung, da keine zusätzlichen Synchronisationsobjekte wie Semaphoren benötigt werden. ◀

200454-02

Ein Beitrag von

Idee und Text: **Warren Gay**
Illustrationen: **Patrick Wielders**

Redaktion: **Stuart Cording**

Übersetzung: **Rolf Gerstendorf**

Layout: **Giel Dols**

WEBLINKS

- [1] **Code für tasknfy.ino:** https://github.com/ve3wwg/FreeRTOS_for_ESP32/blob/master/tasknfy1/tasknfy1.ino
- [2] **FreeRTOS-Dokumentation:** https://www.freertos.org/Documentation/RTOS_book.html
- [3] **FreeRTOS Dokumentation zur Task-Benachrichtigung:** <https://www.freertos.org/RTOS-task-notification-API.html>

50

Jahre Elektor

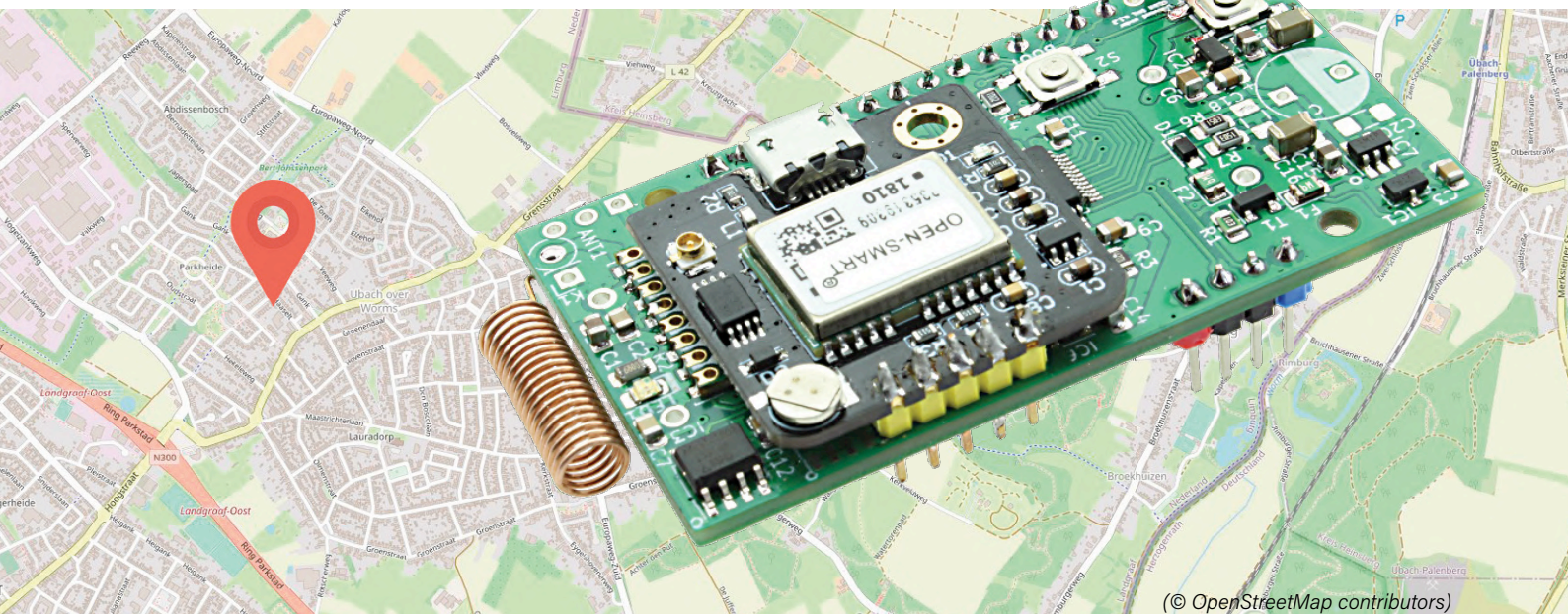
Seit **Mai 1970** gibt Elektor eine deutsche Ausgabe heraus. In diesem Jahr feiern wir **fünf Jahrzehnte** innovativer Elektronik-Projekte, informativer Ingenieur-Tutorials und Fachartikel zu einer Vielzahl von technischen Themen. Schließen Sie sich uns an, um weitere **50 Jahre Elektronik** zu entwerfen, zu teilen und zu verkaufen!

Elektor-Archiv 1970-2019 (USB-Stick)

BESTELLEN SIE NOCH HEUTE!

www.elektor.de/1970-2019

elektor 50.
design > share > sell



○

beutung der Zutaten, was in diesem Fall ein paar Wochen Zeit und Blood, Sweat & Tears (für die Jüngeren: das ist eine sehr bekannte Jazzrockband gewesen) beanspruchte. Im Laufe der Zubereitung wurde aus einem ursprünglich geplanten simplen LoRa-GPS-Tracker der universelle Elektor-LoRa-Knoten (**Bild 1**) mit vielen Einsatzmöglichkeiten, wovon der GPS-Tracker nur noch eine ist. Der Elektor-LoRa-Knoten und alle seine Ableger in Vergangenheit, Gegenwart und Zukunft sind „offen“, sowohl was die Software als auch die Hardware angeht.

Da in der März/April-Ausgabe von Elektor der Elektor-LoRa-Knoten [3] schon so ausführlich behandelt und auch der Einstieg in The Things Network [4] genauestens beschrieben wurde, sollten Sie sich dieses Elektor-Heft

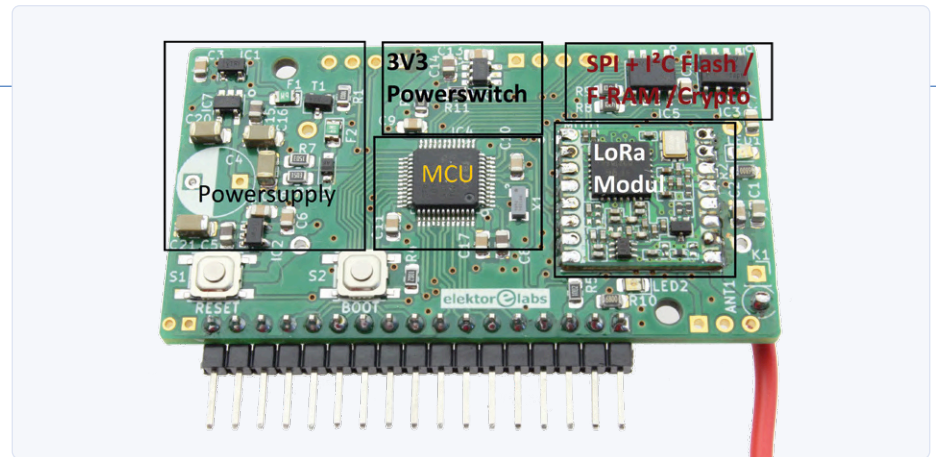


Bild 1. Die bestückte Platine des Elektor-LoRa-Knotens.

jetzt direkt griffbereit auf den Schreibtisch legen. Bei der Beschreibung der Hardware in **Bild 2** können wir uns so auf das Wesentliche konzentrieren und den zur Verfügung

stehenden Raum nutzen, um näher auf das GPS-Modul, die GPS-Software, die Einbindung ins und dem Transport der Daten über das The Things Network sowie auf MQTT

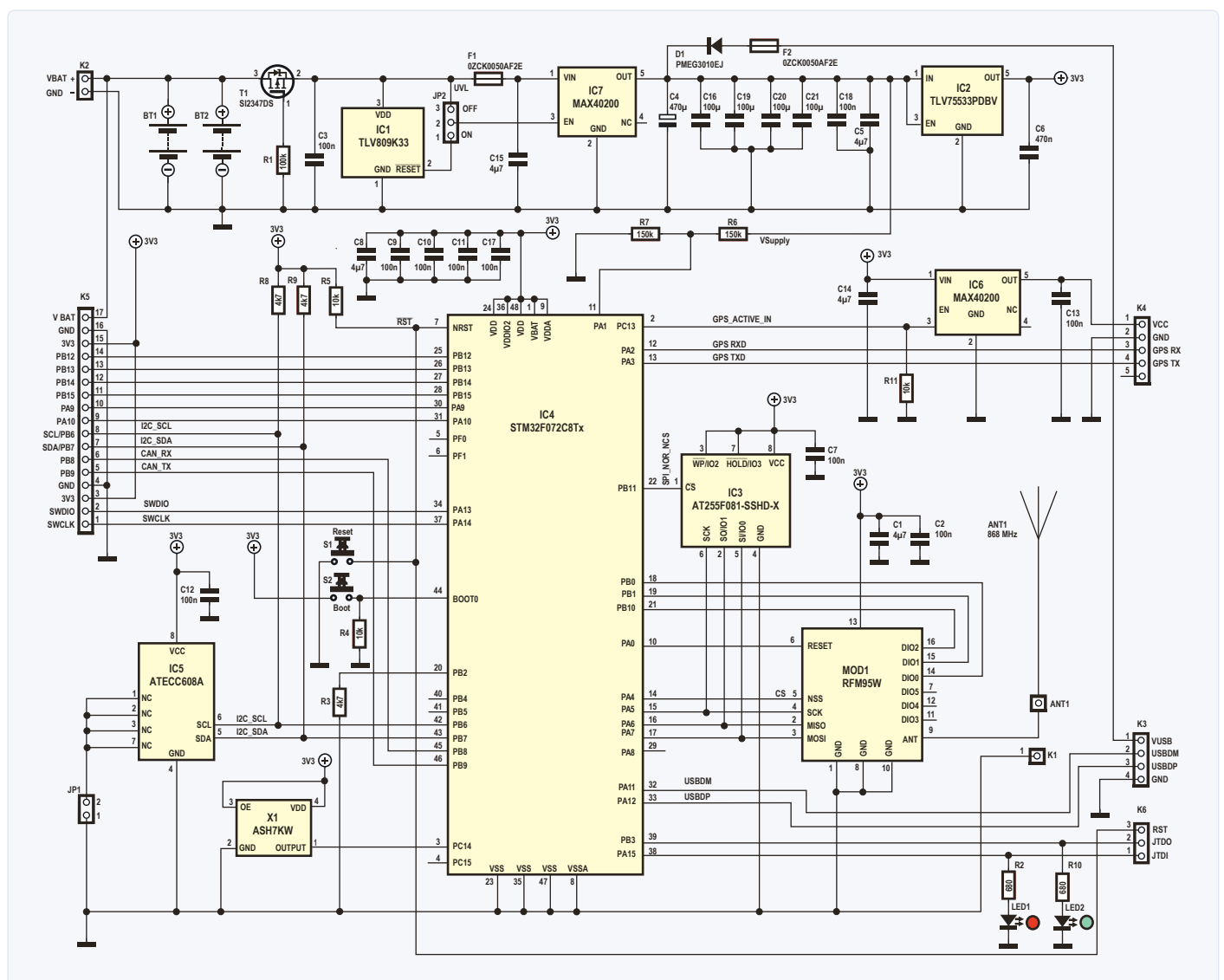


Bild 2. Der Schaltplan des LoRa-Knotens.

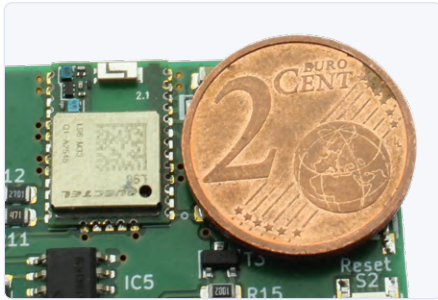


Bild 3. Das winzige GPS-Modul L96.

und Node-RED bis zur Darstellung auf einer Landkarte einzugehen.

Das Gehäuse

Als Gehäuse, in dem der Akku, die Basisplatine und ein GPS-Modul Platz finden sollen, wurde ein Typ aus der 1551-Reihe von Hammond mit Schutzklasse IP54 (staub- und spritzwassergeschützt) ausgewählt, das sich außerhalb des Hauses verwenden lässt.

Der Mikrocontroller

Als Mikrocontroller kommt ein STM32 zum Einsatz, genauer gesagt, ein STM32F072C8T mit 64 kB Flash und 16 kB RAM. Innerhalb der STM32-Familie sind viele Chips im LQFP48-Gehäuse untereinander pin-kompatibel, so dass auf der Platine ohne großen Aufwand auch ein Controller mit mehr Flash oder einem noch geringeren Energiebedarf eingesetzt werden kann. Neben der klassischen Entwicklung mit C/C++ kann der Controller dank des STM32duino-Projekts auch über das Arduino-Framework mit Software versehen werden. Durch den (nicht überschreibbaren) Bootloader in der MCU

reicht ein einfacher USB-Seriell-Wandler, um neue Firmware auf den Chip zu laden.

Der LoRa-Transmitter

Auf der Platine befindet sich auch der LoRa-Transceiver, ein RFM95W, der im 868-MHz-ISM-Band arbeitet.

Die Energieversorgung

Die Schaltung soll von einer preiswerten, wieder aufladbaren Li-Ionen-Batterie aus der Welt der Drohnen versorgt werden. Im Prinzip würden auch AAA-Lithium-Zellen funktionieren, allerdings müsste man dann ein größeres Gehäuse verwenden.

Wir haben bewusst auf eine Ladeschaltung verzichtet, so dass die Zelle extern von einem geeigneten Ladegerät wieder aufgeladen werden muss. Verzichtet haben wir aber nicht auf eine Unterspannungserkennung IC1, die den Akku vor der Tiefenentladung schützt, und einen Schutz vor einem ungewollten Rückwärtsspeisen aus der Schaltung in Form der „idealen Diode“ IC7, damit Akkus nicht überladen werden, auslaufen oder gar explodieren oder Feuer fangen. Die Schutzschaltung selber wird direkt am Akku durch eine 500-mA-Polyfuse geschützt, so dass ein Kurzschluss nicht direkt zu einem Abbrennen der Leiterbahnen führen kann. Diese Sicherheitsmaßnahmen haben außerdem den Vorteil, dass mit einer weiteren Diode die Schaltung auch direkt aus einer 5-V-Quelle versorgt werden könnte, mit dem Lithium-Akku als Backup.

I²C/SPI-Flash/Crypto-Modul

Optional mit an Board sind ein Flash-Speicher (IC3) und ein Crypto-Modul (IC5), die über I²C/SPI am Controller angeschlossen

sind. Beide ICs sind für den GPS-Tracker nicht erforderlich und können weggelassen werden.

3V3-Power-Switch

Der 3,3-V-Ausgangsregler IC2 hat ausreichend Power, um neben der Elektronik des LoRa-Knotens auch das GPS-Modul mit Energie zu versorgen.

Das GPS-Modul

Etwas mehr Zeit haben Auswahl und Einsatz des GPS-Moduls in Anspruch genommen. Ursprünglich war ein winziges Modul (**Bild 3**) namens Quectel L96 vorgesehen, das direkt auf der Basisplatine angebracht werden sollte. In einer Design Note des L96 war etwas über eine passende Groundplane und die korrekte Positionierung des Moduls darauf zu lesen. In der ersten Platinenversion wurde die Design Note „überlesen“, doch selbst als in der zweiten Platinenversion diese Design Note ernst genommen und das Modul entsprechend behandelt wurde, war der GPS-Empfang durch die winzige Antenne (das Bauteil über der Beschriftung 2.1) so schlecht, dass die Position erst nach längerer Wartezeit ermittelt werden konnte, was natürlich der Energieeffizienz des Moduls ziemlich abträglich ist. Daraus entstand die endgültige Schaltungsversion, bei der der GPS-Empfänger nicht mehr Teil der Hardwarebasis ist, sondern als eigenständiges Modul „on top“ platziert wurde. Als GPS-Modul wurde schließlich der Typ GT-7U 1728 von Open-Smart (**Bild 4**) gewählt, eines jener erstaunlich preisgünstigen chinesischen Produkte, das man genialerweise im Elektor-Store [3] erwerben kann. Durch die abgesetzte Montage ist das GPS-Modul einfacher zu handhaben und durch die Möglichkeit, eine externe Antenne anzuschließen, ist das

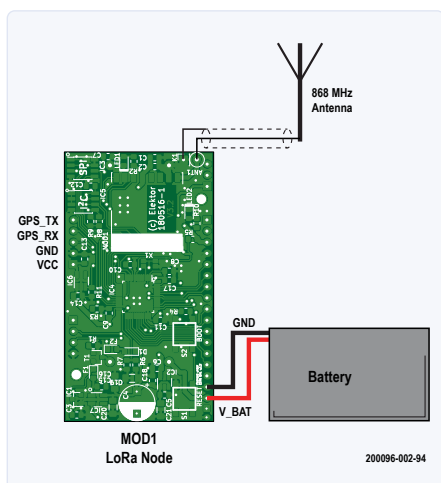


Bild 4. Das GPS-Modul am LoRa-Knoten.

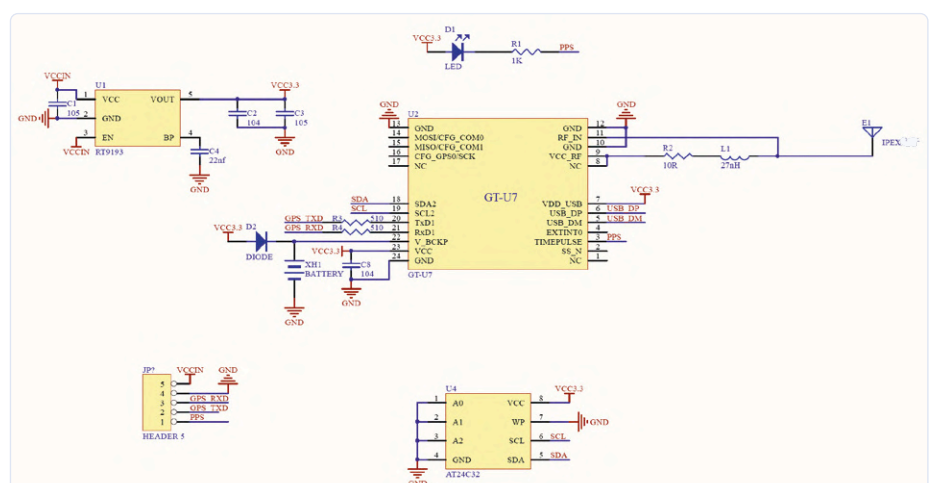


Bild 5. Innenschaltung des GPS-Moduls.

Modul erfreulicherweise sehr „empfangsbereit“. Zudem kann sich die Basisplatine so auf die für alle geplanten Anwendungen notwendigen Funktionsgruppen beschränken.

Im Elektor-Store finden Sie neben einigen grundlegenden Daten zu dem GPS-Modul etwas versteckt unter „For more information click here“ einen Link zu einem Archiv mit Treibern, Handbuch, Arduino- und MCU-Codes, Testwerkzeugen und einem Schaltplan (**Bild 5**). Dabei fällt auf, dass beim Betrieb des GPS-Moduls über den USB-Port auch die Versorgung über den USB-Port erfolgt. Dabei ist die USB-Versorgungslleitung direkt mit dem VCC-Pin (VCC für das Modul ist spezifiziert mit 5 V max.) des Moduls verbunden, so dass der Tracker Schaden nehmen würde, schlösse man eine 5-V-USB-Verbindung an. Ohne weitere Maßnahmen wären dann die 5 V des USB mit der 3,3-V-Versorgung der Platine direkt verbunden.

Doch auch wenn der USB den erlaubten Spannungsbereich einhalten würde (3,0...3,6 V), käme es zu einem Konflikt, wenn das GPS-Modul vom LoRa-Knoten versorgt werden würde. Um dies zu verhindern, kommt mit IC6 wieder eine „ideale Diode“ zum Einsatz. Ist die Spannung auf der Ausgangsseite des MAX40200 höher als auf der Eingangsseite, so sperrt die „ideale Diode“. Wird das Modul dagegen aus den 3,3 V der Basisplatine versorgt, verursacht die „ideale Diode“ einen Spannungsabfall von nur 85 mV, so dass das GPS-Modul innerhalb der Betriebsparameter arbeitet. Zudem hat der Controller die Möglichkeit, über den Enable-Pin die Spannung zum Modul komplett abzuschalten. Am besten wäre es aber, man ließe in unserer Applikation die Finger ganz von dem USB-Anschluss des GPS-Moduls! Um das GPS-Modul in das Gehäuse zu bekommen, wird es huckepack auf die Basisplatine gesteckt (**Bild 6**) und schwebt dann einige wenige Millimeter über den Bauteilen des LoRa-Knotens. So nutzen Drohnenakku, Basisplatine und GPS-Modul den zur Verfügung stehenden Raum im Gehäuse optimal aus. Neben der GPS-Antenne, die im Gehäuse liegt, muss auch die LoRa-Antenne berücksichtigt werden. Eine miniaturisierte Helix-Antenne wäre so klein, dass sie im Gehäuse verbaut werden könnte, jedoch würde dann die Sendeleistung um 30 dBm gedämpft, so dass es fast unmöglich wäre, ein Signal zu einem LoRaWAN-Gateway zu senden. Deshalb ist eine externe Antenne vorteilhaft, sei es als simpler Draht oder formschöne, wasserdichte Konstruktion wie bei [3] gezeigt.

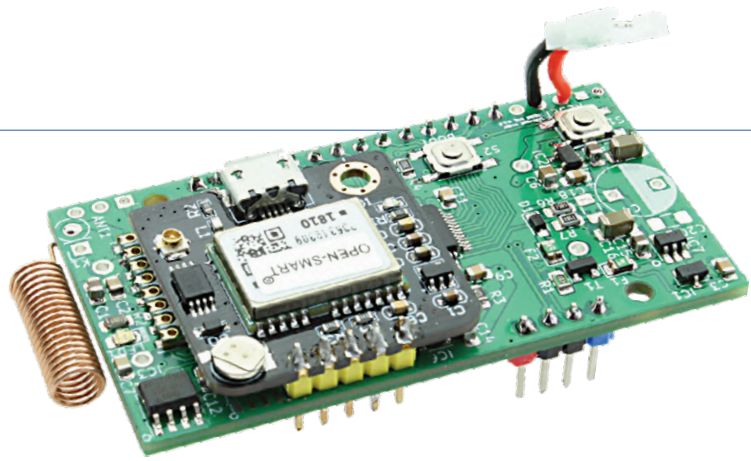


Bild 6. Das GPS-Modul schwebt einige Millimeter über der Basisplatine.

Die Firmware

Die Firmware besteht, wie in **Bild 7** dargestellt, aus vier wesentlichen Teilen, einem Dekoder für GPS-Daten, der Arduino-Bibliothek LMIC als LoRaWAN-Stack, einer Low-Power-Bibliothek und einer seriellen Konsole zur Parametrierung. Das Dekodieren der GPS-Nachrichten übernimmt die Bibliothek TinyGPS++, die Elektor-Lesern vielleicht schon vom ESP32-NTP-Server [5] bekannt ist. Damit die Bibliothek ihre Arbeit verrichten kann, müssen die ankommenden Daten von der seriellen Schnittstelle gelesen und der Bibliothek zum Dekodieren übergeben werden. Um zu vermeiden, dass das GPS-Modul mit der seriellen Schnittstelle kollidiert, die für das Hochladen der Firmware verwendet wird, kommt der zweite UART zum Einsatz, der im Sketch einfach durch `HardwareSerial SerialX(USART2);` initialisiert wird. In der **Boardbeschreibung** sind die Pins für die Ansteuerung des GPS-Moduls schon definiert. Bei der verwendeten Bibliothek muss der Controller die ankommenden Daten aus dem Puffer des UART entnehmen und der Bibliothek übergeben. Um festzustellen, ob das GPS-Modul eine neue Position ermittelt hat, pollt diese Bibliothek die aktuelle Position und wertet die Antwort aus. Für die LoRaWAN-Kommunikation wird die gleiche LMIC-Bibliothek genutzt, die auch schon im LoRa-Switch und einer Vielzahl anderer LoRaWAN-Arduino-Projekte den LoRaWAN-Stack bereitstellt. Beim Einsatz der Bibliothek muss beachtet werden, dass die Funktion `os_runloop_once` so häufig wie möglich aufgerufen werden muss und dass dabei möglichst keine langen Unterbrechungen auftreten sollten. Dies ist übrigens auch einer der Gründe, warum es nicht so einfach ist, die LMIC-Bibliothek auf einem ESP32 zum Laufen zu bringen und die geforderten zeitlichen Abfolgen einzuhalten. Für den Datentransport wird eine Nutzlast an die Bibliothek übergeben und die Rückmel-

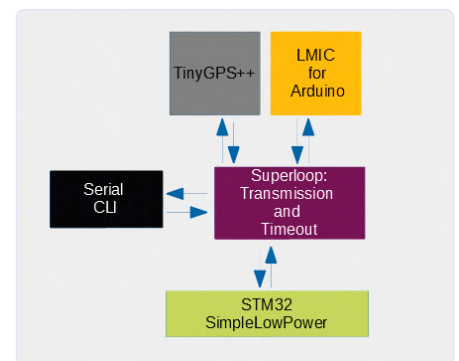
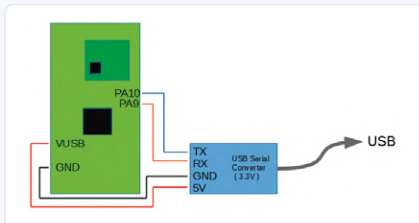


Bild 7. Aufbau der Firmware.

dung abgewartet, dass die Daten gesendet wurden. In den meisten Sketches wird die Konfiguration der LoRaWAN-Bibliothek mit allen Zugangsdaten beim Kompilieren festgelegt, hier hingegen kann diese zur Laufzeit geändert werden. Diese Konfiguration erfolgt über eine serielle Kommandozeile und kann zum Beispiel mit dem Seriellen Monitor der Arduino-IDE (Bild 10) vorgenommen werden. Die Grundlagen dieser Kommandozeile stammen aus der Temperaturgesteuerten DIY-Lötstation [6], wo sie für das Setzen einiger Parameter eingesetzt wurde. Für den LoRa-GPS-Tracker wurden ein paar Funktionen hinzugefügt, etwa das Parsen von HEX-Strings und ein paar kleinere Modifikationen, um neue Befehle komfortabler einfügen zu können. Der Code selber lässt sich nicht nur für den STM32 verwenden, sondern läuft mit ein paar Modifikationen auch auf dem ESP32 und AVR-Mikrocontrollern. Das Interface selber arbeitet auf dem ersten UART und kann über die Pins PA9 (RX MCU) und PA10 (TX MCU) angesprochen werden. Die Schnittstelle arbeitet mit 115200 Baud, 8 Datenbits, keiner Parität und 1 Stopbit (115200Baud 8N1). Im Serial Monitor der Arduino-IDE muss lediglich die Geschwin-

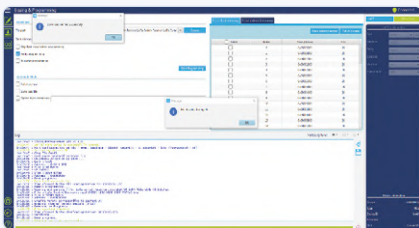
FIRMWARE ZUM CONTROLLER HOCHLADEN

All die Hardware macht noch keinen LoRa-Tracker. Dazu bedarf es „etwas“ Software, die Ihnen auf der Projektseite [9] als HEX-Datei zur Verfügung steht. Um die Software in die Hardware zu bekommen, reichen ein einfacher USB-Seriell-Wandler [siehe Produkte-Kasten] und vier Jumper-Drähte (female-female). Die HEX-Datei kann mit STM32Cube-Programmer programmiert



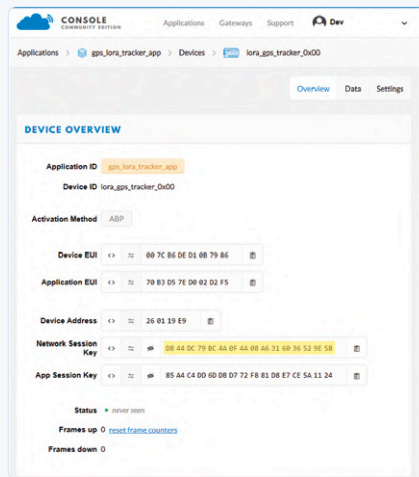
werden. Dazu muss der Chip durch gleichzeitiges (und kurz anhaltendes) Drücken von *Boot* und *Reset* in den Bootloader-Modus geschaltet werden. Dann kann man rechts die passende serielle Schnittstelle wählen und durch Drücken auf *Connect* verbinden. Wenn keine Fehlermeldung erscheint, kann es mit dem Programmieren der Firmware losgehen.

Nach erfolgreicher Programmierung kann der Chip per *Reset* aus dem



Bootloader-Modus geholt werden. Er beginnt dann sofort mit der Ausführung der Firmware.

Wer nicht nur die Firmware in den Chip programmieren möchte, kann in der Arduino-IDE die Firmware an eigene Wünsche anpassen und sie dann neu kompilieren (und hochladen). Die Arduino-IDE unterstützt das Elektor-LoRa Board mit der Version 1.8.0 des STM32duino-Projekts, ohne dass Modifikationen von Dateien notwendig wären. Wer allerdings die aktuelle Version 1.8.0 über den Arduino-Boardmanager installiert, sei gewarnt, da es ein paar Bugs gibt, die zum Absturz der Firmware führen können. Mit der aktuellen Version aus dem Github-Repositorium des STM32duino-Projekts sind diese Fehler aber behoben.



digkeit auf 115200 eingestellt werden. Der vierte Teil ist eine minimalistische Low-Power-Bibliothek, die dafür sorgt, dass der STM32 für eine definierte Zeit in den Sleep-Mode versetzt wird und dann wieder aufwacht. Dabei wird der Inhalt des RAMs beibehalten und auch die interne Echtzeituhr arbeitet weiter. Das Aufwachen wird durch einen eingestellten Alarm in der Echtzeituhr initiiert. Es gibt zwar eine deutlich funktionsstärkere Low-Power-Bibliothek, die allerdings auch mehr als 12 kB Flashspeicher benötigt. Bei der Beschreibung der einzelnen Komponenten darf nicht vergessen werden, dass für das Senden einer Nachricht mit der LMIC-Bibliothek maximal 51 Byte zur Verfügung stehen. Je kürzer die Nachricht, desto mehr

Laufzeit ist verfügbar und das LoRaWAN sowie das 868-MHz-Frequenzband werden weniger belastet. In der aktuellen Firmware-version werden 12 Byte für eine Position gesendet (**Bild 8**), aber dies ist nicht optimal und könnte auf 7 Byte verkürzt werden.

Im ersten IEEE754-Float ist der Längengrad zu finden, im zweiten der Breitengrad. Die letzten 4 Byte sind für diverse Statusinformationen (**Bild 9**) vorgesehen (sind die Positionsdaten gültig, nach welcher Zeit wurde eine GPS-Position ermittelt, Batteriespannung). Diese Daten können als Zusatzinformationen genutzt werden. Doch da der Code offen liegt, lassen sich auch einfach eigene Werte einfügen.

IEEE 754 float (4Byte)	IEEE 754 float (4Byte)	4 Byte Status
---------------------------	---------------------------	---------------

Bild 8. Datenpaket für eine GPS-Position.

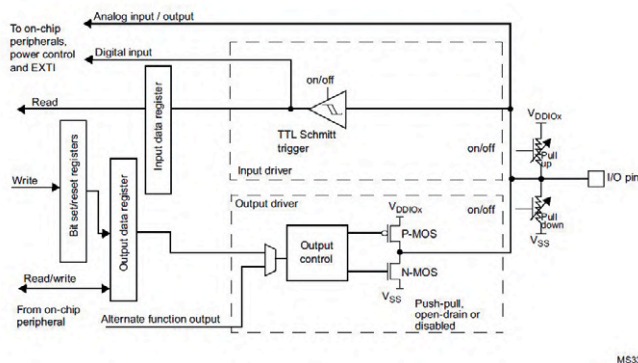
Reserved Bit 31 - 16	VBat/2 Bit 15 - 8	Time until GPS-Fix Bit 7 - 1	GPS Invalid Bit 0
-------------------------	----------------------	---------------------------------	----------------------

Bild 9. Bedeutung des Statusbytes.

Energieeffizienz auf die Spitze getrieben

Da die Energie, die aus den Akkus entnommen werden kann, endlich ist, muss auf Energieeffizienz großen Wert gelegt werden. Das bedeutet, dass alle nicht benötigten Verbraucher abgeschaltet oder in den Sleep-Modus versetzt werden. Das Abschalten des GPS-Moduls ist einfach: Ein I/O-Pin des Controller schaltet über den Enable-Pin der idealen Diode IC6 die Versorgungsspannung des GPS-Moduls ein oder aus. Das LoRa-Modul fällt nach dem Senden ohnehin automatisch in einen Ruhemodus. Das Datenblatt des STM32 lässt nun vermuten, dass der Strom zum Controller in einem Bereich von 68 μ A liegen sollte. Wenn man aber den Ruhestrom der Schaltung ohne weitere Maßnahmen misst, so zeigt das Messgerät immer noch 0,5...0,7 mA an, wovon der STM32 einen Großteil für sich beansprucht. Wohin fließt der Strom? Die Antwort liegt in den nicht verwendeten I/O-Pins. Jeder Pin, der nicht durch den Code entsprechend definiert ist, arbeitet automatisch als Eingang, an dem keine definierte Spannung anliegt (siehe Kasten „Stromaufnahme nicht genutzter I/O-Pins“). Als Gegenmaßnahme kann und sollte man die internen Pull-Up-Widerstände aktivieren, um einen definierten Pegel zu erhalten. Zusätzlich kann noch etwas Energie gespart werden, wenn man die Pins für den UART zum GPS-Modul als Eingangspins definiert, wenn das Modul abgeschaltet ist, sonst könnte der TX-Pin vom STM32 dennoch 3,3 V an den I/O-Pin des GPS-Moduls legen. Neben dem STM32 tragen auch noch andere Komponenten zum Energieverbrauch bei, zum Beispiel der LDO-Regler, die idealen Dioden, das LoRa-Modul und die Kondensatoren. Da wir nicht päpstlicher als der Papst sein wollen, belassen wir es dabei. So kommt der LoRa-GPS-Tracker realistisch gesehen auf einen Ruhestrom von 150 μ A bei 25°C Umgebungstemperatur.

STROMAUFNABME NICHT GENUTZTER I/O-PINS



Im *Reference Manual RM0091* wird der Aufbau für die Logik eines I/O-Pins beschrieben. Der verbaute Schmitt-Trigger sorgt dafür, dass der Pin, wenn er als digitaler Eingang arbeitet, entweder ein logisches High oder Low liefert, in dem die anliegende Spannung mit fest eingestellten Schwellwerten verglichen wird. Wenn der Pin nach außen nicht verbunden ist, fungiert dieser als Antenne, so dass undefinierte, sich ändernde Spannungen zum Schmitt-Trigger gelangen. Dadurch wechselt der Schmitt-Trigger oft zwischen High und Low, wodurch jedes Mal ein kleiner Strom aufgenommen wird. Bei den ungenutzten Pins des Boards summiert sich dies zu einer Stromaufnahme von gemessenen 0,5...0,7 mAh.

Wenn der LoRa-GPS-Tracker erwacht, steigt der Energieverbrauch erheblich. Der Löwenanteil entfällt auf das GPS-Modul, das um die 60 mA benötigt und damit maßgeblich für die Standzeit des GPS-Trackers verantwortlich ist. Jetzt wird auch klar, warum das GPS-Modul so schnell wie möglich die Position erfassen sollte! Das LoRa-Modul benötigt für die Zeit der Datenversendung zwar 100 mA in der Spitze, das Sendeintervall liegt jedoch unter einer Sekunde. Die MCU benötigt etwa 2,5 mA bei einem CPU-Takt von 8 MHz und hat so ausreichend Rechenzeit für das Verarbeiten der Positionsdaten und das Senden im LoRaWAN zur Verfügung. Wenn die maximale Rechenleistung nicht benötigt wird, sollte versucht werden, die Taktrate der CPU so weit wie möglich herunterzusetzen. Damit ist die maximale Standzeit des Akkus im wesentlichen abhängig vom Intervall, in dem die Position gesendet wird, und der Zeitspanne (Timeout), nach der der Versuch, die Position mit dem GPS-Modul zu ermitteln, bei schlechtem Empfang abgebrochen werden soll.

Parameter einstellen

Damit nicht bei jeder Parameteränderung eine neue Firmware auf den Chip gespielt werden muss, gestattet die serielle Konsole, die sich zum Beispiel mit dem Serial Monitor der Arduino-DIE bedienen lässt, einige grundlegende Einstellmöglichkeiten:

- Intervall für das Senden der GPS-Position
- Timeout für dem Empfang einer GPS-Position
- LoRa-Datenrate

Diese drei Parameter haben Einfluss auf den Energiebedarf des LoRa-GPS-Trackers. Das Intervall zum Senden der Position wird mit `set INTERVALL <Minuten>` eingestellt und kann mit `get INTERVALL` abgefragt werden. Gleiches gilt für den Timeout für die GPS-Position, die per `set GPSTIMEOUT <Minuten>` und `get GPSTIMEOUT` eingestellt respektive abgefragt wird. Voreingestellt sind 15 Minuten für das Sendeintervall der Positionsdaten und 10 Minuten für das Timeout bei schlechtem/keinem GPS-Empfang.

Die LoRa-Datenrate kann im Bereich SF7...SF10 eingestellt werden, wobei SF7 die höchste Datenrate und damit die geringste Sendezeit bedeutet und SF10 langsam ist, so dass die Modulation mehr Energie pro Zeiteinheit benötigt. SF7 ist voreingestellt und sorgt für eine zügige Datenübertragung, aber auch eine geringe Reichweite. Wenn der

Wert von SF7 angehoben wird, wird zwar die Reichweite erhöht, sollte aber der Tracker in Bewegung sein, bedeutet es auch, dass es zu Störungen bei der Datenübertragung kommen kann: ein zweischneidiges Schwert! Auch reduziert sich dabei die Anzahl an Positionen, die pro Tag gesendet werden. Das *The Things Network* sieht 30 Sekunden Sendezeit pro Tag vor; bei SF7 braucht die Übermittlung einer Positionsnachricht 62 ms Sendezeit, bei SF10 sind es 412 ms. Damit könnten bei SF7 etwa 450 Nachrichten pro Tag gesendet werden, bei SF10 nur etwa 70 Nachrichten. Damit Daten per LoRaWAN gesendet werden können, müssen die Schlüssel für das Endgerät eingetragen werden. Für den LoRa-GPS-Tracker wird der ABP-Modus verwendet. Damit sind wir schon bei der Konfiguration im *The Things Network* angelangt.

Konfiguration im The Things Network

Der Einstieg in *The Things Network* wurde schon in Elektor März/April [4] ausführlich behandelt, so dass wir dieses Thema hier nur streifen müssen. Nach dem Anlegen eines Benutzerzugangs und der Anmeldung muss zunächst durch Klicken auf *Applications* und auf der folgenden Seite auf *add application* eine Applikation erstellt werden, der der LoRa-GPS-Tracker zugeordnet werden kann. Im folgenden Dialog müssen ein eindeutiger Name (`gps_lora_tracker_app`) und eine Beschreibung (*Test Sensor Nodes*

for Development) der Applikation angegeben werden.

LoRa-Geräte werden immer einer Applikation zugeordnet, so dass die im *The Things Network* eintreffenden Daten direkt passend dekodiert und weiter verarbeitet werden können. Für den LoRa-GPS-Tracker spielt dies erst einmal keine Rolle, da das Aufbereiten der Daten durch einen Server (Raspberry Pi) mit Node-RED vorgenommen wird.

Für die nun erstellte Applikation muss nun noch ein Device, unser LoRa-GPS-Tracker angelegt werden. Mit *register device* öffnet sich ein Dialog, in dem die Daten für den LoRa-GPS-Tracker eingetragen werden können. Für jeden LoRa-GPS-Tracker muss ein eigener, eindeutiger Name vergeben und der Vorgang mit *Register* abgeschlossen werden. In dem nun erstellten Device muss unter *Settings* die Aktivierungsmethode von OTAA auf ABP umgestellt werden, um an die Daten zu kommen, die in die Firmware des LoRa-GPS-Trackers eingetragen werden müssen.

Nach dem *Save* können dann der Übersicht die nötigen Informationen (*Network Session Key*, *Application Session Key* und *Device Address*) entnommen werden, um den LoRa-GPS-Tracker mit der seriellen Konsole passend einzustellen. Die drei Keys muss man nicht mühsam abtippen, sondern man kann sie durch das Klemmbrettsymbol aus dem *TheThingsNet-*

ALTERNATE PIN FUNCTIONS, EIN SPASS FÜR DIE GANZE ENTWICKLUNGSABTEILUNG

Alternate Pin Functions haben erst in jüngster Zeit Einzug in der Welt der AVR-Mikrocontroller gehalten, bei auf Cortex-M basierenden Controllern wie dem STM32 gehören sie aber schon länger zum Standard. Musste man sich beim ATmega328 noch entscheiden, ob ein Pin ein GPIO war oder eine spezielle Funktion hatte wie SPI, kann beim STM32 ein Pin wechselnde Funktionen besitzen. So lässt sich der UART0 an die Pins PA8 und PA9 oder PB6 und PB7 legen. Bei allen Pins mit alternativen Funktionen sollte man frühzeitig entscheiden, ob man sie – auch wenn man sie erst einmal nicht nutzt – nicht doch nach außen auf einen Anschluss routet. Es wäre nämlich ärgerlich, wenn man erst nach Fertigstellung des Projekts bei einem Blick auf das Datenblatt feststellt, dass ein nicht nach außen geführter Pin der Serielle Takt der zweiten, vielleicht doch nutzenswerten SPI-Schnittstelle ist. Deshalb sollte durch Abstimmung zwischen Software- und Hardwareentwicklern das Layout der Platine so gestaltet werden, dass manche *Alternate Pin Functions* auch für spätere Einsatzzwecke zur Verfügung stehen.

work (*Device Overview*) in den Seriellen Monitor kopieren (**Bild 10**). Beim Übertragen werden die Keys intern umgewandelt und im EEPROM gespeichert. Sind alle Parameter eingegeben, ist der LoRa-GPS-Tracker bereit für den ersten Einsatz.

Vom LoRa-GPS-Tracker zur Landkarte

Nachdem der LoRa-GPS-Tracker eingerichtet ist, sendet er Daten in das LoRaWAN, die jedoch weiter verarbeitet werden müssen, bevor eine Position zum Beispiel auf einer

Landkarte angezeigt wird. Dies kann mit Hilfe von Node-RED geschehen, wobei als Rechner ein Raspberry Pi verwendet werden kann (nicht muss, es kann auch eine virtuelle Maschine oder ein Rechner sein, auf dem sich Node-RED befindet). Im Raspberry Pi OS (einst Raspbian) ist kein Node-RED vorhanden. Auf der Webseite [7] wird die Installation (für Raspbian >= Jessie) und die Einrichtung von Node-RED ausführlich beschrieben. Eine Einführung in Node-RED wurde im einem Elektor-Artikel [8] vermittelt.

Momentan werden die Daten von LoRa-GPS-Tracker an das TheThingsNetwork transportiert, dort aber nicht gespeichert, sondern an ein bereit stehendes System weitergeleitet. Als Schnittstelle zum Abholen der Daten kommt das MQTT-Protokoll zum Einsatz, mit dem schon in anderen Elektor-Projekten gearbeitet wurde. Von Vorteil ist hierbei, dass für TheThingsNetwork in Node-RED

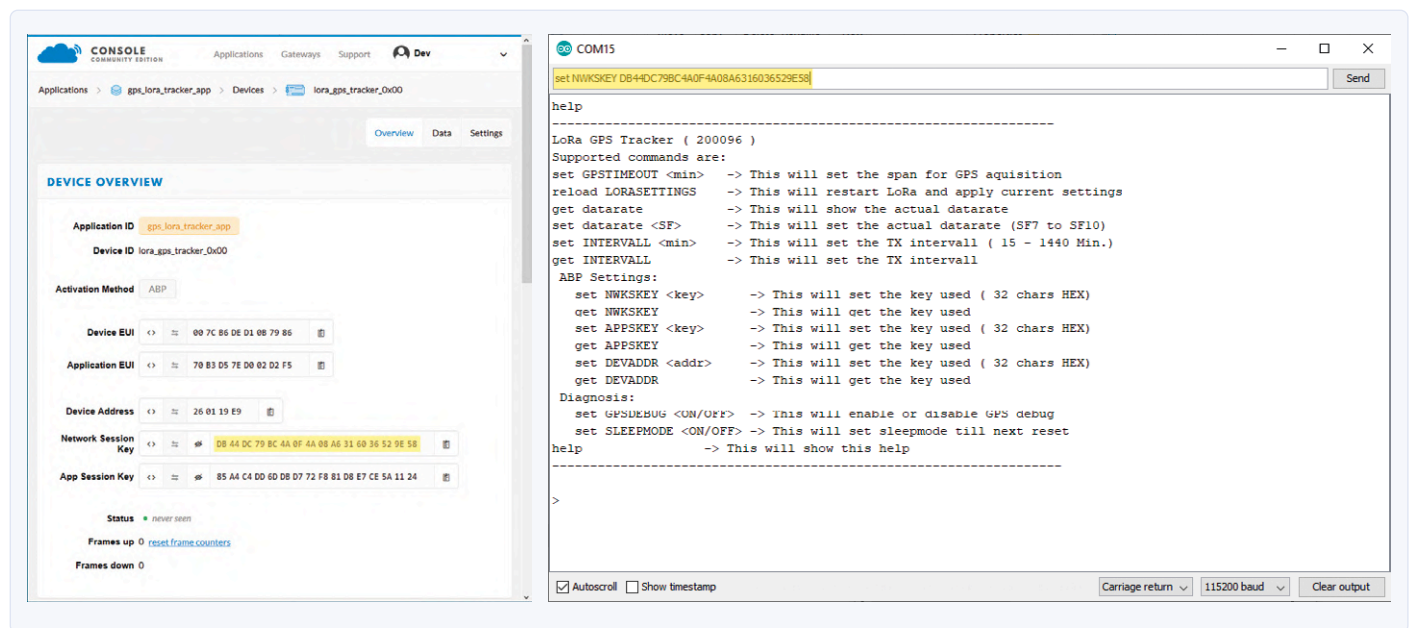


Bild 10. Kopieren der Keys aus der Device Overview in den Seriellen Monitor.

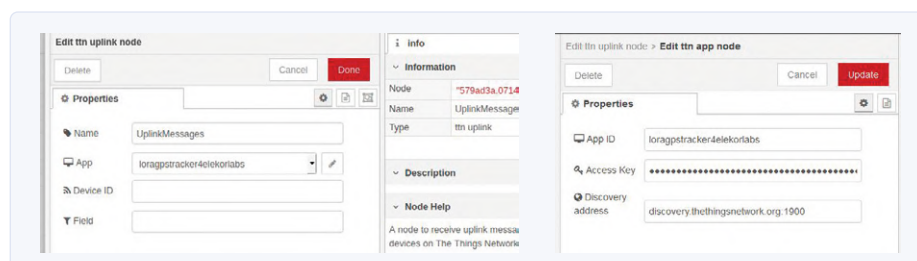


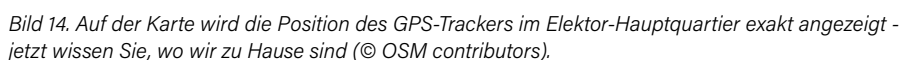
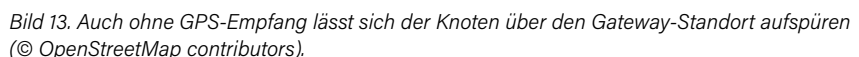
Bild 11. Übertragung der TTN-Daten in Node-RED (Node_Red_TTN_node_Info und TTN_Access).

Sie haben Fragen oder Kommentare?

Gerne können Sie sich an den Autor des Artikels wenden unter der E-Mail-Adresse

mathias.claussen@elektor.com.

Quelle der Kartendaten ist das Material des OpenStreetMap-Projekts. In diese Karte können für Objekte Positionen gesetzt werden, bei denen ein Icon, ein Radius und ein paar weitere Parameter für die Position





STÜCKLISTE

Elektor LoRa Node (Platine bei Elektor erhältlich*, Stückliste und KiCAD-Files unter [10], Firmware downloadbar unter [10])	868-MHz-Antenne SMA
RFM95 Ultra LoRa Transceiver Module (868/915 MHz) *	Gehäuse Hammond 1551K
OPEN-SMART GPS – Serial GPS Module *	Akku-Kabel mit Stecker 2 mm RM (Molex 51005)
SMA-Kabel mit Pigtail und Lötflanke	Akku (siehe Text)
	* siehe Kasten „Passende Produkte“

angezeigt werden. Ein Nachteil ist, dass die Positionen nach dem Schließen des Browsers nicht erhalten bleiben. Daher werden die letzten 2048 Datenpunkte, die in die Karte eingetragen wurden, im RAM

des Node-RED-Servers vorgehalten. Wenn die Karte wieder aufgerufen wird, so werden diese auch wieder angezeigt. Damit dennoch Strecken und Positionsveränderungen über eine längere Zeit nachvollziehbar sind, werden

diese (für jeden einzelnen Tracker) in einer CSV-Datei Excel-kompatibel abgelegt. Die Anzeige auf der World-Map ist daher keine komplett fertige Lösung für die Darstellung der Positionsdaten, sondern soll Ihnen den Datenfluss vom LoRa-GPS-Tracker bis zum Anwender verdeutlichen und gleichermaßen als Vorlage für eigene Ideen dienen.

Zusammenfassung

LoRaWAN bietet eine interessante Plattform, um den Standort von Gegenständen zu verfolgen. Mit der hier vorgestellten Lösung können die Positionen von Gegenständen ermittelt, per LoRaWAN übertragen und auf einem eigenen Rechner dargestellt werden. Durch den quelloffenen Charakter von Hard- und Software kann jeder, der es möchte, einen Blick hinter die Kulissen wagen und diese an eigene Bedürfnisse anpassen.

Die kompletten Daten für den LoRa-GPS-Tracker stehen als KiCad-Projekt auf GitHub [10] bereit und können unter der OSHL V1.2 verwendet werden. Der Code der Software sowie das Node-Red Projekt zur Anzeige der Position sind ebenfalls auf GitHub zu finden. Neben der Verwendung als GPS-Tracker können und werden in – wegen der COVID19-Lage leider unbestimmter – Zukunft noch weitere Anwendungen mit dem Elektor-LoRa-Knoten realisiert werden. 

200096-01



PASSENDE PRODUKTE

- > **Elektor LoRa Node Leerplatine 180516-1**
www.elektor.de/elektor-lora-node-bare-pcb-180516-1
- > **RFM95 Ultra LoRa Transceiver Module (868/915 MHz)**
www.elektor.de/rfm95-ultra-lora-transceiver-module-868-915-mhz
- > **OPEN-SMART GPS – Serial GPS Module**
www.elektor.de/open-smart-gps-serial-gps-module-for-arduino-arduino-2-5-flight-control
- > **USB to TTL Converter UART Module CH340G (3.3 V/5.5 V)**
www.elektor.de/ch340-usb-to-ttl-converter-uart-module-ch340g-3-3-v-5-5-v
- > **Buch „Programming with Node-RED“**
www.elektor.de/programming-with-node-red
www.elektor.de/programming-with-node-red-e-book

Ein Beitrag von

Idee, Entwicklung, Text und Bilder:
Mathias Claußen

Text und Redaktion: **Rolf Gerstendorf**

Layout: **Giel Dols**

WEBLINKS

- [1] **ElekTrack:** www.elektormagazine.de/magazine/elektor-200710/2826
- [2] **Teures Roaming:**
www.welt.de/vermischtes/article202525470/Roaming-Gebuehren-Als-die-Adler-in-den-Iran-flogen-wurde-es-teuer-fuer-die-Forscher.html
- [3] **Der Elektor-LoRa-Knoten (Elektor 3-4/2020):** www.elektormagazine.de/magazine/elektor-140/57120
- [4] **LoRaWAN – ein einfacher Einstieg (Elektor 3-4/2020):** www.elektormagazine.de/magazine/elektor-140/57101
- [5] **ESP32 als Zeit-Server:** www.elektormagazine.de/magazine/elektor-99/50869
- [6] **Temperaturgesteuerte DIY-Lötstation:** www.elektormagazine.de/magazine/elektor-69/42252
- [7] **Node-RED Getting Started:** <https://nodered.org/docs/getting-started/raspberrypi>
- [8] **Einstieg in Node-RED:** www.elektormagazine.de/articles/einstieg-in-nodered
- [9] **Projektseite:** www.elektormagazine.de/200096-01
- [10] **KiCad Files auf GitHub:** https://github.com/ElektorLabs/180516-Elektor_LoRa_Node

Programmierung eines Endlichen Zustandsautomaten

Mit 8-Bit-PICs in Assembler und C

Von **Andrew Pratt** (Großbritannien)

In dieser Ausgabe präsentieren wir drei „Learning-by-doing“- und zwei „Nice-to-know“-Kapitel aus dem Buch „Programming the Finite State Machine - with 8-Bit PICs in Assembly and C“. Das Buch richtet sich an Nutzer von Microsoft Windows und Linux und basiert auf einer einfachen Toolchain mit einem FTDI-USB-TTL-Serial-Kabel als Programmieradapter und dem Open-Source-Assembler „gpasm“. Wir zeigen einen LED-Blinker in PIC-Assemblersprache, Multiple-FSMs in einem Programm und einen 7-Segment-LED-Treiber.

Ein komplizierterer LED-Blinker

Einer der Kritikpunkte an Programmen in der Assembler-Sprache ist, dass sie schwer zu lesen sind. Dies kann stimmen, selbst wenn Sie das Programm gerade erst selbst erstellt haben! Mit einem Standard-Layout für Ihr Programm ist das Ganze sehr viel übersichtlicher. Außerdem gibt das Endliche Zustandsdiagramm einen Überblick über die gesamte Vorgehensweise des Programms. Der Assemblercode befindet sich in separaten Blöcken, die die states darstellen. Sie werden feststellen, dass es durch das Zustandsdiagramm viel einfacher ist, dem Assemblercode zu folgen.

Als Beispiel für ein komplizierteres Programm ist in **Bild 2-5** das Zustandsdiagramm für die gleiche LED zu sehen, sie blinkt jetzt mit den gleichen Intervallen wie vorher, aber nur für drei Blinkzyklen [Bezugnahme auf ein Programm, das in einem früheren Kapitel des Buchs beschrieben wurde – die Redaktion]. Danach blinkt sie für zehn Zyklen zehn Mal so schnell, bevor sie wieder auf das längere Intervall zurückfällt. Es gibt dabei vier Zustände mit sechs Übergängen. Es gibt Ein- und Aus-Zustände für schnelles Blinken und Ein- und Aus-Zustände für langsames Blinken. Das gesamte Diagramm kann in vier Quadranten mit zwei Trennlinien unterteilt werden: eine zwischen schnell und langsam und eine

zwischen ein und aus.

Bild 2-6 zeigt das Oszillogramm für die Spannung über der LED. Das Programm ist in **Listing 1** dargestellt. Wenn Sie das Zustandsdiagramm zu Rate ziehen, sollten Sie das Listing leicht verstehen. Es gibt eine Neuigkeit zu vermelden, nämlich die Verwendung des \$-Zeichens. Damit ist die Adresse der aktuellen Zeile gemeint, so dass wir, anstatt zu einem bestimmten Label zu springen, eine Anzahl von Zeilen überspringen können. Wenn Sie zum Beispiel **GOTO \$+3** schreiben, springen Sie drei Zeilen vorwärts.

Mehrere Maschinen im Programm

Die beiden Finite-State-Machine- oder FSM-Programme besaßen bisher einen kleinen Codeblock für jeden Zustand und haben diesen Code immer wieder ausgeführt, bis ein Eingabeereignis einen Sprung in einen anderen Zustand bewirkte. Es ist aber auch möglich, mehr als eine Maschine gleichzeitig in einem Programm laufen zu lassen. Das ist so, als würde man zwei Dinge quasi gleichzeitig tun, indem man einen Teil einer Aufgabe ausführt, dann einen Teil einer anderen Aufgabe und dann zurück zur ersten und so weiter. Dieses Zeitmultiplexing ist nötig, da die CPU im Mikrocontroller immer nur einen Befehl zur gleichen Zeit ausführen kann. Die

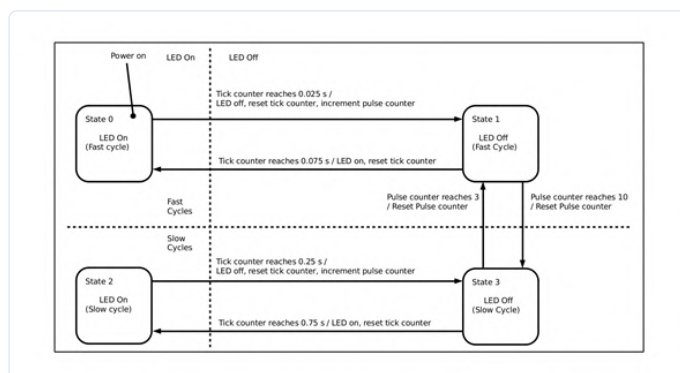


Bild 2-5. Zustandsdiagramm für das LED-Blink-Programm „Cycle of Cycles“.

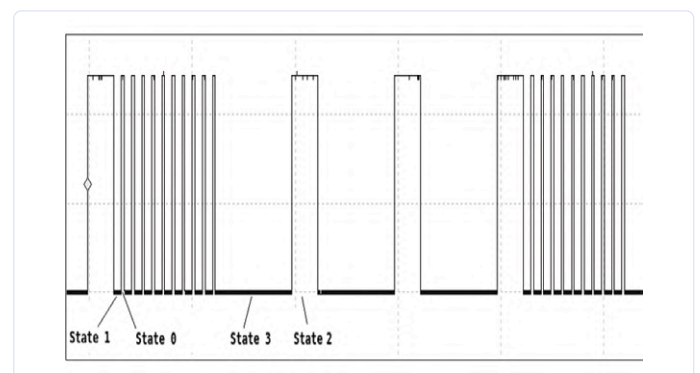


Bild 2-6. Oszillogramm für das Programm „Cycle of Cycles“.

Listing 1. Das Programm „Cycle of Cycles“ (prog_02_02.asm).

```
; prog_02_02.asm
; Page numbers refer to the data sheet DS40001413E.
; Tables refer to the book.

LIST      P=12f1822
#include <p12f1822.inc>

RADIX DEC                ; Default numbers are to base 10.

__CONFIG 0x8007, (_FOSC_INTOSC & _WDTE_OFF & _PWRTE_OFF & _CP_OFF & _BOREN_OFF & _IESO_OFF & _FCMEN_
OFF )
__CONFIG 0x8008, (_LVP_ON)

CBLOCK 0x70              ; In common RAM, accessible in all banks.
TICK_COUNTER
PULSE_COUNTER
ENDC

ORG 0x00
GOTO START

ORG 0x04                 ; Interrupt vector.
BCF INTCON, TMR0IF       ; Clear the tmr0 overflow interrupt flag. INTCON is in all banks.
INCF TICK_COUNTER, F     ; Increment by one counter_off.
RETFIE                  ; Return from interrupt

START
MOVLB 1                 ; To access TRISA, OSCCON, OPTION_REG.
BCF TRISA, 2            ; Configure PORTA bit 2 (chip pin 5) as an output
MOVLW 0x70
MOVWF OSCCON            ; Page 65 32MHz. See Table 2-1 in the book.
MOVLW 0xD7
MOVWF OPTION_REG        ; Page 164 in the data sheet and Table 2-2 in the book.
MOVLW 0xE0
MOVWF INTCON            ; Page 86 in the data sheet and Table 2-3 in the book.
MOVLB 0                 ; To access PORTA for the rest of the program.
BSF PORTA, 2            ; Inialise LED on;
;-----Machine States.
S0                       ; LED on. Fast cycle.
MOVLW 3
SUBWF TICK_COUNTER, W   ; C in STATUS is set when TICK_COUNTER >= to 3.
BTFSS STATUS, C         ; Skip the next instruction if C in STATUS is found set.
GOTO S0                 ; C in STATUS found not set. Stay in this state.
BCF PORTA, 2            ; Turn LED off
CLRF TICK_COUNTER       ; Reset the tick counter.
INCF PULSE_COUNTER, F   ; Add 1 to PULSE_COUNTER and put the result in PULSE_COUNTER.
                        ; Continue to S1

S1                       ; LED off. Fast cycle.
MOVLW 10
SUBWF PULSE_COUNTER, W  ; C in STATUS is set when PULSE_COUNTER >= to 10.
BTFSS STATUS, C         ; Skip the next instruction if C in STATUS is found set.
GOTO $+3                ; Have not reached 10 pulses jump to ticks check.
CLRF PULSE_COUNTER      ; 10 Pulses have been counted, reset PULSE_COUNTER.
GOTO S3                 ; Transition to S3.
MOVLW 9                 ; Load W with 9 for ticks check.
SUBWF TICK_COUNTER, W   ; C in STATUS is set when TICK_COUNTER >= to 9.
BTFSS STATUS, C         ; Skip the next instruction if C in STATUS is found set.
GOTO S1                 ; C in STATUS found set, 9 ticks counted.
CLRF TICK_COUNTER
BSF PORTA, 2            ; Turn LED on.
GOTO S0

S2
MOVLW 31
SUBWF TICK_COUNTER, W   ; C in STATUS is set when TICK_COUNTER >= to 3.
BTFSS STATUS, C         ; Skip the next instruction if C in STATUS is found set.
GOTO S2                 ; C in STATUS found not set.
BCF PORTA, 2            ; Turn LED off
```

```

CLRF TICK_COUNTER          ; Reset the tick counter.
INCF PULSE_COUNTER, F      ; Add 1 to PULSE_COUNTER and put the result in PULSE_COUNTER.
                           ; Continue to S3

S3                          ; LED on. Slow cycle.
MOVLW 3
SUBWF PULSE_COUNTER, W     ; C in STATUS is set when PULSE_COUNTER >= to 3.
BTFSS STATUS, C            ; Skip the next instruction if C in STATUS is found set.
GOTO $+3                   ; ave not reached 3 pulses jump to ticks check.
CLRF PULSE_COUNTER         ; Reset PULSE_COUNTER.
GOTO S1                     ; Transition to S1.
MOVLW 91
SUBWF TICK_COUNTER, W      ; C in STATUS is set when TICK_COUNTER >= to 9.
BTFSS STATUS, C            ; Skip the next instruction if C in STATUS is found set.
GOTO S3                     ; C in STATUS found set.
CLRF TICK_COUNTER
BSF PORTA, 2               ; Turn LED on.
GOTO S2

END

```

beiden Aufgaben müssen sich die Zeit des Prozessors teilen, wenn die Aufgaben gemeinsam ausgeführt werden sollen. Im Gegensatz zu Timern und anderer Peripherie des PIC, die separate Hardwareeinheiten sind und tatsächlich gleichzeitig laufen können, ist der gleichzeitige Ablauf mehrerer Aufgaben im Programm also nur scheinbar möglich. Sie werden als Threads ausgeführt, wobei sich jeder Thread die Zeit des Prozessors teilt. Um dieses Multi-Threading zu erreichen, führt die erste Maschine den Codeblock für ihren aktuellen Zustand aus, dann ist der Codeblock für den aktuellen Zustand der anderen Maschine an der Reihe und so weiter. Die verschiedenen Maschinen sind aber unabhängig voneinander, sie

verwenden nur zufällig dieselbe CPU im Time-Sharing. Sie können sich auch den Speicher teilen, um auf die Daten der jeweils anderen Maschine zuzugreifen.

In den bisherigen Beispielen wurde der Zustand dadurch definiert, dass das Programm sich in einem bestimmten Codeblock befand, jetzt wird der Zustand durch den in einem Register gespeicherten Wert definiert. Das Register, das den Zustand von Maschine 0 enthält, sei `STATE_M0`, für Maschine 1 sei es `STATE_M1`. **Bild 2-7** zeigt, wie ein solcher Programmablauf aussehen könnte.

Um dieses Umschalten zwischen den Zuständen zu steuern, können wir die Anweisung `BRW` verwenden. Die `BRW`-Anweisung ist ein

Listing 2. Denkbare Programm mit einer 2-Status-Maschine 0 und einer 3-Status-Maschine 1.

```

SWITCH_M0
    MOVFW STATE_M0          ; Moves the value of STATE_M0 to W.
    BRW                     ; The program will jump from here depending on W.
    GOTO M0_S0              ; The jump will be to this line if W = 0.
    GOTO M0_S1              ; The jump will be to this line if W = 1.

SWITCH_M1
    MOVFW STATE_M0          ; Moves the value of STATE_M0 to W.
    BRW                     ; The program will jump from here depending on W.
    GOTO M1_S0              ; The jump will be to this line if W = 0.
    GOTO M1_S1              ; The jump will be to this line if W = 1.
    GOTO M1_S2              ; The jump will be to this line if W = 1.

M0_S0
    -----                ; code to decide if a change of state is needed if so change STATE_M0. GOTO
    SWITCH_M1              ; Go to the other machine's switch.

M0_S1
    -----                ; code to decide if a change of state is needed if so change STATE_M0.
    GOTO SWITCH_M1         ; Go to the other machine's switch.

M1_S0
    -----                ; code to decide if a change of state is needed if so change STATE_M1.
    GOTO SWITCH_M0         ; Go to the other machine's switch.

M1_S1
    -----                ; code to decide if a change of state is needed if so change STATE_M1.
    GOTO SWITCH_M0         ; Go to the other machine's switch.

M1_S2
    -----                ; code to decide if a change of state is needed if so change STATE_M1.
    GOTO SWITCH_M0         ; Go to the other machine's switch.

```

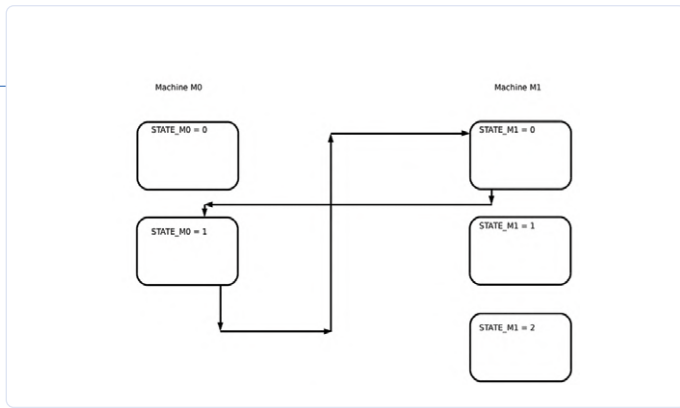


Bild 2-7. Beispiel für den Programmablauf mit zwei Maschinen im selben Programm.

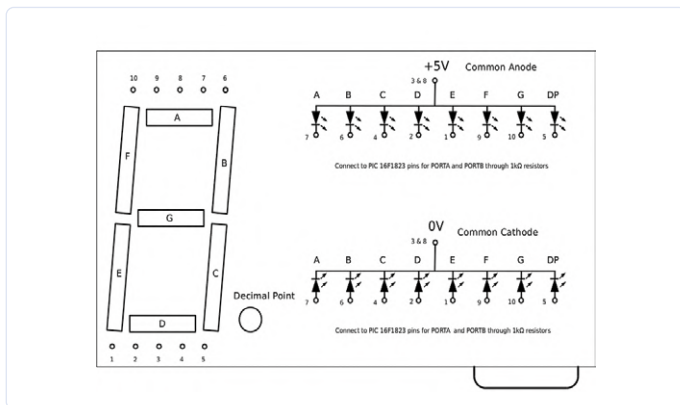


Bild 2-8. Anschlüsse für typische 7-Segment-LED-Anzeigen.

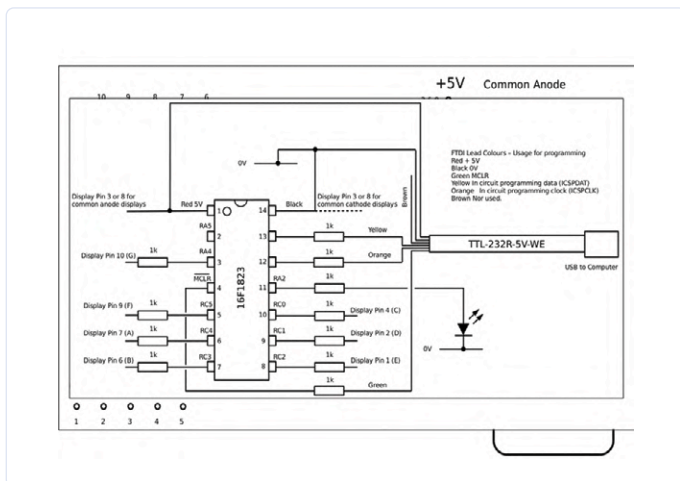


Bild 2-9. Anschlüsse für den PIC16F1823 und die 7-Segment-LED-Anzeige.

Schalter, ein **branch-with-w**, was bedeutet, dass der Programmzähler um den in **w** gespeicherte Wert steigt. Mit anderen Worten: Sie können einen definierten Sprung durchführen. Nachdem ein Zustands-Codeblock ausgeführt wurde, springt das Programm zurück zum Schalter für die nächste Maschine, wo die **BRW**-Anweisung zur nächsten Zeile springt, wenn der Zustandswert 0 ist, eine Zeile überspringt, wenn der Wert 1 ist und zwei Zeilen überspringt, wenn der Wert 2 ist. Der Programmfluss ist nicht in den Zustands-Codeblöcken gefangen, sondern wird über den Switch-Code umgeleitet. Um zwei Maschinen als Threads laufen zu lassen,

gibt es die beiden Zustandsvariablen **STATE_M0** und **STATE_M1** sowie zwei Switches. **Listing 2** zeigt einen vereinfachten Codeschnipsel dieses fiktiven Programms, wobei Maschine 0 zwei Zustände und Maschine 1 drei Zustände kennt.

Beachten Sie, wie der Programmfluss beim Verlassen eines Zustandscodeblocks zwischen den Maschinen springt.

Ansteuerung einer 7-Segment-LED-Anzeige

Um die praktische Verwendung der Switches zur Steuerung zweier Maschinen im selben Programm zu demonstrieren, lässt dieses Beispiel eine LED blinken und eine 7-Segment-Anzeige die Blink-Anzahl zählen. Man hat hier die Qual der Wahl zwischen einer Flüssigkristall- und einer LED-Anzeige. 7-Segment-LED-Anzeigen, die vielfältig und leicht erhältlich sind, können mit Gleichstrom betrieben werden, Flüssigkristallanzeigen benötigen eine Wechselspannung, LEDs „ziehen“ aber mehrere Milliampere pro Segment. Ich habe mich für eine einstellige LED-Anzeige entschieden, deren typische Anschlussbelegung in **Bild 2-8** zu sehen ist. Es gibt solche Anzeigen in zwei verschiedenen Ausführungen, mit einer gemeinsamen Kathode und einer gemeinsamen Anode. Im ersten Fall müssen die Kathoden an 0 V angeschlossen werden, bei gemeinsamer Anode müssen die Anoden mit +5 V verbunden werden. Die Segmentanschlüsse werden dann mit den Ausgängen des PIC verbunden. Die Logik des Ausgangscodes muss entsprechend der Polarität der Anzeige ausgelegt werden. Wenn Sie ein anderes Display als mit einem solchen Pinout verwenden, müssen Sie Ihre eigene Verdrahtung erarbeiten. Die Anzeige mit der gemeinsamen Anoden LA-601VB, die von RS Components geliefert werden kann, ist der von mir verwendeten ähnlich. Auch Farnell hat eine ähnliche Anzeige (Artikelnummer SA52-11SRWA).

Da insgesamt acht LEDs angesteuert werden müssen, wird der PIC 16F1823 mit dem zusätzlichen I/O-Port PORTC verwendet.

Bild 2-9 zeigt den Schaltplan des Versuchsaufbaus mit diesem PIC-Controller.

Der nächste Schritt in der Entwicklung besteht darin, einen Plan für den Zustandsautomaten zu zeichnen (**Bild 2-10**). Maschine M1 ist der Anzeigetreiber. Beim Einschalten wird die LED auf on initialisiert, dann startet die Maschine M0 im Zustand 0. Nach 0,25 s läuft der Timer ab und Maschine 0 geht in den Zustand 1 über. Der Timer wird zurückgesetzt, die LED erlischt, der Impulszähler wird um 1 inkrementiert und ein Signal namens UPDATE wird auf high gesetzt. Dieses UPDATE-Signal wird von Maschine 1 verwendet. Wenn der Timer im Zustand 1 bei 0,75 s abläuft, geht die Maschine in den Zustand 0 über; der Timer wird zurückgesetzt und die LED eingeschaltet. Dieser Ablauf wiederholt sich. Im Zustand 1, wenn der Impulszähler 10 erreicht hat, erfolgt ein Übergang in denselben Zustand. Dadurch wird der Impulszähler auf Null zurückgesetzt. In der Zwischenzeit läuft die Maschine M1 unabhängig, jedoch nur mit einem Teil der CPU-Zeit. Maschine M1 startet im Zustand 0 und die Anzeige ist leer. Wenn das UPDATE-Signal auf high geht, wechselt die Maschine in den Zustand 2 über und setzt das UPDATE-Signal auf low zurück. Je nach dem Wert der Variablen N geht die Maschine wieder in Zustand 1 über und schaltet die Anzeige an und aus. Die Maschine bleibt im Zustand 0 und wartet auf ein weiteres UPDATE-Signal. Die für die Maschine M1 angezeigte Ausgabe ist für eine Anzeige mit gemeinsamer Anode bestimmt. Die Kommentare im Code zeigen die Alternativen für eine Anzeige mit gemeinsamer Kathode.

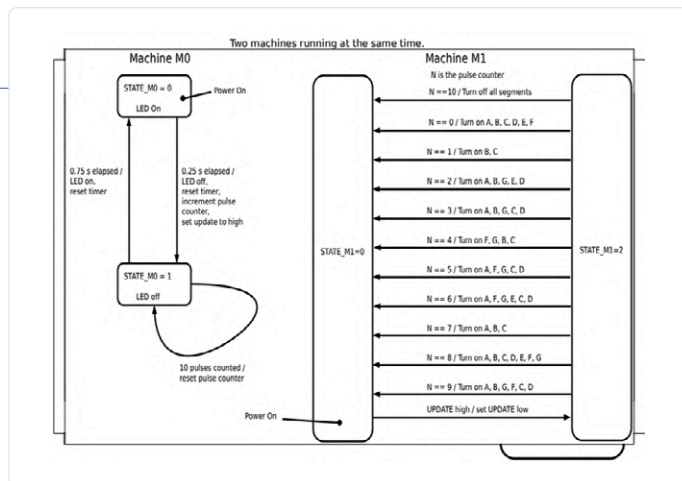


Bild 2-10. Zustandsdiagramm für das Programm 02-03.

Nahezu alles was im Programm „prog_02_03.asm“ in **Listing 3** gezeigt wird und im Download des Buches (siehe Kasten „Passende Produkte“) verfügbar ist, wurde bereits erläutert, mit einer Ausnahme, auf die Sie achten müssen: das **READ MODIFY WRITE**-Problem, das bei der Verwendung der **BSF**- und **BCF**-Befehle auftreten kann. Bis jetzt haben wir nur einen Ausgangspin geschaltet, indem wir den Bitwert auf **PORTA** gesetzt haben. Dies funktioniert, wenn Sie nur mit einem Ausgangspin an einem Port arbeiten, aber wenn Sie versuchen, einen Ausgang kurz nach der Änderung eines anderen Pins zu ändern, passiert Seltsames. Um das Problem zu verstehen, müssen wir uns an die Theorie bezüglich des Ladens und Entladens von Kondensatoren erinnern und uns fragen, wie der PIC das Ändern eines Pinwerts handhabt. Wenn der PIC auf die Ausgänge schreibt, kann er nur auf ein komplettes Byte beziehungsweise acht Bits schreiben. Um also ein Bit zu ändern, muss er den Zustand aller Pins lesen, den zu ändernden umschalten und dann die Ausgänge zurückschreiben. Das Problem besteht darin, dass zum Beispiel **PORTA** Bit 5 (physikalischer Pin 2) von 0 auf 1 geändert wurde und Sie dann in der nächsten Anweisung den Wert von **PORTA** Bit 4 (physikalischer Pin 3) ändern wollen. Der PIC liest die Werte an den **PORTA**-Pins – und Sie erwarten, dass der Wert von Bit 5 high ist, genau wie Sie ihn gerade eingestellt haben, aber nein, in der realen Welt sorgt die innere Kapazität des Pins dafür, dass es eine gewisse Zeit dauert, bis die Spannung am Pin tatsächlich high ist. Dies führt dazu, dass vom nächsten Befehl der Wert von Bit 5 als low gelesen wird! Und wenn die Ausgänge wieder an den Port zurückgeschrieben werden, wird Bit 5 (Pin 2) wieder auf low gesetzt. Dies ist ein Beispiel dafür, wie das Testen in einem Simulator Sie im Stich lassen kann. Deshalb sollten Sie bei Verwendung von **BCF** und **BSF** stets in die Latch-Register der Ports (**LATA** und **LATC**) schreiben. Wenn Sie jedoch auf den gesamten Port als Register schreiben, zum Beispiel mit **MOVWF PORTA**, ist dies eine sichere Methode.

PIC 12F1822 versus 16F1823

Der offensichtliche Unterschied besteht darin, dass der 12F1822 acht Anschlüsse besitzt und der 16F1823 14 Anschlüsse aufweist. Dies liegt daran, dass der 16F1823 zwei I/O-Ports besitzt. Abgesehen von diesem Unterschied ist die Programmierung fast gleich. Sie müssen nur die richtige Include-Datei am Anfang Ihres Quellcodes angeben und den richtigen Controller als Ihren PIC angeben:

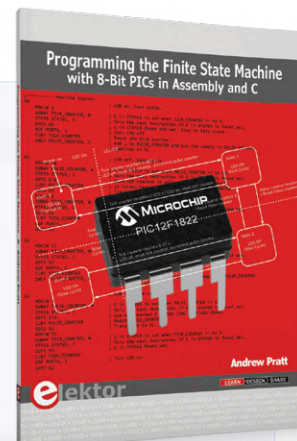


PASSENDEN PRODUKTE

Programmierung von Endlichen Zustandsautomaten mit 8-Bit-PICs in Assembler und C

In diesem Elektor-Buch führt Andrew Pratt detailliert in die Programmierung von PIC-Mikrocontrollern ein und gibt einen gründlichen Überblick über die Programmierung von Endlichen Zustandsautomaten (Finite State Machine, FSM).

- Druckversion:
www.elektor.de/programming-the-finite-state-machine
- E-Buch:
www.elektor.de/programming-the-finite-state-machine-e-book



```
LIST P=12F1822
#include <p12f1822.inc>
```

```
LIST P=16F1823
#include <p16f1823.inc>
```

Wenn Sie den zusätzlichen Port nicht benötigen, besteht der Vorteil eines kleineren Chips darin, dass er weniger Platz auf einer Platine einnimmt und es weniger Pins zu löten gibt. Sie müssen das Datenblatt konsultieren, da es immer Fallen gibt, in die man fallen kann. Es wird Gelegenheiten geben, bei denen Sie auf Ihren Code starren und versuchen zu verstehen, was Sie falsch gemacht haben, aber die Antwort wird im Datenblatt zu finden sein!

Interrupts und Zustandsdiagramme

Ich habe nicht versucht, Interrupts in irgendeiner Weise in den Zustandsdiagrammen darzustellen, da sie den Zustand der Maschine nicht verändern. Wenn ein Interrupt auftritt, wird das Hauptprogramm angehalten, wie es im vorherigen Abschnitt des Buchs erläutert wurde. Daher erfolgt keine Zustandsänderung bis zur Rückkehr vom Interrupt und dann nur, wenn sich im Interrupt die Eingabe in die Zustandsmaschine geändert hat (wie eine Erhöhung eines Zählers, der das Ergebnis des Vergleichs ändert).



200447-03

Ein Beitrag von
Autor: **Andrew Pratt**
Redaktion: **Jan Buiting**

Übersetzung:
Rolf Gerstendorf
Gestaltung: **Giel Dols**

Sie haben Fragen oder Kommentare?

Haben Sie Fragen oder Anmerkungen zu diesem Artikel? Bitte nehmen Sie Kontakt mit Elektor auf: editor@elektor.com.

WEBLINK

- [1] **Assembler-Programmdateien** (nach unten scrollen zu Downloads):
www.elektor.de/programming-the-finite-state-machine

Der Elektor Store

Nie teuer, immer überraschend!

Der Elektor Store hat sich vom Community-Store für Elektor-eigene Produkte wie Bücher, Zeitschriften, Bausätze und Module zu einem umfassenden Webshop entwickelt, der einen großen Wert auf überraschende Elektronik legt.

Wir bieten die Produkte an, von denen wir selbst begeistert sind oder die wir einfach ausprobieren wollen. Wenn Sie einen Produktvorschlag haben, sind wir hier erreichbar (sale@elektor.com).

Unsere Bedingungen:

Nie teuer, immer überraschend!

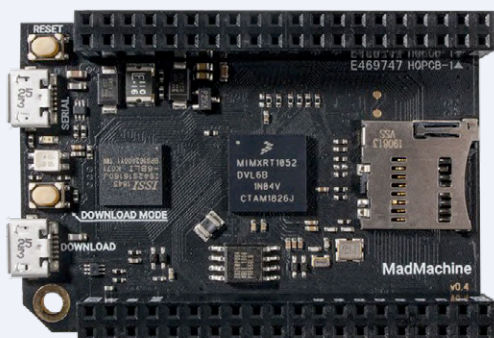


Raspberry Pi 4 Starterkit

Preis: 79,95 €

Mitgliederpreis: 71,96 €

🛒 www.elektor.de/19427

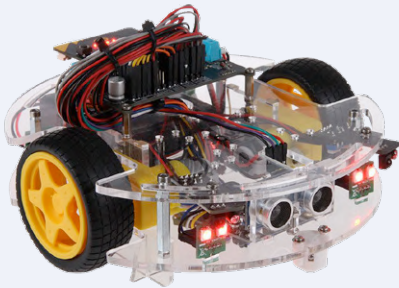


SwiftIO – Swift-basierendes Mikrocontroller-Board

Preis: 72,95 €

Mitgliederpreis: 65,66 €

🛒 www.elektor.de/19426



Joy-Car Roboter (inkl. BBC micro:bit)

Preis: 84,95 €

Mitgliederpreis: 76,46 €

www.elektor.de/19408



Elektronik-Projekte mit dem SensorTile.box Bundle

Preis: 69,95 €

Mitgliederpreis: 62,96 €

www.elektor.de/19404

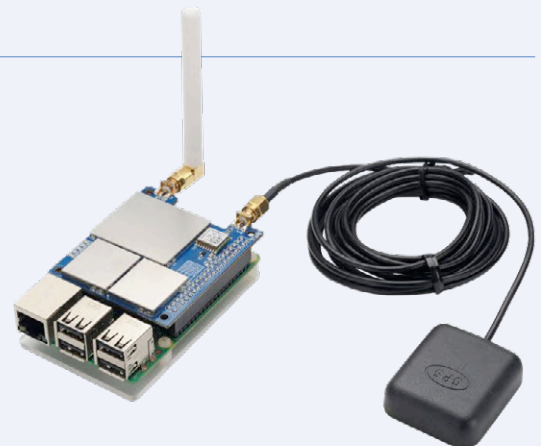


Peak UTP05E Netzwerkkabel- Analyser (Ultimate Atlas IT Kit)

Preis: 259,00 €

Mitgliederpreis: 233,10 €

www.elektor.de/19373



Dragino PG1301 LoRaWAN GPS Konzentrador für Raspberry Pi (868 MHz)

Preis: 124,95 €

Mitgliederpreis: 112,46 €

www.elektor.de/19367

Hexadoku

Sudoku für Elektroniker

Wie in jeder Ausgabe finden Sie auch in diesem Heft unser ganz spezielles Sudoku. PC, Oszilloskop und Lötkolben können sich erholen, während Ihre kleinen grauen Zellen auf Hochtouren arbeiten. Wenn Sie alle Hex-Ziffern in den grauen Kästchen herausgefunden haben, sollten Sie uns diese gleich zumailen – denn hier warten fünf Elektor-Gutscheine!

Die Regeln dieses Rätsels sind ganz einfach zu verstehen: Bei einem Hexadoku werden die Hexadezimalzahlen 0 bis F verwendet, was für Elektroniker und Programmierer ja durchaus passend ist. Füllen Sie das Diagramm mit seinen 16 x 16 Kästchen so aus, dass alle Hexadezimalzahlen von 0 bis F (also 0 bis 9 und A bis F) in jeder Reihe, jeder Spalte und in jedem

Fach mit 4 x 4 Kästchen (markiert durch die dickeren schwarzen Linien) **genau einmal** vorkommen. Einige Zahlen sind bereits eingetragen, was die Ausgangssituation des Rätsels bestimmt. Wer das Rätsel löst – sprich die Zahlen in den grauen Kästchen herausfindet – kann einen von fünf Gutscheinen im Wert von 50 Euro gewinnen!



EINSENDEN

Schicken Sie die Lösung (die Zahlen in den grauen Kästchen) per E-Mail, Fax oder Post an:

Elektor Redaktion
Kackertstr. 10
52072 Aachen

Fax: 0241 / 955 09-013

E-Mail: hexadoku@elektor.de

Als Betreff bitte nur die Ziffern der Lösung angeben!

Einsendeschluss ist der 8. Dezember 2020.

Die Gewinner des Hexadokus aus der Ausgabe September/Okttober 2020 stehen fest!

Die richtige Lösung ist: **35904**

Einen Elektor-Wertgutschein über je 50 € haben gewonnen:

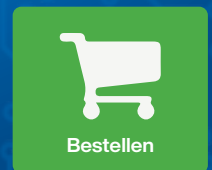
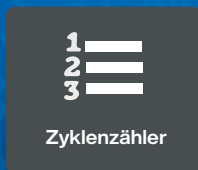
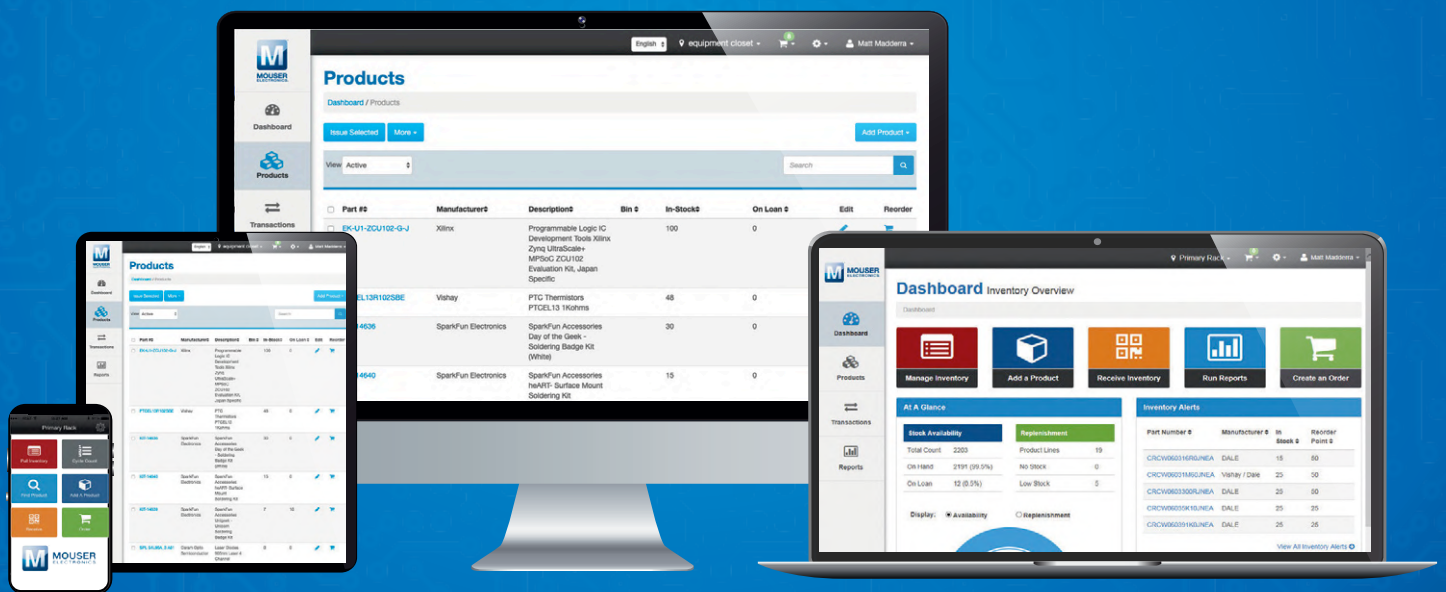
Charlotte Mies, Jean-Marie Mahieu, Roberto Visentin, Herman Pusch und Neil Wood.

Herzlichen Glückwunsch!

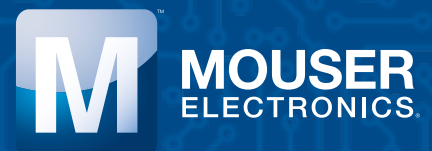
	B	3			E	4	8		2		7	1	C		
		6			B		2		3			F			
C						6	A	B		F					2
9									C	8	0			5	
	5	B	7			D	0			2		4	E	C	
		8				1				6	B			7	
E	0						B				8		6		A
	3			A			C		D	1				F	0
1	C				4	B		3			A			2	
6		A		C				7						D	3
	8			6	9				1				A		
	2	5	B		1			F	4			9	7	0	
		2			A	3	5								C
B					2		6	1	5						D
			3			E		2		A			0		
		0	D	B		7		8	6	9			2	3	

0	F	7	6	8	4	D	B	1	E	A	5	9	C	2	3
E	9	1	A	3	6	F	7	8	B	C	2	5	0	4	D
8	2	C	3	9	A	0	5	F	7	D	4	1	B	E	6
B	D	4	5	1	2	C	E	3	6	9	0	F	7	A	8
9	3	6	8	B	5	2	A	E	1	0	C	D	F	7	4
A	4	5	D	0	7	1	3	9	F	B	8	6	2	C	E
1	0	B	E	C	8	6	F	4	D	2	7	3	5	9	A
7	C	F	2	D	9	E	4	A	3	5	6	B	8	0	1
C	1	2	7	E	D	4	8	B	0	6	9	A	3	5	F
D	A	8	9	F	B	5	0	C	2	4	3	E	1	6	7
F	E	0	B	6	3	7	9	D	5	1	A	C	4	8	2
5	6	3	4	2	1	A	C	7	8	E	F	0	9	D	B
2	8	D	C	7	F	B	6	0	9	3	E	4	A	1	5
3	5	9	0	4	E	8	1	6	A	7	B	2	D	F	C
4	7	E	1	A	0	3	2	5	C	F	D	8	6	B	9
6	B	A	F	5	C	9	D	2	4	8	1	7	E	3	0

Kostenloses Lagerverwaltungstool



mouser.de/inventory-management



Treten Sie jetzt der Elektor Community bei!

Jetzt



Mitglied werden!



- ✓ Komplettes Webarchiv ab 1970
- ✓ 6x Elektor Doppelheft (Print)
- ✓ 9x Digital (PDF) inkl. Elektor Industry
- ✓ 10% Rabatt im Online-Shop und exklusive Angebote
- ✓ Elektor Jahrgangs-DVD

- ✓ Mit Tausenden von Mitgliedern des Online-Labors gemeinsam entwickeln mit Zugang zu über 1.000 Gerber-Dateien und direktem Kontakt zu unseren Experten!
- ✓ Veröffentlichen Sie Ihr eigenes Projekt oder verkaufen Sie direkt über unseren Shop!

Auch erhältlich

Die digitale
Mitgliedschaft!



- ✓ Zugang zu unserem Webarchiv
- ✓ 10% Rabatt in unserem Online-Shop
- ✓ 6x Elektor Doppelheft (PDF)
- ✓ Exklusive Angebote
- ✓ Zugang zu über 1.000 Gerber-Dateien



www.elektor.de/mitglied