G3078 E (D) € 14,90 | CHF 27,00 | (A, B, L) € 16,00

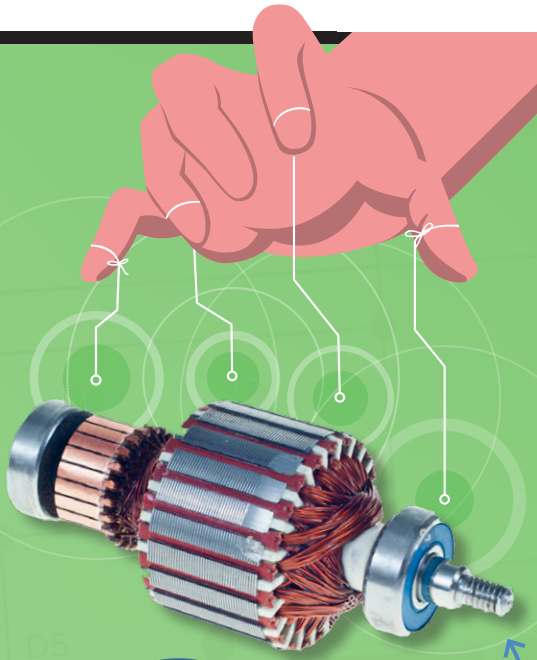
elektor MAG

design › share › sell

583 JANUAR/FEBRUAR 2022
ELEKTORMAGAZINE.DE

Motorsteuerung mit H-Brücken

S. 6



IM FOKUS

Motor-STEUERUNG

Vielseitiger Servo-Tester

S. 73

Größere Elektromotoren

S. 52

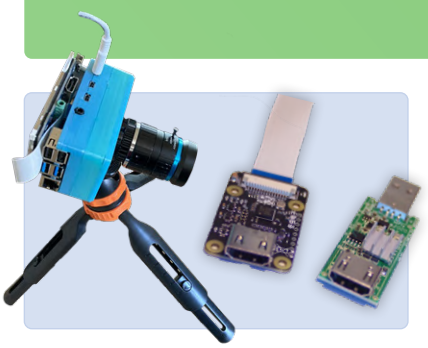
In dieser Ausgabe

- Hauptnetzschalter für den Labortisch selbst gebaut
- Raspberry Pi Zero 2 W mit vier Kernen
- Das Team im Elektor-Labor
- Soundkarte für die Raspberry-Pi-Familie
- Praktische Neuronen
- Im Inneren eines Open-Source-Prozessors

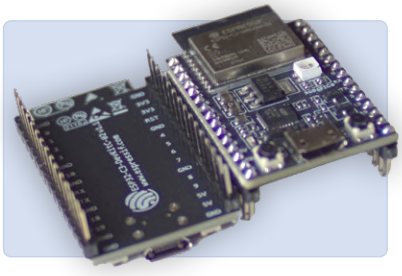
ELEKTOR INDUSTRY

- Wie die Motorsteuerung weniger kompliziert wird
- Impressionen vom World Ethical Electronics Forum

und vieles mehr!



Raspberry Pi als KVM-Fernsteuerung Software im Praxistest S. 22



ESP32-C3: 32-Bit-RISC-Einkerner Ein erstes Hands-on im Elektor-Labor S. 59

Autonomes Fahrzeug mit 2D-Lidar Fernsteuerung über Bluetooth S. 36

4 198630 314904 20002

UNSER SORTIMENT
VON TECHNIKERN FÜR TECHNIKER

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

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.

Unschlagbares Preis-Leistungs-Verhältnis für professionelles Pick-and-Place

UFactory xArm5 Lite

Ob Positionieraufgaben, Baugruppenmontage oder als vollautomatisierter Bartender: Mit Roboterarmen der xArm-Serie bringen Sie Ihr Automatisierungskonzept auf ein neues Level, und das zu überschaubaren Kosten.

- Freiheitsgrade: 5
- Wiederholgenauigkeit: 0,1 mm
- Reichweite: 700 mm
- Tragkraft: 3 kg
- System: ROS, Python, C++
- Protokoll: RS485

UFACTORY

Bestell-Nr.:
UF XARM5 LITE

6.545,00

Entdecken Sie unser umfangreiches Robotik-Sortiment ▶
www.reichelt.de/robotik

STUDIE:
Künstliche Intelligenz: Nahezu die Hälfte der Deutschen würde Implementierung nach heutigem Stand anders angehen.

Jetzt informieren
▶ https://rch.lt/MG726

- Top Preis-Leistungs-Verhältnis
- über 120.000 ausgesuchte Produkte
- zuverlässige Lieferung – aus Deutschland in alle Welt

www.reichelt.de
Bestellhotline: +49 (0)4422 955-333

reichelt elektronik – The best part of your project

Es gelten die gesetzlichen Widerrufsregelungen. Alle angegebenen Preise in € inklusive der gesetzlichen MwSt., zzgl. Versandspesen 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: 15. 12. 2021

IMPRESSUM

53. Jahrgang, Nr. 583
Januar/Februar 2022
ISSN 0932-5468

Erscheinungsweise: 8x jährlich

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
Raoul Morreau (Leitung)
Mobil: +31 6 440 399 07
E-Mail: raoul.morreau@elektor.com

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

Es gilt die Anzeigenpreisliste ab 01.01.2021.

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.

© 2022 elektor international media b.v.

EDITORIAL

Von Jens Nickel

Chefredakteur ElektorMag

Motoren steuern und mehr

Aus Umfragen wissen wir, dass die meisten unserer Leser auch beruflich Elektroniker sind - sie sind in der Entwicklung, dem Einkauf oder dem Vertrieb von Komponenten und Geräten tätig. Wir waren daher schon immer ein Magazin, das sich der üblichen Einteilung in Hobby- und Profi-Elektronik entzogen hat. Denn so mancher Berufs-Entwickler holte sich wertvolle Anregungen auch und gerade aus unseren spielerisch präsentierten Projekten. Und in Zeiten, in denen Maker eigene Unternehmen gründen oder von Entwicklungsabteilungen großer Halbleiterfirmen umworben werden, ist das alte Schwarzweiß-Denken wohl endgültig obsolet. Vor einigen Jahren haben wir unser Industry Special (zuerst unter dem Namen „Elektor Business") gestartet. Hier erfährt man aus erster Hand, was die Entwicklungsabteilungen der Hersteller gerade zur Marktreife gebracht haben. Zu den neuesten Trends in der Industrie gibt es darüber hinaus interessante Überblicksartikel, zum Beispiel von Stuart Cording. Der Ingenieur ist für uns auf Messen unterwegs und hat einen hervorragenden Draht zu vielen Experten, die ihn mit Hintergrundinfos versorgen.
Im Jahr 2022, beginnend mit diesem Heft, kehren wir zu einem einheitlichen Konzept zurück und integrieren die interessanten Artikel aus der Branche in das Elektor Magazin. Mit einer stärkeren Fokussierung auf ein Hauptthema - ohne auf die Abdeckung des gesamten Spektrums der Elektronik in unseren Ausgaben zu verzichten - können wir verschiedene Bereiche umfassender und tiefgehender abdecken.
In dieser Ausgabe beginnen wir mit dem Schwerpunkt „Motorsteuerung". Auf den Seiten 6, 52 und 73 finden Sie zwei Grundlagen-Artikel von Mathias Claußen und Thomas Scherer sowie (natürlich!) ein Schaltungsprojekt zum Thema. Und auf den Industry-Seiten zeigt Stuart Cording in einem weiteren Artikel, wie sich Motorsteuerungen vereinfachen lassen.
Bleiben Sie neugierig!

Unser Team

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

DEUTSCHE FACHPRESSE

Elektor ist Mitglied des 1929 gegründeten VDZ (Verband Deutscher Zeitschriftenverleger), der „die gemeinsamen Interessen von 500 deutschen Consumer- und B2B-Verlagen vertritt."

FIPP Connecting Global Media

Elektor ist Mitglied von FIPP, einer Organisation, die „über fast 100 Jahre gewachsen ist und Medienbesitzer und Content-Ersteller aus der ganzen Welt umfasst".

Motorsteuerung mit H-Brücken

Für DC-, Schritt- und bürstenlose Motoren

6

Rubriken

3 **Impressum**

72 **productronica Fast Forward 2021**
Die Preisträger

92 **Aller Anfang ...**
Die Spule lässt uns nicht los!

96 **Projekt 2.0**
Korrekturen, Updates und Leserbriefe

110 **Ethics**
Armut und Elektronik

114 **Hexadoku**
Sudoku für Elektroniker

Hintergrund

FOCUS

6 **Motorsteuerung mit H-Brücken**
Einführung in DC-, Schritt- und bürstenlose Motoren

16 **Das Team im Elektor-Labor**
Unser Ansatz, unsere bevorzugten Werkzeuge und mehr

28 **Review: IQaudio Codec Zero**
Soundkarte für die Raspberry-Pi-Familie

31 **Das PiKVM-Projekt und seine Lehren**
Interview mit Maxim Devaev (Entwickler von PiKVM)

40 **Raspberry Pi Zero 2 W mit vier Kernen**

FOCUS

52 **Größere Elektromotoren**
Prinzipien und Wissenswertes

68 **GUIs programmieren mit Python**
Teil 2: Nickname-Generator

82 **Neuronen in neuronalen Netzen verstehen**
Teil 3: Prakische Neuronen

87 **Im Inneren eines Open-Source-Prozessors**
Beispiel-Kapitel: Lattice- und Xilinx-FPGAs im Vergleich

102 **BattLab One**
Batterie-Betriebsdauer messen und optimieren

Industry

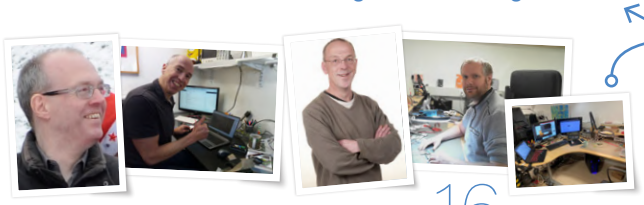
44 **Impressionen vom World Ethical Electronics Forum**

FOCUS

46 **Motorsteuerung**
Wie die Motorsteuerung weniger kompliziert wird

Das Team im elektor-LABOR

Unser Ansatz, unsere bevorzugten Werkzeuge und mehr



16

Raspberry Pi Zero 2 W **mit 4 Kernen**

40

Schützen Sie sich selbst und andere!

Hauptnetzschalter für den Labortisch selbst gebaut

64

Projekte

22 Raspberry Pi als KVM-Fernsteuerung
Die Software PiKVM im Elektor-Labor-Test

36 Autonomes Fahrzeug mit 2D-Lidar
ESP32 Pico interpretiert die Daten des Lidar-Moduls

59 Einstieg in den ESP32-C3: 32-Bit-RISC-Einkerner

64 Schützen Sie sich selbst und andere!
Hauptnetzschalter für den Labortisch selbst gebaut

FOCUS

73 Vielseitiger Servo-Tester
Servos ohne Datenblatt analysieren

78 Modbus über WLAN
Teil 2: Software für das Modbus-TCP-WLAN-Modul

98 Color to Sound
Wie man einen Farbsensor über I2C ausliest

106 Einfacher Erdschlussprüfer
Isolationstester für Netzspannungsinstallationen

Vorschau

Elektor März/April 2022
Das nächste Heft ist wie immer randvoll gefüllt mit Schaltungsprojekten, Grundlagen sowie Tipps und Tricks für Elektroniker.

Aus dem Inhalt:

- Workshop: RISC-V-Core auf einem FPGA
- Buffer-Board für den Raspberry Pi 400
- 4-Bit- und andere Low-Cost-Controller
- Frühlingskollektion der Raspberry-Pi-RP2040-Boards
- Portable Geräte zur Temperatur- und Feuchtemessung
- Audiomessungen mit dem USB-Audio-Interface
- WinUI 3: Neues Grafisches Framework für Windows-Apps
- Over-the-Air-Updates für Arduino und ESP
- NB-IoT in der Praxis

Und vieles mehr!

Elektor März/April 2022 erscheint am 10. März 2022.
Änderungen vorbehalten.

Einfacher Erdschlussprüfer
Isolationstester für Netzspannungsinstallationen

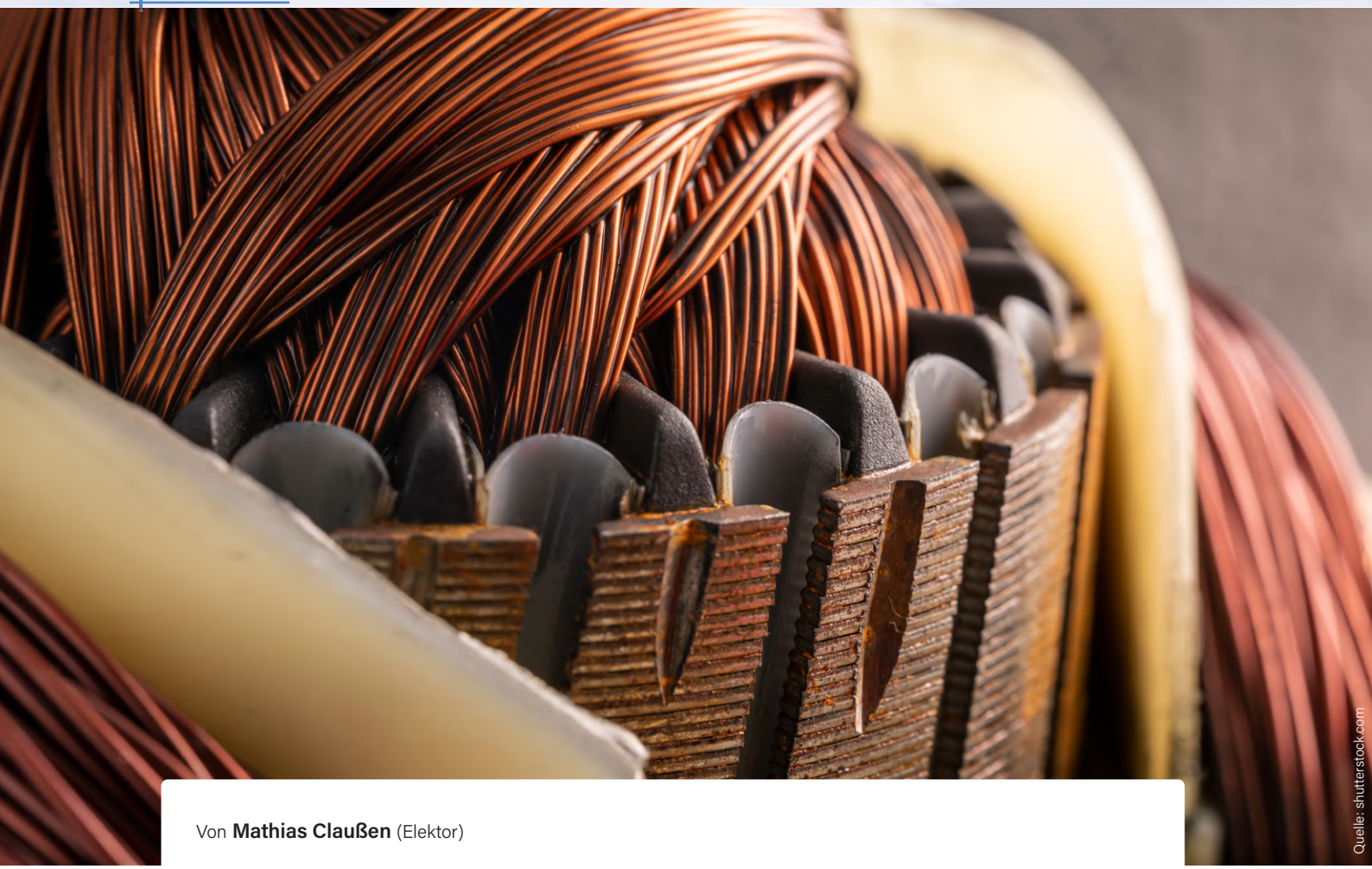
106

FOCUS ON Motor Control

Motorsteuerung mit H-Brücken

Für DC-, Schritt- und bürstenlose Motoren

Motor Control



Quelle: shutterstock.com

Von **Mathias Claußen** (Elektor)

Die Steuerung von Motoren wird in vielen Bereichen benötigt, nicht nur für Ventilatoren oder Lüftungssysteme. Auch Förderbänder, Pumpen oder 3D-Drucker sind mit Motoren ausgestattet, die nicht nur einfach ein- oder ausgeschaltet werden müssen. Auch die Leistung und Drehrichtung müssen gesteuert werden. Um den Einsteig einfach zu halten, sollen hier nur klassische DC-Motoren, Schrittmotoren und bürstenlose Motoren kleiner Leistung behandelt werden.

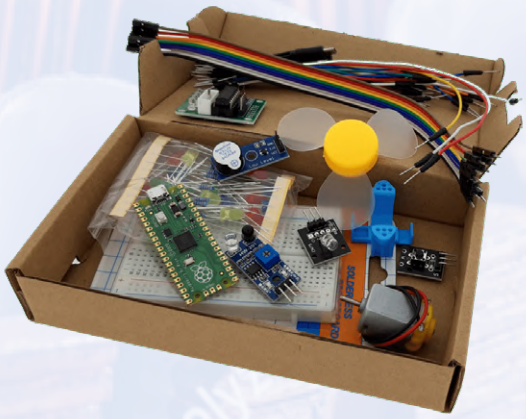


Bild 1. Das Basic Kit for Raspberry Pi Pico mit DC-Motor. (Quelle: Kuongshun)

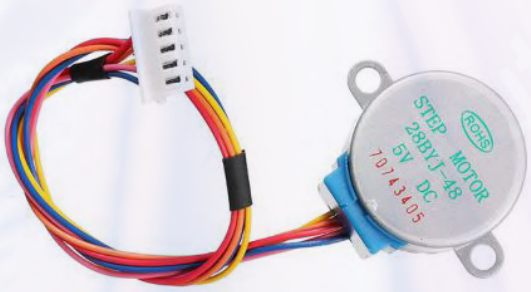


Bild 2. 28BYJ-48 Schrittmotor für 5 V DC. (Quelle: Geekcreit)

Die hier gezeigten Schaltungen dienen der Illustration der einzelnen Abschnitte und sollten so nicht 1:1 nachgebaut werden, sondern als Gedankengrundlage für eigene Entwürfe dienen.

Zuerst soll ein einfacher DC-Motor betrachtet werden. In unserem Fall darf dieser mit 3 V bis 6 V betrieben werden und benötigt maximal 1 A, ein Motor, wie er auch in vielen Starterkits wie dem Basic Kit für Raspberry Pi Pico [1] (**Bild 1**) zu finden ist. Als Spannungsversorgung werden hier 5 V verwendet, wie sie ein USB-Port bereitstellen würde. Etwas, was einigen dieser Kits auch beiliegt oder preiswert als Ergänzung hinzugekauft werden kann, ist ein Schrittmotor. Ein gängiger Typ ist der 28BYJ-48 [2] (**Bild 2**), der direkt mit 5 V versorgt werden kann.

Vorwärts und Rückwärts

Im ersten Schritt schauen wir uns die Schaltung in **Bild 3** an. Unser Relais K1 dient als Schalter, um den Motor an- oder auszuschalten. Gesteuert wird das Relais durch den Taster S1, so dass der Strom, der für den Motor notwendig ist, nur durch das Relais fließt und nicht durch den Taster. Die Freilaufdiode D1 soll verhindern, dass die Spule des Relais beim Öffnen des Tasters eine Überspannung erzeugt.
Wird der Taster S1 gedrückt, dann zieht das Relais K1 an und der Motor M1 kann anlaufen. Wird der Taster S1 losgelassen, so fällt das Relais ab und der Motor wird nicht länger mit Spannung versorgt. Die Spulen der DC-Motoren selbst können auch eine Überspannung erzeugen, genau wie die Spule im Relais. D2 dient hier als Freilaufdiode für den Motor und soll dafür sorgen, das dieser nicht ungewollt beim Abschalten eine Spannungsspitze erzeugt. Mit dieser Schaltung kann der Motor aber nur vorwärts laufen. Was aber, wenn der Motor auch rückwärts laufen soll, zum Beispiel wie bei einer Winde? Dazu muss die Schaltung um einen Rückwärtslauf erweitert werden.
Hier bietet sich eine sogenannte H-Brücke an, wie sie in **Bild 4** zu sehen ist. Die Herkunft des Namens kann man am Aufbau im Schaltplan erahnen. Der Motor ist in der Mitte des H angeschlossen. Wird der Taster S1 bedient (Motor vorwärts), dann ziehen nun die Relais K1 und K4 an, so dass ein Strom in Richtung Pin 1 nach Pin 2 des Motors fließen kann. Wenn der Motor rückwärts laufen soll, kann die Taste S2 betätigt werden und die Relais K3 und K2 ziehen an. Der Strom kann nun vom Pin 2 des Motors zum Pin 1 des Motors fließen, und der Motor läuft rückwärts. Auf keinen Fall dürfen hier die Tasten S1 und S2 zur gleichen Zeit gedrückt werden! Sollte das Relais K1 und K2 oder K3 und K4 jemals zur gleichen Zeit geschlossen sein, so würde der Strom als Kurzschluss durch diese Relais fließen und könnte sie dabei zerstören.

Von langsam bis schnell

Mit der H-Brücke können wir nun die Richtung bestimmen, mit der sich unser Motor dreht, momentan jedoch noch nicht seine Geschwindigkeit. Relais unter Last zu schalten, vor allem bei Gleichstrom, verkürzt deren Lebenserwartung erheblich. Neben dem Nachteil der verkürzten Lebensdauer bei vielen Schaltspielen sind Relais weder kompakt noch energiesparsam.
Die erste offensichtliche Optimierung ist die Verwendung von Halbleitern. In diesem Fall schauen wir uns den Aufbau mit NPN- und PNP-Transistoren an. In **Bild 5** ist der modifizierte Schaltplan zu sehen. Die Relais wurden durch PNP-Transistoren (T1 und T3) sowie NPN-Transistoren (T2 und T4) ersetzt. Auch die Ansteuerung hat sich geändert. Der Schalter S1 sorgt nun dafür, dass die Basis von T1 und T2 mit den Vorwiderständen R1 und R2 gegen Masse gezogen wird, wenn der Schalter geschlossen wird, oder durch R5 zur Versorgungspannung, wenn dieser geöffnet ist. Der Schalter S2 bewirkt das gleiche für die Transistoren T3 und T4.
Im Ruhezustand der Schaltung, Schalter S1 und S2 offen, liegt an den Widerständen R1 bis R4 über die Widerstände R5 und R6 Versorgungspannung an. Damit werden T3 und T1 sehr hochohmig (im Megaohm-Bereich) und sperren. T2 und T4 hingegen sind durch den fließenden Basisstrom nun leitend und damit niederohmig. Somit sind beide Seiten des Motors mit Masse verbunden und es fließt kein Strom durch den Motor.
Was passiert nun, wenn der Schalter S1 geschlossen ist und der Schalter S2 offen? In diesem Fall liegt an den Widerständen R1 und R2 Massepotential an. Damit fließt auch kein Basisstrom mehr durch den Transistor T2, Basis und Emitter haben nun das gleiche Potential. Der Transistor T1 hingegen wird nun leitend, da ein Strom durch den Emitter in die Basis fließen kann. Damit erhält nun eine Seite des Motors eine positive Versorgungsspannung, während die andere auf Masse liegt, der Motor kann so nun anfangen zu laufen. Sollte der Taster S1 offen sein und S2 geschlossen, so wird der Transistor T3 leitend und

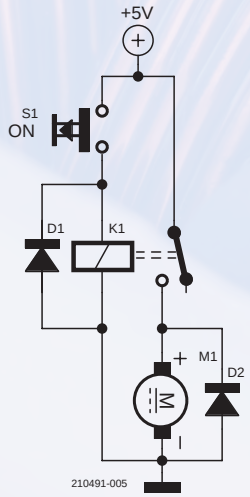


Bild 3. Motorsteuerung mit Relais.

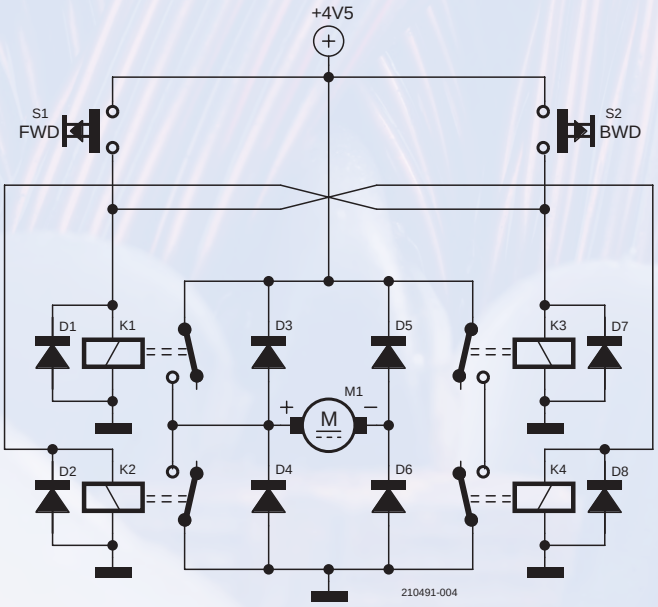


Bild 4. H-Brücke mit Relais.

T4 hochohmig. Der Motor beginnt so auch zu laufen, aber rückwärts. Was, wenn nun die Geschwindigkeit geregelt werden soll, zum Beispiel von einem Mikrocontroller? Anstatt des Tasters können auch zwei Pins eines Mikrocontrollers verwendet werden, die als *PWM_A* und *PWM_B* in **Bild 6** gekennzeichnet sind. Wenn wir nur vorwärts oder rückwärts wollen, reicht es, die Pins entsprechend auf Low oder High zu setzen. Soll nun die Geschwindigkeit geregelt werden, so kann eine Pulsweiten-Modulation (PWM) zum Einsatz kommen. Je nach Puls-Pausen-Verhältnis kann so die Geschwindigkeit bestimmt werden und in Abhängigkeit, ob PWM_A oder PWM_B gepulst werden, auch die Richtung. Dieser Aufbau ist für 5 V Versorgungsspannung ausgelegt, und somit müssen auch die Ausgänge der MCU mit 5-V-Pegeln liefern können. Wenn nun der Motor mit einer höheren Spannung betrieben werden soll beziehungsweise die MCU (wie der Raspberry Pi Pico) nur 3,3 V an seinen I/O-Pins vertragen kann, so muss die Schaltung ein klein wenig angepasst werden.

Bild 7 zeigt nun zwei weitere Transistoren, T5 und T6, die mit einem Basiswiderstand durch die PWM angesteuert werden. Wegen dieser Transistoren können MCUs verwendet werden, die mit einer Versorgungspannung arbeiten, die kleiner als die Spannung ist, die für den DC-Motor bereitsteht. Dabei ist zu beachten, dass nun das PWM-Signal invertiert wird. Liegt eine Spannung an T5 oder T6 an, so werden diese leitend und ziehen damit die Spannung am Eingang der Transistoren T2 und T1 oder T3 und T4 auf Masse. Dies muss bei der Ansteuerung durch die MCU berücksichtigt werden, da nun vorwärts und rückwärts vertauscht sind.

PNP- und NPN-Transistoren haben aber hier in der Anwendung Nachteile. Sie benötigen einen Basisstrom und damit Energie, auch ist die Frequenz der PWM begrenzt. Da der Wechsel von hoch- zu niederohmig und zurück nicht in Nullkommanichts geschieht, kommt es zu Schaltverlusten. Überdies gibt es zu dem Zeitpunkt, an dem T1 und T2 oder T3 und T4 ihren Zustand wechseln, einen Moment, bei dem beide Transistoren dieser Pärchen bedingt leitend sind. Damit kann ein höherer Strom direkt von der Versorgungsspannung nach Masse fließen. Auch das erhöht die Verlustleistung und kann sogar zu einem frühen Ausfall führen.

H-Brücken mit FETs

Um einigen Nachteile der bipolaren Transistoren zu begegnen, bietet es sich an, Feldeffekttransistoren (FET) einzusetzen. Diese können

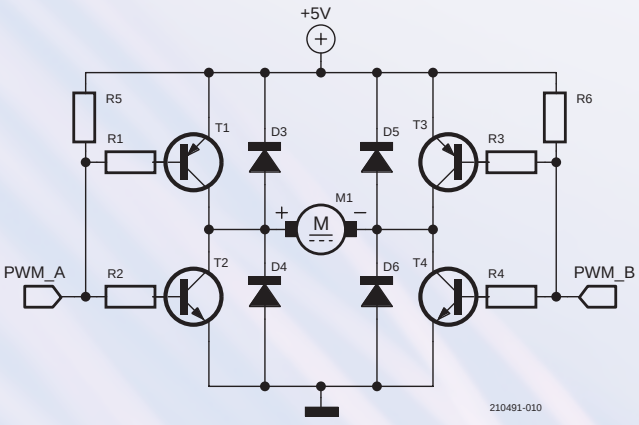


Bild 6. Ansteuerung durch die PWM einer MCU.

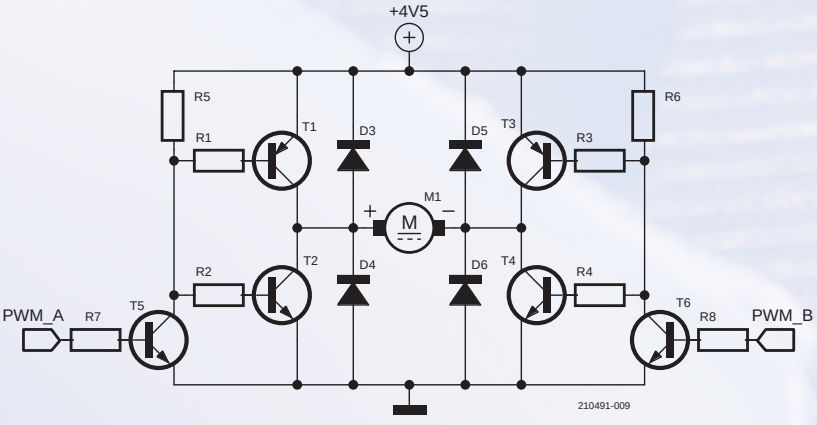


Bild 7. Ansteuerung nun mit 3,3 V

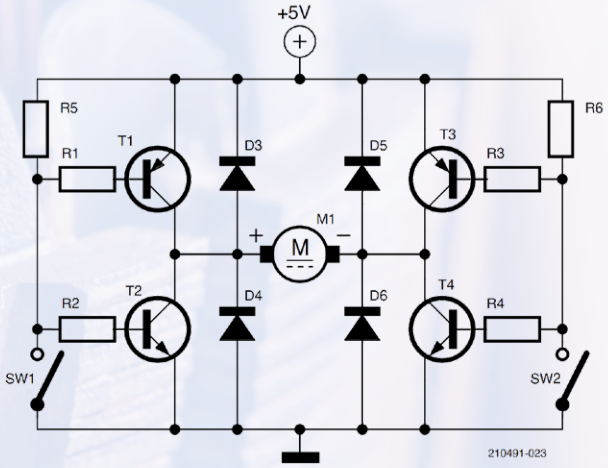


Bild 5. H-Brücke mit NPN- und PNP-Transistoren.

höhere Schaltfrequenzen erreichen und haben eine geringere Verlustleistung. Da bis jetzt bei den gezeigten H-Brücken immer noch das Problem besteht, dass die Transistoren für einen kurzen Moment einen Kurzschluss erzeugen können, gilt es dieses zu verhindern. Daher ist die H-Brücke in **Bild 8** mit FETs aufgebaut, die einzeln angesteuert werden können.

T1 und T3 sind hier PMOS-FETs, T2 und T4 sind NMOS-FETs. Ist die Spannungsdifferenz bei einem PMOS-FET zwischen Source und Gate nahe Null, so ist der FET hochohmig, sperrt also. Um den PMOS-FET leitend, also niederohmig zu bekommen, muss die Spannung am Gate negativer sein als an der Source. Für T1 und T3 bedeutet das also: Wenn das Gate auf Masse gezogen wird, ist der FET leitend. Dabei gilt es, die maximal zulässige Spannung zwischen Gate und Source zu beachten. Wird diese überschritten, so kommt es zu Beschädigungen des FETs. Bei einem NMOS muss die Spannung am Gate positiver sein als die an der Source, damit der FET leitend wird. Das bedeutet für T2 und T4, dass diese hochohmig sind, wenn sie auf Masse gezogen werden, und anfangen zu leiten, wenn am Gate eine positive Spannung (hier bezogen auf Masse) angelegt wird.

In Bild 8 sind auch noch die Dioden D4 bis D6 zu sehen. Beim Einsatz von FETs sind diese überflüssig, da ihre Funktion durch die Body-Dioden der FETs ausgeübt werden kann. Das Symbol (**Bild 9**) von T1 bis T4 verdeutlicht schon die Body-Diode als Bestandteil eines FETs. So kann man D4 bis D6 entfernen, so dass der Schaltplan in **Bild 10** entsteht. Könnte man bei nur 5 V nicht einfach PWM_A und !PWM_A sowie PWM_B und !PWM_B verbinden? Theoretisch wäre das möglich, würde aber, noch schlimmer als bei den bipolaren Transistoren, zu einem kurzzeitigen Kurzschluss führen. Als theoretisches Experiment gehen wir davon aus, dass nun PWM_A und !PWM_A verbunden sind. Mit 5 V an T1 und T2 als Ausgangsbedingung ist T1 leitend und T2 hochohmig. Nun wird PWM_A und !PWM_A auf Masse gezogen. Die 5 V fallen nun auf 0 V ab, jedoch nicht schlagartig, sondern, auch wenn die Zeit im Nanosekunden-Bereich liegt, als Kurve. Bei diesem Vorgang gibt es einen Zeitpunkt, an dem beide Transistoren leitend sind: T2 sperrt noch nicht komplett und T1 ist schon leitend. Beide Transistoren haben nun einen Widerstand von wenigen Ohm und erzeugen einen Kurzschlussstrom direkt durch T1 und T2. Dies wird als *shot-through* bezeichnet und ist in H-Brücken ungewollt, da die Bauteile zerstört werden. Also ist dringend von dem Versuch abzuraten, diese Signale zu verbinden. Moderne MCUs haben in ihren PWM-Einheiten

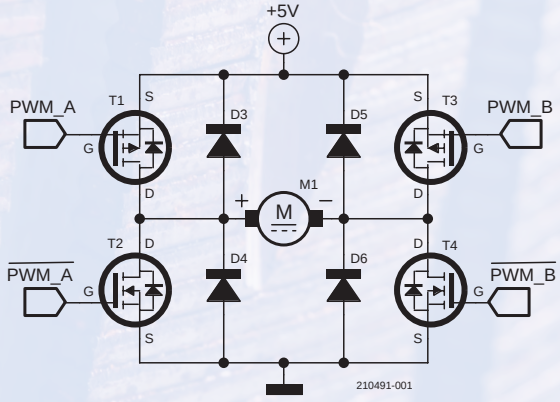


Bild 8. H-Brücke mit FETs.

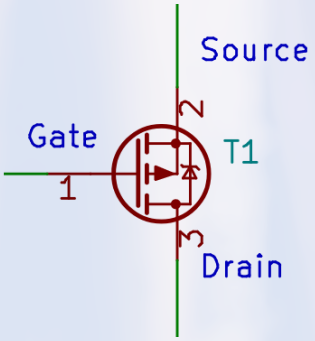


Bild 9. Symbol eines FET mit Body-Diode.

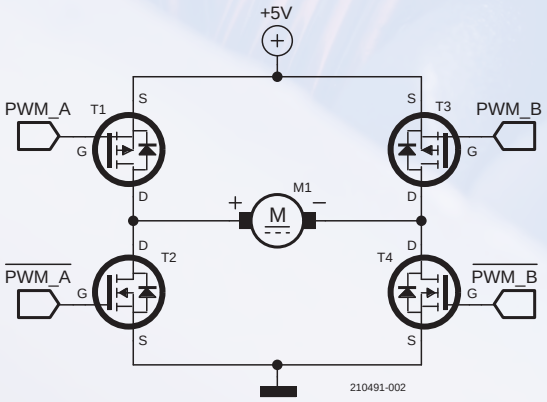


Bild 10. Schaltplan ohne Freilaufdioden.

teilweise die Option, passende Signale zur Ansteuerung von H-Brücken zu erzeugen und auch passende Totzeiten einzufügen, um ein *shot-through* zu verhindern.

H-Brücken Treiber

Man kann also durch die MCU Totzeiten erzeugen, so dass niemals T1 und T2 gleichzeitig leitend sind, oder man verwendet passende H-Brücken-Treiber-ICs.

Die vorgestellte H-Brücke mit zwei NMOS- und zwei PMOS-FETs hat den Nachteil, dass die PMOS-FETs um mindestens eine Zehnerpotenz schlechter leiten als ihre NMOS-Gegenstücke. Das bedeutet, dass auch bei höheren Strömen höhere Verluste auftreten. In **Bild 11** ist nun die Schaltung mit zwei Halbbrücken-Treibern (NCP5901B von ON Semiconductor) sowie ein Aufbau mit vier NMOS-FETs zu sehen. In dem Schaltplan sind keine Abblockkondensatoren eingezeichnet, die Kondensatoren C1 und C2 gehören zur Ladungspumpe des NCP5901B. Doch warum braucht der Halbbrückentreiber nun eine Ladungspumpe? T1 und T3 sind nun NMOS-FETs, die wie schon beschrieben eine Spannung am Gate benötigen, die größer als die an der Source ist, um leitend zu werden. Das bedeutet im Fall von T1 und T3, dass die Spannung am Gate größer sein muss als die Versorgungspannung der H-Brücke. Hierfür kommt die Ladungspumpe zum Einsatz. **Bild 12** zeigt die inneren Funktionsblöcke des H-Brücken-Treibers. Dort ist auch die Totzeitgenerierung zum Verhindern eines *shot-through* zu erkennen, bezeichnet als *Anti-Cross Conduction*.

So lässt sich eine H-Brücke aus diskreten Bauteilen mit passenden FETs für den jeweiligen Anwendungsfall aufbauen, mit der auch Ströme über 25 A mit der H-Brücke möglich sind. Wer eine kompaktere Lösung für kleinere Ströme sucht, kann zur einer komplett integrierten H-Brücke greifen.

Der STM L298N, der auf vielen preiswerten Motortreiber-Platinen sitzt (**Bild 13**), stellt zwei komplette H-Brücken bereit, jedoch nicht mit integrierten FETs, sondern noch mit bipolaren Transistoren. Dies erlaubt einen Treiberstrom von bis zu 4 A bei 46 V. Ein Blick in das Blockdiagramm (**Bild 14**) zeigt uns sehr vertraute Strukturen. Zur Ansteuerung des L298N sind MCUs mit 3,3-V-Pegeln durchaus geeignet. Das Modul kann bei Versandhändlern für etwa 3 € bis 6 € bestellt werden. Ein noch deutlich kompakterer Treiber ist der Infineon IFX9201SG, der 6 A treiben kann und über eine integrierte Schutzbeschaltung und Diagnosefunktionen verfügt, wie in **Bild 15** zu sehen ist. Er liegt preislich bei etwa 4 € pro Chip. Es wäre einfach, mit dem Chip einen DC-Motortreiber zu entwerfen, aber leider ist der Chip (Stand 1.9.2021) bis Mitte 2022 nicht lieferbar, wie so manch anderer Halbleiter in diesen Zeiten. Wer dennoch auf der Suche nach einer fertig aufgebauten H-Brücke für die Motorsteuerung ist, sollte sich deshalb einmal die Cytron-Motortreiber [3] aus dem Elektor-Store ansehen.

Schrittmotoren, Schritt für Schritt

Schrittmotoren laufen, wie der Name schon verrät, nicht einfach los, sondern führen immer einen Teil einer Umdrehung aus. Sie sind nicht für hohe Drehzahlen im Dauerbetrieb gedacht, sondern können einen definierten Teil einer Umdrehung mit hohem Drehmoment ausführen. In den meisten Haushalten dürften sich Schrittmotoren

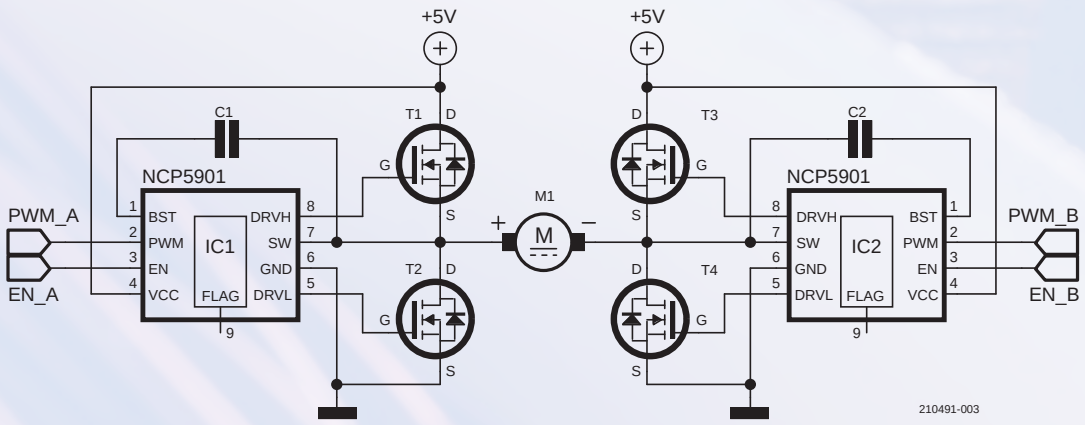


Bild 11. Vier NMOS-FETs mit Treibern.

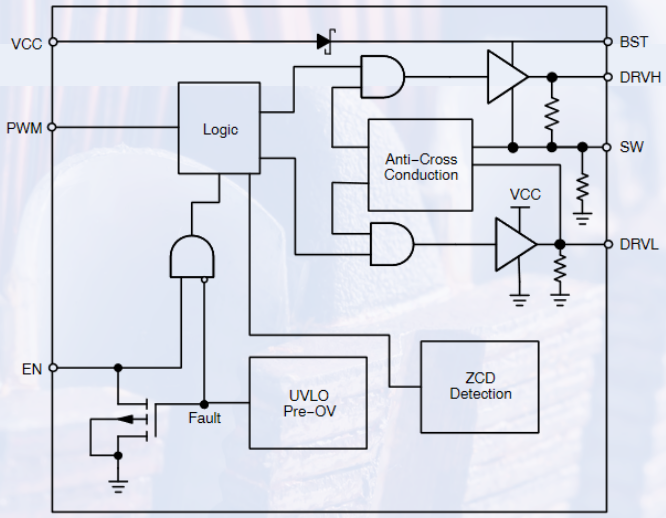


Bild 12. Interne Blöcke des Onsemi NCP5901B (Quelle: Onsemi).

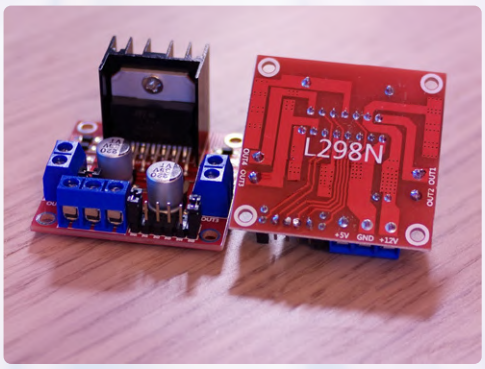


Bild 13. L298N-Module als Dual-H-Brücke.

in Tintenstrahldruckern finden lassen. Dort werden sie genutzt, um die Druckköpfe zu bewegen oder Papier zu transportieren. Durch die definierte Schrittweite, also den Teil einer Umdrehung, die ein Schrittmotor ausführt, kann sehr einfach errechnet werden, an welcher Stelle sich der Druckkopf befinden müsste. Einige werden den Klang eines Schrittmotors auch noch von Diskettenlaufwerken kennen, der dort den Schreib/Lese-Kopf bewegt hat.

Doch wie steuert man nun einen solchen Schrittmotor an und warum macht dieser überhaupt Schritte? Es gibt Reluktanz-, Permanentmagnet- und Hybrid-Schrittmotoren. Wir betrachten hier den Hybrid-Schrittmotor, da dieser sich durch sein gutes Drehmoment und hohe Schrittanzahl pro Umdrehung in vielen Anwendungsgebieten durchgesetzt hat. **Bild 16** zeigt den Aufbau eines solchen Schrittmotors. Im Inneren liegt der Rotor, eine Kombination aus einem Permanentmagneten und Weicheisenzahnkränzen. Der Stator besteht aus einer Reihe von Spulen, die dafür sorgen, dass sich der Rotor immer um einen Schritt bewegt. Bei der Ansteuerung sind am häufigsten Schrittmotoren mit zwei Polen anzutreffen, andere Polzahlen sind möglich, werden hier aber nicht betrachtet. Vom Aufbau her gibt es zwei Arten dieser Schrittmotoren, unipolare und bipolare, die wir uns nachfolgend ansehen wollen.

Unipolare Schrittmotoren

Als Beispiel für einen Schrittmotor verwenden wir hier einen 28BYJ-48 aus Bild 2**,** wie er in vielen Experimentierkits vorhanden ist. Dieser kleine unipolare Schrittmotor hat zusätzlich ein Getriebe mit einer 63,68395:1-Untersetzung. Der Motor besitzt eine Schrittweite von 11,25°, so dass sich für eine Umdrehung am Getriebeausgang 2037,8864 Schritte ergeben, grob gerundet also 2038 pro Umdrehung. Für den Anschluss werden fünf Adern benötigt: A,B,C,D und die Versorgungsspannung. Die Ansteuerung eines unipolaren Schrittmotors ist sehr einfach und kann bei dieser Größe mit einem ULN2003, der von unterschiedlichen Herstellern gebaut wird, geschehen. In dem ULN2003 sind lediglich sieben Darlington-Transistoren untergebracht, wie es im Blockschaltbild in **Bild 17** zu sehen ist. Zusätzlich sind in dem Chip Freilaufdioden integriert, so dass sich dieser sich bestens eignet, induktive Lasten zu treiben. **Bild 18** zeigt eine Schaltung mit einem ULN2003, der einen 28BYJ-48 Schrittmotor treiben kann. Der 28BYJ-48 kann aber nicht nur in ganzen Schritten gedreht werden, sondern erlaubt auch halbe Schritte, was laut Datenblatt die empfohlene Methode zur Ansteuerung ist. Die Sequenz zur Ansteuerung im Uhrzeigersinn kann der **Tabelle 1** entnommen werden. Soll ein Schritt entgegen dem Uhrzeigersinn gemacht werden, so muss die Sequenz rückwärts

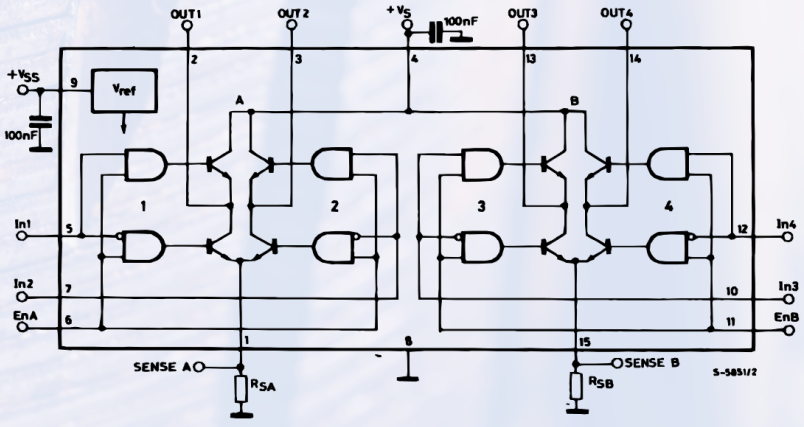


Bild 14. Intere Struktur des L294N (Quelle: STMicroelectronics).

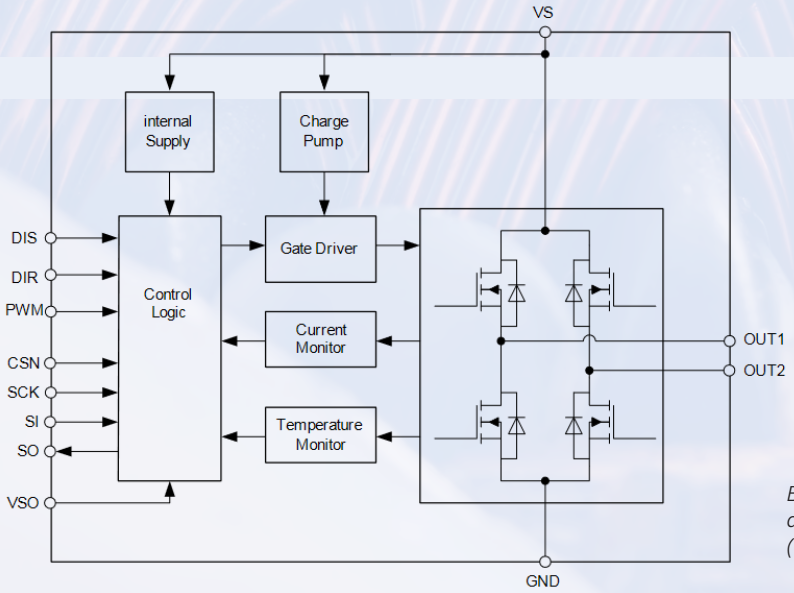


Bild 15. Interne Blöcke des Infineon IFX9201SG (Quelle: Infineon).

durchlaufen werden. Unipolare Schrittmotoren sind inzwischen selten geworden, da sie im Vergleich zu bipolaren Schrittmotoren nur etwa die Hälfte des Drehmoments bei gleicher Baugröße liefern können. Dafür ist aber der Aufbau der Treiberstufe mit einem ULN2003 denkbar einfach. Wer möchte, kann einen 28BYJ-48 Schrittmotor für den bipolaren Betrieb modifizieren, wie in dem Blog von jangeox [4] nachgelesen werden kann. Damit kann der Schrittmotor ein höheres Drehmoment erzeugen, benötigt dann aber eine andere Ansteuerung.

Bipolare Schrittmotoren

Bipolare Schrittmotoren, die in den meisten 3D-Druckern verbaut sind, haben nur vier Anschlüsse. Für diese Schrittmotoren gibt es Normgrößen nach NEMA (National Electrical Manufactures Assosiation). So bezeichnet NEMA17-03 die Baugröße des Schrittmotors, diese werden dann von unterschiedlichen Unternehmen produziert und/oder vertrieben. Der NEMA17-03 von Joy-IT [5] ist ein solcher Schrittmotor (**Bild 19**). Für die Ansteuerung ist ein ULN2003 nicht ausreichend, stattdessen werden zwei H-Brücken benötigt. Ein L298N, wie schon bei den DC-Motoren erwähnt, stellt zwei H-Brücken in einem Gehäuse zur Verfügung und ist in der Schaltung in **Bild 20** als Schrittmotortreiber im Einsatz. Da dies eine H-Brücke mit bipolaren Transistoren ist, sind am Ausgang noch acht Freilaufdioden nötig. Um einen Schritt im Uhrzeigersinn mit dem Motor zu erzeugen, muss die Sequenz aus **Tabelle 2** durchgeführt werden.

Anders als bei den unipolaren Schrittmotoren wird hier die komplette Spule genutzt und nicht nur die Hälfte bis zu einem Mittelabgriff. Das ermöglicht ein höheres Drehmoment als unipolare Schrittmotoren bei gleicher Baugröße. Die Ansteuerung mit H-Brücken ist sehr einfach und hochintegrierte Treiber wie der MP6500 von MPS oder der TMC2300-LA von Trinamic machen die Ansteuerung noch leichter. Diese ICs enthalten nämlich schon einen Teil der Kontrolllogik für die Schrittsequenzen. Die MCU gibt nur die Wunschrichtung vor, mit oder gegen den Uhrzeigersinn, und einen Impuls pro gewünschten Schritt. Das Schöne an den Schrittmotoren ist, dass sie ein hohes Drehmoment besitzen und es keine Bürsten oder Schleifer gibt, die einen Strom in Richtung Rotor transportieren müssten wie bei DC-Motoren. Wenn man die Eigenschaften eines DC-Motors mit dem Drehmoment und der Wartungsarmut eines Schrittmotors kombiniert, so erhält man einen bürstenlosen Motor, der sich perfekt als Antrieb in den unterschiedlichsten Anwendungen eignet.

Tabelle 1. Sequenz für einen Schritt

Schritt/ Farbe	Orange	Gelb	Pink	Blau
1.	1	0	0	0
2.	1	1	0	0
3.	0	1	0	0
4.	0	1	1	0
5.	0	0	1	0
6.	0	0	1	1
7.	0	0	0	1
8.	1	0	0	1

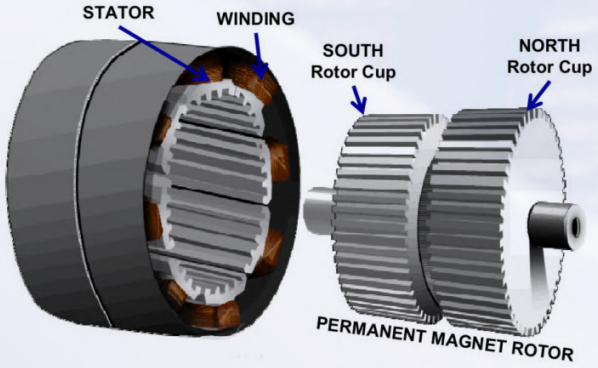


Bild 16. Aufbau eines Schrittmotors (Quelle: Microchip).

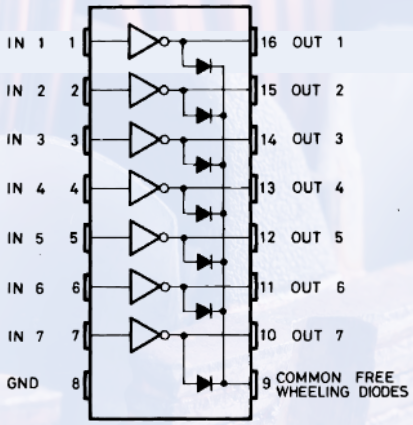


Bild 17. Interna des ULN2003 Darlington Arrays (Quelle: STMicroelectronics).

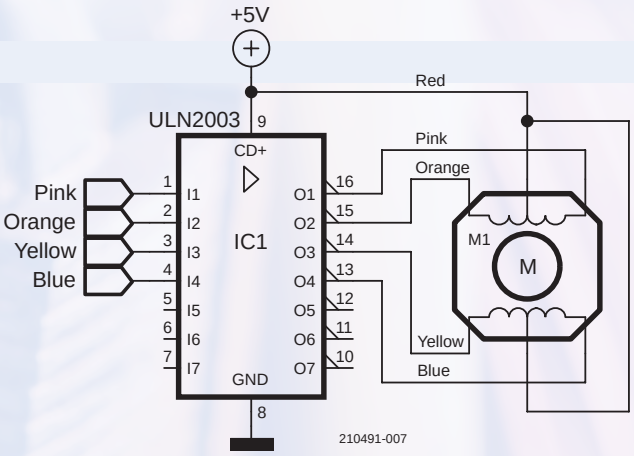


Bild 18. Schaltplan mit ULN2003 und Schrittmotor.

Bürstenlose DC-Motoren

Bei konventionellen DC-Motoren sorgen Bürsten für einen Stromfluss in den Wicklungen des Rotors. Da diese Bürsten bei jeder Umdrehung abnutzen und sich elektrisch leitender Staub im Motor verteilt, müssen DC-Motoren zyklisch neue Bürsten erhalten und eventuell eine Reinigung erhalten. Bürstenlose Motoren haben eine solche Wartung nicht nötig und liefern bei kompakter Bauweise ein hohes Drehmoment. Die andere Seite der Medaille ist eine alles andere als einfache Ansteuerung, bei der Fehler zu einer Beschädigung führen können. Daher hier auch nur ein sehr grober Auszug aus der Theorie zur Ansteuerung.

Der Aufbau eines bürstenlosen Motors erinnert stark an den eines Schrittmotors. Es gibt den nicht beweglichen Stator und einen entweder innen oder außen liegenden Rotor. Je nach Modell gibt es bürstenlose Motoren mit integrierten Hall-Sensoren, um die Lage des Rotors zu bestimmen. Motoren mit nur drei Anschlusskabeln haben diese Rückmeldung nicht, so dass ein wenig höherer Schaltungs- und Rechenaufwand betrieben werden muss. Für sensorlose Vertreter dieser Motoren ist die AVR444 Application Note [6] eine gute Basis, um sich etwas näher mit dem Thema zu beschäftigen.

Bild 21 zeigt den Aufbau eines bürstenlosen DC-Motors mit Außenläufer. Der Außenläufer hier ist ein Permanentmagnet, im Inneren befinden sich sechs Spulen, die in drei Paaren angeordnet sind. Jedes Paar ist so gewickelt, dass eine Spule den Nordpol und die andere den Südpol ausbildet, wenn ein Strom hindurchfließt. Die jeweilige Seite des Rotors wird sich an dem Spulenpaar ausrichten wollen, durch das gerade ein Strom fließt. **Bild 22** zeigt, wie der Rotor durch das Magnetfeld der Spulen gezogen und in Rotation versetzt wird. So wird jedoch momentan immer nur ein Spulenpaar von dreien benutzt, was sich noch optimieren lässt. Während im ersten Schritt das Spulenpaar A den Permanentmagneten anzieht, können wir das Spulenpaar C den Permanentmagneten abstoßen lassen. So tragen zwei Spulen zur Erzeugung der Rotation bei, und das Drehmoment und damit die Leistung des Motors erhöht sich. **Bild 23** zeigt die Rotation mit zwei angetriebenen Spulen. Die Ansteuerung muss durch ein passendes Timing erfolgen.

Die Ansteuerung der Spulen erfolgt durch drei Halbbrücken, wie sie schon aus dem Abschnitt über DC-Motoren bekannt sein sollten. Die Schaltung in **Bild 24** zeigt den theoretischen Aufbau dieser drei halben H-Brücken, die es von verschiedenen Herstellern auch als IC gibt. Bei der Ansteuerung des Motors müssen die Eingänge A bis C entsprechend mit Signalen versorgt werden. Bei dem NCP5901B

Tabelle 2. Sequenz für bipolaren Schrittmotor

	A	B	C	D
1.	1	1	0	0
2.	0	1	1	0
3.	0	0	1	1
4.	1	0	0	1

Bild 19. Joy-IT NEMA17-03 Schrittmotor (Quelle: Joy-IT).

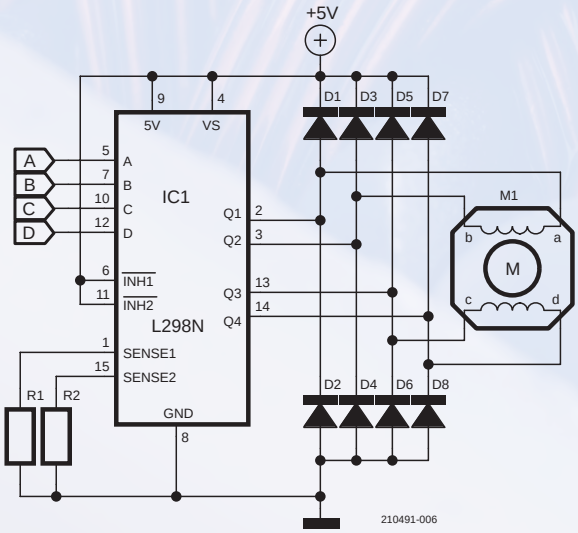


Bild 20. L294N als Schrittmotortreiber.

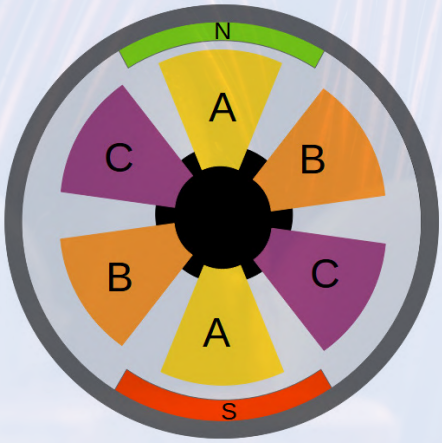


Bild 21. Aufbau eines bürstenlosen Motors.

sehen wir an den Eingängen einen Spannungsteiler, der den Pin auf 2,0 V bringt, wenn der Eingang von der MCU nicht getrieben wird. Im Bereich zwischen 1,3 V und 2,7 V wechselt der NCP5901B in die Dioden-Emulation, also den Freilaufmodus, damit die Spule weder an Masse noch mit der Versorgungsspannung des Motors verbunden ist. Das ermöglicht es später, die gegenelektromotorische Kraft (GEMK), in Englisch *back electromotive force* (BEMF) zu messen und die Rotorposition zu bestimmen.

Davon ausgehend, dass es schon eine Rotation gibt, und keine Hall-Sensoren im Motor verbaut sind, macht man sich die GEMK der nicht getriebenen Spule zunutze, um die Position des Rotors zu bestimmen oder besser der Zeitpunkt, an dem der Rotor sich soweit gedreht hat, dass er sich zwischen zwei Spulenpaaren befindet. In Bild 23 sind die drei Spulenpaare A, B und C eingezeichnet. Wird nun eine Spannung zwischen A und B angelegt, so kann man über die Spule C die Spannung zwischen den beiden Spulen abgreifen. Zu dem Zeitpunkt, an dem eine Spannung zwischen A und B angelegt wurde, wird ein Timer gestartet und die Spannung an der Spule C überwacht. Sobald die Spannung die Hälfte der Versorgungsspannung des Motors erreicht hat, hat der Rotor die Hälfte der Rotation zwischen zwei Spulenpaaren zurückgelegt. Mit dieser Zeit kann nun berechnet werden, wann die nächsten Spulen mit welcher Polarität angesteuert werden müssen. Dies sorgt dafür, dass der Permanentmagnet durch das nun rotierende Magnetfeld weiter in Rotation bleibt.

Die Geschwindigkeit des Motors kann durch eine PWM geregelt werden, die entsprechend an die Eingänge A, B und C der Treiber-ICs angelegt werden muss. Bei der Hardware, die in der *AVR444 Application Note* verwendet wird, handelt es sich um einen ATAVRMC100, dessen Beschreibung und Schaltplan sich auf der Microchip-Seite [7] finden lassen. Mehr Informationen liefert auch die *Application Note AN1946* von ST [8] oder die *Application Note AN12435* [9] von NXP. Bürstenlose Motoren anzusteuern bedarf etwas mehr Software, als es bei Schrittmotoren oder einfachen DC-Motoren der Fall ist. Die Ermittlung der Rotorposition, die interne Signalfilterung und Aufbereitung durch die Software machen die Ansteuerung weniger intuitiv als bei klassischen DC-Motoren.

Eine H-Brücke für alle Fälle

Eine H-Brücke ist eine wichtige Basis für DC-Antriebe, seien es Schrittmotoren, Motoren mit Bürsten oder ohne. Man muss die Brücke nicht

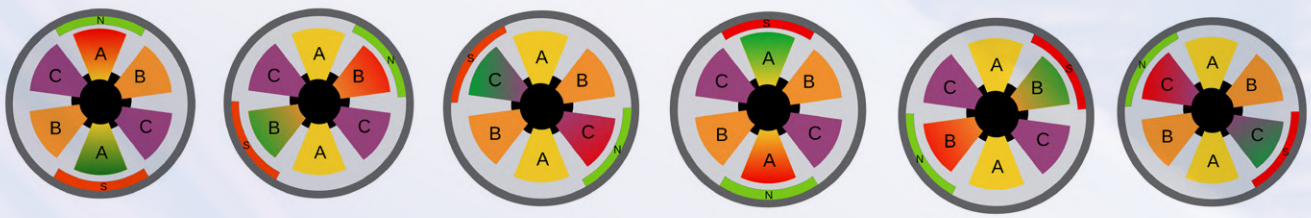


Bild 22. Der Rotor wird durch ein Magnetfeld gezogen.

Bild 23. Zwei von drei Spulenparen treiben nun den Rotor.

diskret aufbauen und kann bei kleineren Motoren und kleineren Strömen fertige ICs einsetzen, sofern verfügbar. Bürstenlose Motoren sind von der Ansteuerung her schwieriger und es ist zu überlegen, ob man selbst Erfahrungen mit dem Bau von Controllern für bürstenlose Motoren sammeln möchte oder es ausreichend ist, fertige Lösungen einzusetzen. Die grundlegende Theorie zu kennen, schadet aber sicherlich nicht!

210491-02

Ein Beitrag von

Entwurf und Text: **Mathias Claußen**
Redaktion: **Jens Nickel**
Layout: **Harmen Heida**

Sie haben Fragen oder Kommentare?

Haben Sie technische Fragen oder Kommentare zu diesem Artikel? Schicken Sie eine E-Mail an den Autor unter mathias.claussen@elektor.com oder kontaktieren Sie Elektor unter editor@elektor.com.

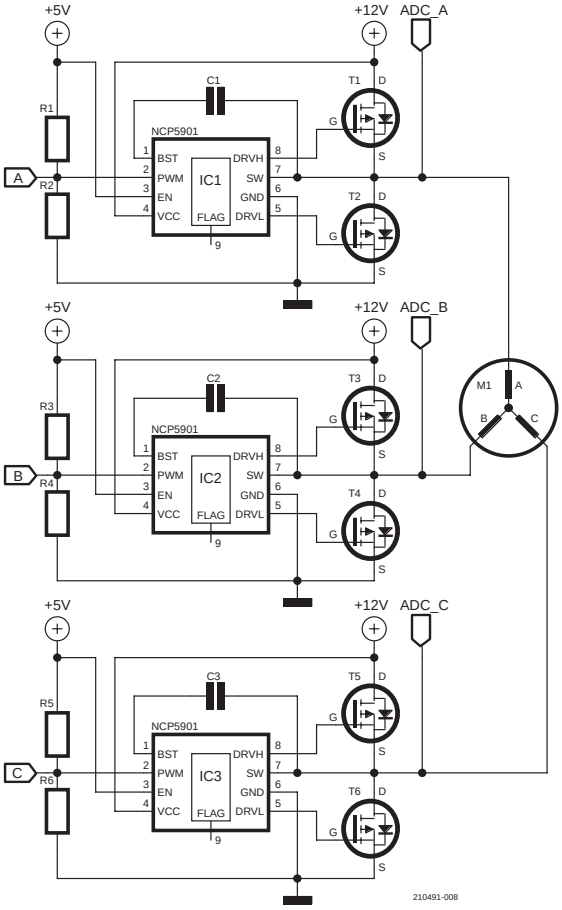


Bild 24. Bürstenloser Motor mit drei halben H-Brücken.

PASSENDE PRODUKTE

- **Geekcreit 5 Stück 5-V-Schrittmotoren mit ULN2003 Treiber-Boards und Dupont Cable (SKU 19783)**
  www.elektor.de/19783
- **Cytron 3 A 4-16 V DC-Motortreiber (2 Kanäle) (SKU 18999)**
  www.elektor.de/18999
- **Cytron 25 A 7-58 V DC-Motortreiber (SKU 19062)**
  www.elektor.de/19062
- **Cytron 10 A 7-30 V DC-Motortreiber-Shield für Arduino (SKU 19063)**
  www.elektor.de/19063
- **PDF: Motorsteuerung mit Arduino und Raspberry Pi (SKU 18726)**
  www.elektor.de/18726
- **Buch: Motors for Makers (SKU 18194)**
  www.elektor.de/18194

WEBLINKS

[1] Basic Kit für den Raspberry Pi-Pico: https://www.elektor.de/basic-kit-for-raspberry-pi-pico-pico-included
[2] 28BYJ-48 Schrittmotor: https://www.elektor.de/geekcreit-5pcs-5v-stepper-motor-with-uln2003-driver-board-dupont-cable
[3] Cytron H-Brücken zur Motoransteuerung: https://www.elektor.de/catalogsearch/result/?q=Cytron%20Driver
[4] Jangeox-Blog: http://www.jangeox.be/2013/10/change-unipolar-28byj-48-to-bipolar.html
[5] Joy-IT NEMA17-03: https://joy-it.net/de/products/NEMA17-03
[6] Microchip AN AVR444: https://www.microchip.com/en-us/application-notes/an8012
[7] ATAVRMC100 Hardware User Guide: https://ww1.microchip.com/downloads/en/DeviceDoc/doc7551.pdf
[8] STM AN1946 : https://www.st.com/resource/en/application_note/cd00020086-sensorless-bldc-motor-control-and-bemf-sampling-methods-with--st7mc-stmicroelectronics.pdf
[9] NXP Semiconductors AN12435: https://www.nxp.com/docs/en/application-note/AN12435.pdf

Das Team im Elektor-LABOR

Unser Ansatz, unsere bevorzugten Werkzeuge und mehr

Von **Jens Nickel** (Chefredakteur)

Es gibt viele Elektronikzeitschriften auf der Welt, aber die meisten berichten nur über neue Entwicklungen und Anwendungen. Elektor ist anders. Mit unserem hauseigenen Labor, einem Team von Ingenieuren, entwickeln, verfeinern und testen wir ständig elektronische Projekte und Produkte.

Seit sechs Jahrzehnten bietet die Zeitschrift Elektor eine gesunde Mischung aus Theorie und Praxis. Das liegt in unserer DNA. Angefangen hat alles vor mehr als 60 Jahren mit dem Elektronik-Enthusiasten Bob van der Horst, der von trockenen Datenblättern und theoretischen Artikeln voller Formeln und Diagramme gelangweilt war. Stattdessen wollte er eine Zeitschrift mit Schaltungen herausgeben, die die Leser selbst bauen und verändern konnten [1]. Von Anfang an wurden deshalb die meisten Schaltungen und Projekte gebaut und getestet, bevor sie in der Zeitschrift Elektor veröffentlicht wurden. Und so wurde das Elektor-Labor schnell berühmt und ist es heute noch. Sind Sie neugierig, wie es im Elektor-Labor des Jahres 2022 aussieht? Dann lesen Sie weiter und machen Sie mit!

Was ist das Elektor-Labor?

Das Elektor-Labor ist sowohl ein Ort als auch ein Team. Jedes Mitglied des Elektor-Labors verfügt zu Hause über seine eigene Elektronik-Werkbank zum Löten, Reverse-Engineering, Programmieren und Tüfteln. Elektor unterhält außerdem einen Elektronik-Arbeitsbereich in dem „Hauptquartier“ in Aachen, der mit praktischen Laborgeräten, Bauteilen und technischen Werkzeugen ausgestattet ist. Die Mitglieder des Laborteams können jederzeit in den Arbeitsbereich kommen, um Entwürfe zu testen oder an Projekten mitzuarbeiten. Das Team besteht aus Ingenieuren verschiedenen Alters, mit unterschiedlichem Hintergrund und unterschiedlichen Persönlichkeiten. Von Anfang an waren im Labor sowohl Analogtechniker, die monatelang an großen Schaltungen herumtüfteln, als auch Programmierbegeisterte, die neue Ideen einbringen wollen, vertreten. Um es einfach auszudrücken, wir haben eine vielfältige Crew. Manche Zeitschriften berichten nur über Bausätze oder neue Produkte. Nicht so Elektor! Wir machen viel mehr. Wir haben uns immer bemüht, alle Details eines Projekts aufzudecken, einschließlich des Quellcodes, egal ob es sich um ein neues Projekt von einem unserer Ingenieure oder um eines aus dem Heimlabor eines begeisterten Lesers handelt. Warum hat Elektor diesen Ansatz gewählt? Unsere Beweggründe sind ist klar: Wir möchten, dass unsere Leser die gleichen Projekte bauen können, und wir laden sie ein, alles, was wir veröffentlichen, zu verändern, zu verbessern und zu erweitern. Ob Sie nun ein professioneller Ingenieur oder ein Wochenend-Elektronikbastler sind, wir möchten, dass Sie verstehen, wie alles funktioniert. Es ist also keine Übertreibung zu sagen, dass Elektor seit langem zu den Vorreitern der Open-Source-Bewegung gehört!

Die meisten Elektor-Leser und Community-Mitglieder werden auch wissen, dass es keine strikte Trennung zwischen unseren Laboringenieuren und dem Redaktionsteam gibt. Alle Ingenieure des Labors (Clemens Valens, Luc Lemmens, Mathias Claussen und Ton Giesberts) schreiben Artikel, und alle Redakteure, darunter auch ich und Content Director C. J. Abate, haben Spaß daran, Ideen und Beiträge zu den Projekten des Labors beizusteuern.

Zu Beginn des Jahres 2022 möchten wir alle anregen, das Team des Elektor-Labors kennenzulernen. Auf diesen Seiten finden Sie eine kurze Präsentation der Mitglieder des Elektor-Labor-Teams und einige der Plattformen und Tools, die wir verwenden. Wir laden Sie ein, Ihre Ideen, Ihr Feedback, Ihre Projekte und Artikel einzubringen!

Das Team im Elektor-Labor

Frankreich

Clemens Valens ist der Ingenieur, der die Online-Plattform Elektor Labs verwaltet. Er hat einen BSc in Elektronik und einen MSc in Elektronik und Informationstechnologie. Clemens Valens begann 2008 als Chefredakteur von Elektor Frankreich für Elektor zu arbeiten und war auch als Redakteur für Elektor UK/US und ElektorMagazine.com tätig. Später war Clemens Leiter der Elektor-Entwicklungslabore in den Niederlanden, Deutschland und Indien. Heute ist er bei Elektor als kreativer Techniker für die Website der Elektor-Labs-Community verantwortlich, auf der Elektronik-Enthusiasten ihre Arbeiten veröffentlichen und sich mit Gleichgesinnten aus der ganzen Welt austauschen können. Neben eigenen Projekten und anderen Artikeln für die Zeitschrift produziert er auch regelmäßig Videos für Elektor-TV und moderiert Webinare. Seine Hauptinteressen sind Sound-Erzeugung und Signalverarbeitung.

Deutschland

Mathias Claußen begann als gelernter IT-Systemelektroniker und erwarb einen Bachelor in Elektro- und Informationstechnik sowie einen Master in Mikroelektroniksystemen. Danach arbeitete er als Projektingenieur für Embedded-Entwicklung und konzentrierte sich auf ARM-Cortex-M-basierte Architekturen und Echtzeitbetriebssysteme. Mathias kam 2018 zu Elektor und ist Teil des Elektor-Labor-Teams, das sich hauptsächlich mit Software beschäftigt. Heute ist Mathias mehr als nur Softwareentwickler, der sich im Hintergrund hält, sondern schreibt auch Artikel über Hardware- und Softwareentwicklung, erstellt Videos für Elektor-TV und vieles mehr. Möchten Sie mehr über seinen Arbeitsplatz zu Hause erfahren? Lesen Sie darüber online bei Elektor [2]. Dort können Sie einige der Geräte und Werkzeuge sehen, die Mathias für viele tägliche Aufgaben verwendet.

Niederlande

Luc Lemmens begann im März 1990 nach seinem Studium an der Technischen Universität Eindhoven für Elektor zu arbeiten. Zu dieser Zeit gab er auch Elex heraus, eine Elektronikzeitschrift, die sich vor allem an Einsteiger richtete. Luc hat Projekte für beide Zeitschriften entworfen und bearbeitet. Er hat viele Interessen, was bedeutet, dass er sich in vielen Bereichen der Elektronik auskennt. Natürlich hat er auch Software in einer Vielzahl von Programmiersprachen geschrieben oder bearbeitet - und, vor allem in seiner Anfangszeit bei Elektor, in Assembler. Heute beschränkt er sich meist auf die Arduino-IDE, die für die meisten einfachen Projekte perfekt ist. Im Laufe der Jahre hat Luc auch technische Texte und sogar Elektor-Bücher geschrieben und übersetzt. In seiner Freizeit repariert und restauriert Luc sowohl moderne elektronische als auch elektromechanische Flipper (mit Relais und Schrittmotoren). Und natürlich spielt er auch damit!

Niederlande

Ton Giesberts begann nach seinem Studium bei Elektuur (jetzt Elektor) zu arbeiten, als man jemanden mit einer Affinität für Audio suchte. Im Laufe der Jahre hat er auch hauptsächlich an Audio-Projekten gearbeitet, denn analoges Design war schon immer seine Vorliebe. Aber natürlich gehören auch Projekte in anderen Bereichen der Elektronik zu seinem Job. Eines der Mottos von Ton ist: „Wenn du es besser haben willst, mach es selbst.“ Zum Beispiel beim Entwurf eines Platinenlayouts, denn bei einem Audioprojekt mit Verzerrungswerten in der Größenordnung von 0,001% ist ein gutes Layout entscheidend.

Aktuelle Elektor-Lab-Projekte und Produkte

Das Team des Elektor-Labors hat viele Elektronikprojekte für Elektor vorbereitet. Manchmal wird ein Design komplett im Elektor-Labor entwickelt. In anderen Fällen adaptieren unsere Labormitarbeiter Projekte von externen Entwicklern. Und manchmal werden Ideen zur Zusammenarbeit mit unseren Partnern ausgelagert. Das Endergebnis kann ein Artikel (oder eine Artikelserie) oder ein Produkt sein, das im Elektor-Store verkauft wird. Im Folgenden finden Sie eine Liste einiger erfolgreicher Elektor-Labor-Projekte der letzten Jahre.

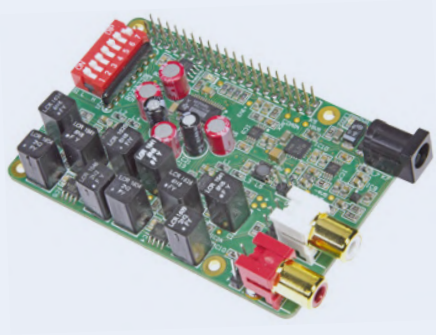
Uhren

Elektronische Uhren sind bei Elektor-Lesern seit jeher sehr beliebt. Auch weil sie die Geräte schlechthin sind, um alten Anzeigen eine neue Anwendung zu geben, wie bei der 6-stelligen Nixie-Uhr [3] und der Flipper-Uhr [4].

Wetterstation mit ESP32

Kaufen Sie ein erschwingliches, komplettes Set von Wetterstationssensoren und fügen Sie modernste Elektronik und Software hinzu, um Ihre eigene Wetterstation zu bauen [5].

Audio-DAC für Raspberry Pi



Raspberry-Pi-Boards haben zwar Audioausgänge, aber wenn Sie einen wirklich hochwertigen Klang wünschen, benötigen Sie einen externen High-End-Digital-Analog-Wandler wie dieses Board [6], das von unserem Audio-Experten Ton Giesberts entwickelt wurde. .

DIY LiPo Supercharger Bundle

Benötigen Sie eine LiPo-Stromversorgung für Ausgangsspannungen von 5 V und 12 V, die man wieder aufladen kann? Dann sehen Sie sich den Bausatz an, den das Elektor-Labor in Zusammenarbeit mit GreatScott entwickelt hat! [7]

Neues LCR-Messgerät 50 Hz bis 2 MHz

Das LCR-Meter 50 Hz bis 2 MHz ist ein hochpräzises Gerät zum Testen und Messen passiver Bauteile. Es kann mit handelsüblichen Geräten nicht nur mithalten, sondern diese sogar übertreffen! [8].

Buch:

Mastering Microcontrollers
Helped by Arduino

Das Buch *Mastering Microcontrollers Helped by Arduino* von Clemens Valens macht Sie vertraut mit der Arduino-Welt. Sie werden die Grundlagen der Mikrocontroller-Programmierung anhand von interessanten Projekten erlernen, zum Beispiel einem Auto-GPS-Scrambler, einer Wetterstation, einer Infrarot-Fernbedienung und einem PID-Regler.

210545-02

Elektor Needs You!

Im Gegensatz zu vielen anderen On- und Offline-Zeitschriften fließen Artikel und andere interessante Inhalte nicht nur in die eine Richtung von Elektor zu Ihnen, sondern es geht auch andersherum. Wir ermutigen unsere wachsende weltweite Community von Ingenieuren und Makern, Ideen, Anleitungen, Tipps und Tricks, Schaltungen und Projekte beizusteuern.
Können Sie anderen etwas über einen bestimmten Teilbereich der Elektronik oder Software mitteilen? Haben Sie ein schönes Elektronikprojekt entwickelt? Haben Sie ein spezielles technisches Fachgebiet? Unsere Leser könnten daran sehr interessiert sein!

Es gibt zwei Möglichkeiten, Tipps, Tutorials und Projekte zu teilen:
1) Sie posten etwas auf Elektor Labs (siehe Kasten *Elektor Labs*). Hier können Sie sofort mit anderen Lesern in Kontakt treten, vielleicht auf der anderen Seite des Globus, um positives und kritisches Feedback zu erhalten! Sie haben auch die Möglichkeit, Unterstützer für Ihr Projekt zu gewinnen (siehe Kasten *Elektor Jumpstarter*). Auf der Plattform Elektor Labs sind Ideen und Fragen stets willkommen!
2) Sie können Ihren Artikel oder ihr Projekt per Mail an die Redaktion schicken: redaktion@elektor.de. Sie müssen keinen vorgefertigten Artikeltext haben, aber ein Schaltplan und eine kurze Beschreibung müssen sein, damit wir wissen, worum es sich bei Ihrer Arbeit dreht. Natürlich werden wir ohne Ihr Einverständnis keine Details Ihres Projekts enthüllen!
Die Elektor-Redakteure und die Mitglieder des Labor-Teams prüfen bei unseren wöchentlichen internen Treffen alle Beiträge. Wenn wir Ihren Artikel zur Online- und/oder Print-Veröffentlichung in auswählen, teilen wir Ihnen das mit und besprechen mit Ihnen die nächsten Schritte.

Die Plattform Elektor-Labs

Die Online-Plattform Elektor Labs wurde für Ingenieure, Maker und Studenten entwickelt, die sich für Elektronik begeistern. Sie ist ein Ort, an dem sie (auch Sie!) ihre Projekte mit anderen teilen, sich an den Projekten anderer beteiligen und über Projektentwicklung und Elektronik diskutieren können. Elektor Labs steht jedem offen; die einzige Voraussetzung ist eine kostenlose Elektor-ID.
Im zehnten Jahr ihres Bestehens hat die Plattform Elektor Labs mehr als 10.000 aktive Nutzer, die mehr als 2.000 Projekte und unzählige Kommentare veröffentlicht haben. Die Online-Plattform ist viel mehr als nur ein kostenloser Raum, in dem man sein Können zeigen kann. Unser Team von Elektronikspezialisten hilft Ihnen nicht nur bei der erfolgreichen Umsetzung Ihres Projekts, sondern auch, wenn Sie daraus ein Produkt verwirklichen wollen, das im Elektor-Store verkauft werden soll. Da jedes Projekt, das in Elektor Labs eingestellt wird, von unserer Redaktion bewertet wird, ist Elektor Labs auch der richtige Ort, um einen Artikel für die Zeitschrift Elektor vorzuschlagen.

www.elektor-labs.com

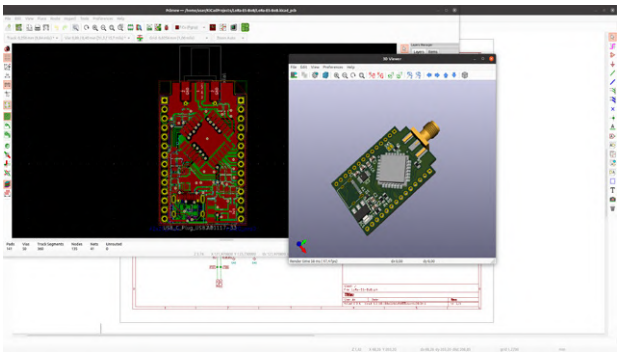
Elektor-Jumpstarter

Elektor Jumpstarter ist die Elektor-Variante des Crowdfunding. Die Online-Plattform soll Innovatoren wie Ihnen helfen, Ihre Elektronikprojekte in kommerzielle Produkte umzusetzen.
Um sich zu bewerben, stellen Sie einfach ein Projekt bei Elektor Labs ein und klicken dann auf den Jumpstarter-Button. Ihr Projekt wird dann von Elektor bewertet. Und wenn ein Projekt akzeptiert wird, kann die Community es unterstützen.
Die Unterstützung eines Projekts ist nicht mit irgendwelchen Zahlungen verbunden. Wenn jemand ein Projekt unterstützt, bekundet er vielmehr sein Interesse, das Endprodukt kaufen zu wollen, wenn die Jumpstarter-Kampagne ihr Ziel, also eine bestimmte Produktionsmenge zu einem vorher festgelegten Preis erreicht. Wenn die Kampagne erfolgreich ist, produziert Elektor das Projekt und stellt es in den Store, wo es jeder kaufen kann. Kurzum, bei Elektor Labs wird unser Motto *Design, Share, Sell* wirklich lebendig.

www.elektormagazine.com/jumpstarter-info

elektor JUMPSTARTER
from project to product

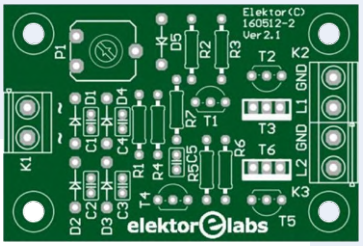
Open-Source-Design-Tool: KiCad



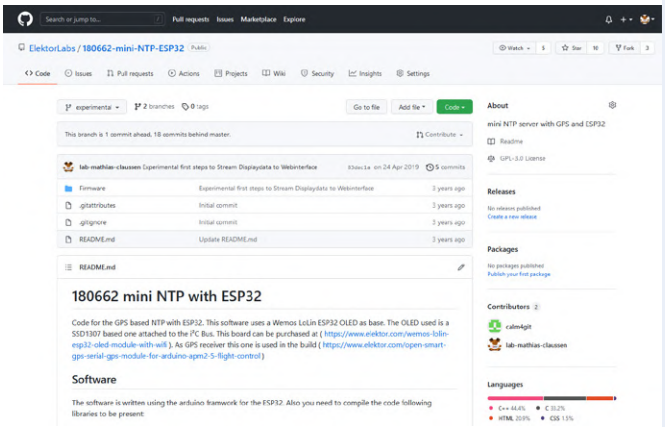
Elektor liebt Open-Source-Lösungen, daher verwenden die Mitarbeiter im Team des Elektor-Labors häufig Open-Source-Tools. Ein Beispiel dafür ist KiCad, das wir (neben Altium und Eagle) für den Entwurf von Schaltplänen und Platinenlayouts verwenden. KiCad ist kostenlos, quelloffen und läuft auf allen drei gängigen Betriebssystemen. Dadurch können Sie Ihre Arbeit mit anderen teilen, so dass auch diese Anderen Ihre Kreation nach Bedarf anpassen oder verändern können. Online-Artikel und Videos über KiCad finden Sie auf der Elektor-Website: **www.elektormagazine.de/tags/kicad**

Elektor-Platinen

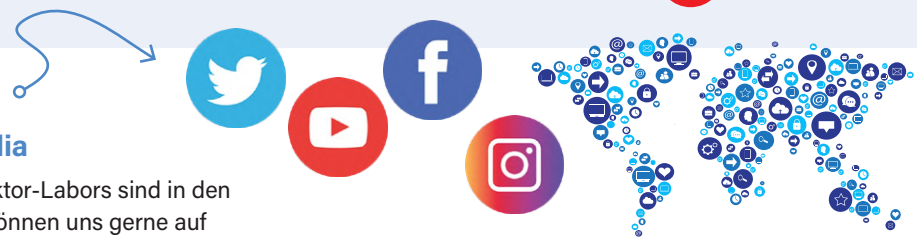
Elektor-Leser sind es gewohnt, in der Zeitschrift diese grünen Abbildungen von (meist kleinen) Platinen vorzufinden. Als Service gab es viele Jahre lang für diejenigen, die die Bauteile selbst zusammensuchen und löten wollten, fast all diese Platinen in unserem Shop. Von den meisten dieser Leerplatinen haben wir aber nur wenige Stück verkauft, und diese kleinen Zahlen sind in den letzten Jahren sogar zurückgegangen. Wir müssen der Tatsache ins Auge sehen, dass sich die Welt der Elektronik verändert. Es hat keinen Sinn, einen Mikrocontroller oder eine kleine Audioverstärkerplatine (um nur zwei Beispiele zu nennen) von Grund auf zu entwickeln und in kleinen Stückzahlen zu (zu) hohen Preisen zu produzieren. Heutzutage sind Maker wie Profis daran gewöhnt (und erwarten es sogar), preiswerte Module zu bekommen, bei denen zumindest die wichtigsten Komponenten – wie SMD-Ladekondensatoren oder Pull-up-Widerstände – bereits bestückt sind. Zeit ist Geld, und deshalb wollen die meisten Elektronikentwickler sofort einsatzbereite Prototyping-Lösungen. Sie erwarten, dass sie den Karton öffnen, alles anschließen und loslegen können.
Das heißt aber nicht, dass wir gar keine Platinen oder Bausätze mehr verkaufen! Wir konzentrieren uns auf die meistversprechenden und interessantesten Projekte, bei denen wir wirklich etwas Originelles erfunden haben, das man nicht überall für ein paar Euro bestellen kann. Für diese Projekte arbeiten wir mit Partnern zusammen, die Erfahrung darin haben, kleine und bei Bedarf auch größere Stückzahlen zu produzieren. Zwei dieser Partner sind Eurocircuits und SIMAC (Joy-iT).
Wenn eine kleine Platine benötigt wird, die nicht über eine Zusammenarbeit mit einem Elektor-Partner erhältlich ist, bieten wir die CAD-Dateien kostenlos an. Jeder Leser kann das Platinenlayout so verwenden, wie es ist, oder es nach eigenem Gusto anpassen und die Platine über einen Leiterplatten-Hersteller bestellen. Wir empfehlen die Dienste von *www.elektorpcb4makers.com* und *www.elektorpcbservice.com*, die wir in Zusammenarbeit mit unserem Partner Eurocircuits anbieten.



Elektor auf GitHub



Das Elektor-Labor nutzt die Plattform Elektor Labs (www.elektormagazine.com/labs), um eigene Projekte und Projekte externer Autoren zu hosten. Sie finden dort die für die Projekte benötigten Dateien (Software und Hardware).
Bei größeren Projekten, die kontinuierlich weiterentwickelt werden, kann die Verwaltung der verschiedenen Versionen unübersichtlich werden. Hier kommt das Versionsverwaltungssystem GitHub ins Spiel, mit dem Sie leicht zu einem bestimmten Entwicklungsstand zurückkehren oder vorgenommene Änderungen nachverfolgen können. GitHub bietet darüber hinaus eine Reihe weiterer Vorteile, wie experimentelle Alternativversionen, Issue-Tracker für Vorschläge und Kommentare oder von der Community bereitgestellte Patches in Form von Pull-Requests.
Daher werden wir in Zukunft immer mehr Software und auch CAD-Daten auf GitHub zum Download anbieten (https://github.com/ElektorLabs). Der entsprechende Weblink findet sich am Ende eines jeden Projektartikels.
Für GitHub-Einsteiger hat Clemens Valens vom Elektor-Labor ein Video-Tutorial auf Elektor TV erstellt: *www.youtube.com/watch?v=X5e3xQBeqf8*.



Folgen Sie uns auf Social Media

Die Mitglieder und Redakteure des Elektor-Labors sind in den sozialen Medien leicht erreichbar. Sie können uns gerne auf Twitter, Facebook und YouTube folgen.

Twitter

Mathias Claussen: **https://twitter.com/elektormathias**
Clemens Valens: **https://twitter.com/clemens_elektor**
Elektor: **https://twitter.com/Elektor**
C. J. Abate: **https://twitter.com/elektor_us**

YouTube

ElektorTV: **https://www.youtube.com/user/ElektorIM**

Facebook und Instagram

Elektor Labs (FB): **https://www.facebook.com/ElektorLabs/**
Elektor Labs (IG): **https://www.instagram.com/elektorlabs**

WEBLINKS

[1] „60 Jahre Elektor", ElektorMag Mai/Juni 2021: https://www.elektormagazine.de/magazine/elektor-174/59550
[2] „An Elektor Engineer's Workspace for Embedded Software Development", Elektormagazine.com: https://www.elektormagazine.com/articles/elektor-workspace-embedded-software-development
[3] 6-Digit Nixie-Uhr: https://www.elektormagazine.de/magazine/elektor-147/58579
[4] Flipper-Uhr: https://www.elektormagazine.de/magazine/elektor-87/42488
[5] Wetterstation mit ESP32: https://www.elektormagazine.de/magazine/elektor-69/42263
[6] Audio-DAC für Raspberry Pi: https://www.elektormagazine.de/magazine/elektor-201707/40438
[7] DIY LiPo Supercharger Bundle: https://www.elektormagazine.de/magazine/elektor-166/59237
[8] Neues LCR-Messgerät 50 Hz bis 2 MHz: https://www.elektormagazine.de/magazine/elektor-158/59082

Anzeige

Mehr lieferbar

Die größte Auswahl an Halbleitern und elektronischen Bauelementen auf Lager und versandfertig™

mouser.de

MOUSER ELECTRONICS

Raspberry Pi als KVM-Fernsteuerung

Die Software Pi-KVM im Elektor-Labor-Test

Von **Mathias Claussen** (Elektor)

KVM steht für Keyboard, Video, Mouse - wer darauf Zugriff besitzt, kann einen Rechner (fern)steuern. Die clevere Software Pi-KVM und ein Raspberry Pi 4 sind eine preiswerte Möglichkeit, nicht nur einen PC, sondern auch andere Geräte über das Internet zu steuern, ohne dass dafür Software auf dem ferngesteuerten Rechner installiert werden muss. Daneben erlaubt Pi-KVM aber auch das Bereitstellen von virtuellen Datenträgern, so dass ein Rechner nicht nur aus der Ferne gesteuert und gewartet werden kann, sondern eine komplette Neuinstallation möglich ist.

Während der Pandemie erreichte mich die Information, dass in meinem Elternhaus noch Rechner laufen, die mit Windows 7 (einige gar mit Windows XP) ausgestattet sind. Wegen der Sicherheitslücken, die auch nicht mehr behoben werden können, müssen diese Rechner dringendst aktualisiert werden. Mit einem Update ist es bei den Maschinen leider nicht getan, sie verlangen nach einer kompletten Neuinstallation. Hinfahren ist keine Option, wegen der Pandemie und mehr als acht Stunden Fahrzeit.
Wer schon einmal versucht hat, jemanden mit einer Mischung aus Telefonat und Videochat durch Dutzende von Installationsschritten zu leiten, kann sich leicht vorstellen, dass so etwas bei einer Betriebssysteminstallation und Datensicherung keine gute Idee ist. Ein Remotezugriff muss her. Und genau das ist es, was mit einem Raspberry Pi als KVM-Fernsteuerung ermöglicht wird.
An dieser Stelle kann in dem Text auch Eltern gegen Kunden, Rechner gegen Maschinensteuerung und Neuinstallation durch Fehleranalyse ersetzt werden. Ein Paar Augen, mit dem man sich aus der Ferne hardwareunabhängig ein System ansehen kann, ermöglicht es, lange Anfahrten zu vermeiden. Man kann so auch Geräte und Messinstrumente fernbedienen, die dafür nie vom Hersteller vorgesehen waren. **Bild 1** soll den Einsatz von Pi-KVM für die Fernsteuerung eines PCs verdeutlichen. Eine Maschine könnte wie in **Bild 2** ferngewartet werden.

Es gibt doch Fernwartungssoftware?

Wer auf einem Raspberry Pi oder einem PC unterwegs ist, wird sicherlich schon einmal ein Tool wie VNC, AnyDesk oder TeamViewer benutzt haben. Neben diesen gibt es noch etliche weitere Lösungen für die Fernwartung und Steuerung. Alle diese Lösungen sind preiswert oder für den Privatanwender sogar kostenfrei zu nutzen. Um jemanden neue Software einzurichten oder beim Erstellen von Social Media Posts zu helfen, ist so etwas eine schnelle und unkomplizierte Lösung. Diese Helfer setzen aber alle ein laufendes Betriebssystem mit einer Internetanbindung voraus. Doch was, wenn der Rechner nicht mehr bootet oder keine Netzwerkverbindung mehr aufbauen kann? Oder wenn das Betriebssystem neu installiert werden muss? Das

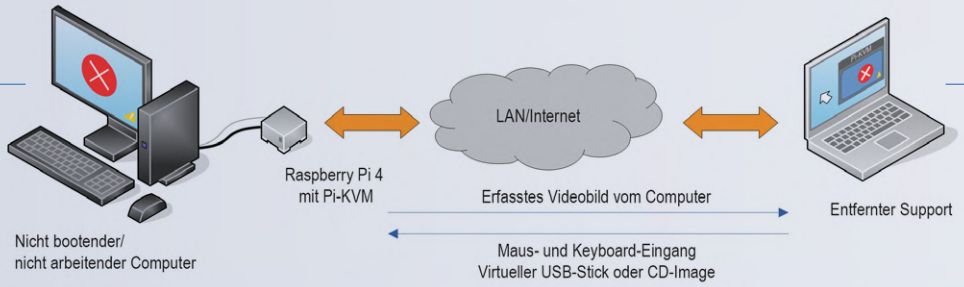


Bild 1. Remotesupport mit Pi-KVM am entfernten PC.

ist der Punkt, an dem jemand mit passenden Fachkenntnissen vor Ort sein muss, um das Betriebssystem wieder zum Booten zu bewegen und ein Netzwerk oder eine Internetverbindung herzustellen. Dies gilt auch für die Neuinstallation eines Betriebssystems. **Bild 3** zeigt die BIOS-Meldung eines PCs, der keine bootfähigen Datenträger finden konnte. Pi-KVM arbeitet unabhängig von dem zu steuernden Rechner und überträgt das Videobild, das die Grafikarte ausgibt, über das Netzwerk an ein zweites System, auf dem nur ein HTML5-fähiger Webbrowser laufen muss. Dort kann dann nicht nur die Bildausgabe des ferngesteuerten Rechners dargestellt werden, die Mausbewegung und Tastatureingaben werden auch wieder an den zu steuernden Rechner zurückgegeben. Damit kann man aus der Ferne den Rechner bedienen, als ob man vor dessen Bildschirm, Maus und Tastatur sitzen würde. **Bild 4** zeigt die BIOS-Meldung des PCs nun bequem im Browser des entfernten Rechners. Auch der Zugriff auf das BIOS (wie in **Bild 5**) lässt sich bequem per entferntem Browser steuern, so das es nicht zwingend erforderlich ist, in der Nähe des PCs zu sein.

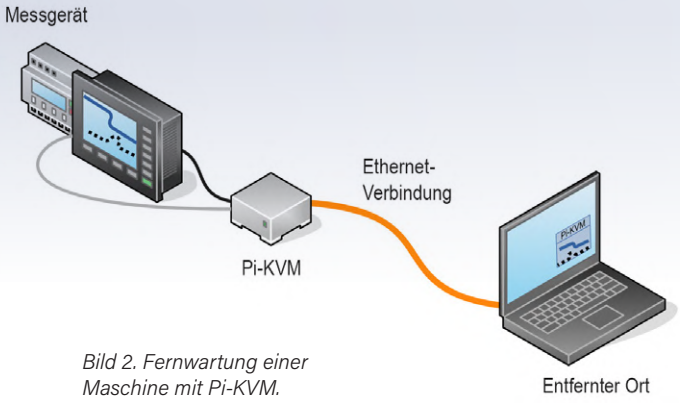


Bild 2. Fernwartung einer Maschine mit Pi-KVM.

Virtuelle Datenträger

Etwas, das über größere Distanz (meist reichen schon 2 m) kompliziert wird, ist das Einstecken von USB-Bootsticks oder rotierenden Datenträgern wie DVD oder CD. Man kann nun einen Bootstick zu dem fernzusteuernden PC senden und hoffen, dass jemand vor Ort diesen auch richtig einsteckt. Pi-KVM bietet die Option virtueller Datenträger, kann also einen USB-Stick oder ein USB-CD-ROM-Laufwerk emulieren. Dazu lässt sich einfach per Browser das passende Datenträgerabbild hochladen und einbinden. **Bild 6** zeigt das Menü der virtuellen Datenträger. Aktuell ist man bei der CD-Emulation auf ISO-Dateien beschränkt, so dass diese nicht größer als 2 GB sein dürfen. Für ISO-Dateien größer 2 GB (**Bild 7**) müssen Hybrid-ISOs vorliegen, also solche, die auch auf einen USB-Stick geschrieben werden könnten. Diese lassen sich dann mit der

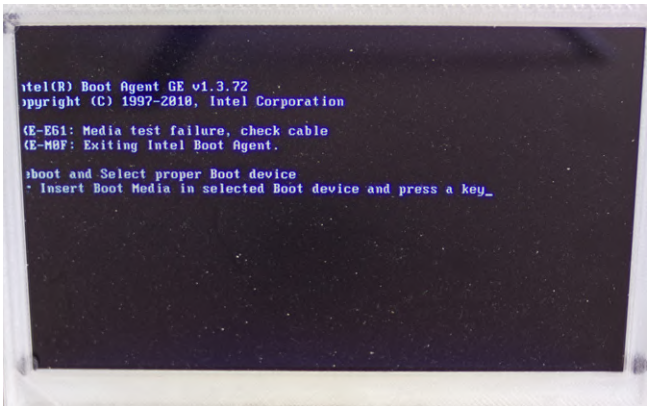


Bild 3. Fehlermeldung beim Booten.

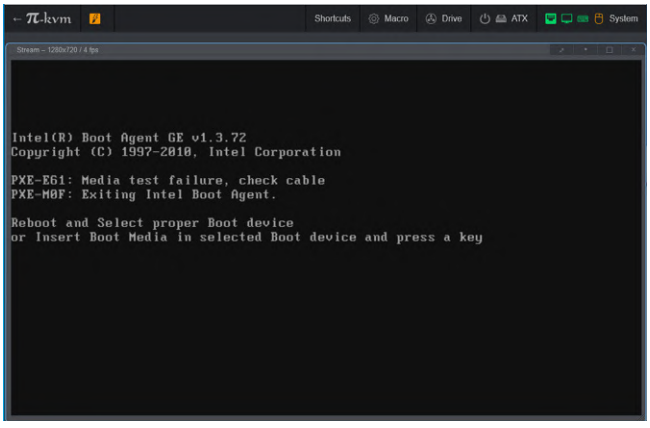


Bild 4. Fehlermeldung im Browser mit Pi-KVM.

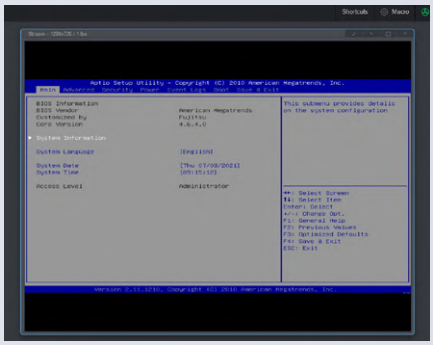


Bild 5. BIOS-Einstellungen aus der Ferne.

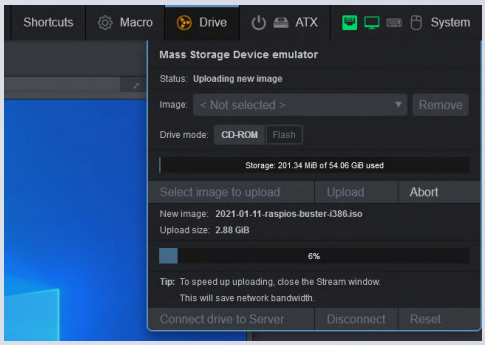


Bild 6. Virtuelle Datenträger.

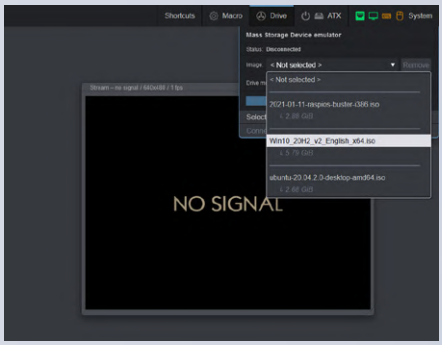


Bild 7. ISOs größer 2 GB als Massenspeicher.

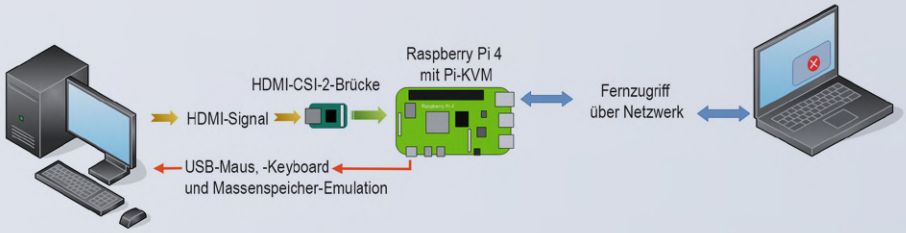


Bild 8. Datenfluss mit HDMI-CSI-Bridge.

Bild 10. Raspberry Pi 4 2-GB-Version.

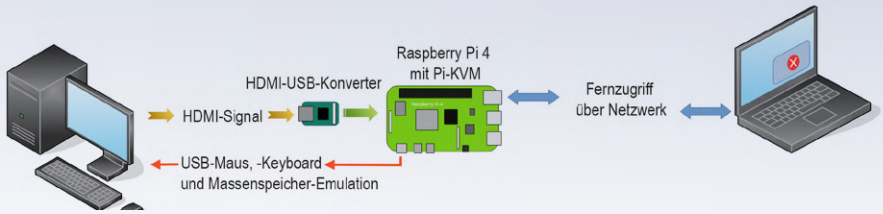


Bild 9. Datenfluss mit HDMI-USB-Dnongle.

Emulationsart *Flash* verwenden und erscheinen als USB-Massenspeicher.

Gibt es nicht sowas wie Pi-KVM schon?

Die Funktonen, die ein Pi-KVM bereitstellt, werden auch von anderen Geräten anderer Hersteller angeboten, jedoch meistens deutlich teurer als die Pi-KVM-Lösung und außerdem mit Closed-Source-Firmware. Bei diesen Geräten sind dann auch etliche Komfortfunktionen aufpreispflichtig. Der Pi-KVM kommt als Open-Source-Projekt und kann mit gängiger Hardware zusammengesetzt werden. Am Ende sind etwa 100 Euro für die DIY-Lösung zu veranschlagen. Es gibt zwei Varianten, einen Pi-KVM zu bauen, eine mit einer HDMI-CSI-Bridge und eine mit einem USB-HDMI-Dongle. **Bild 8** und **Bild 9** zeigen den Datenfluss durch den Raspberry Pi 4 bei der jeweiligen Lösung. An dieser Stelle wollen wir einmal die benötigten Komponenten vorstellen und ein paar Hinweise zum Nachbau geben.

Benötigte Hardware

Die Hardware, die für den Pi KVM benötigt wird, ist recht simpel:

- Ein Raspberry Pi Model 4 (ab 2 GB RAM) (**Bild 10**)
- Eine Micro-SD-Karte (16 GB empfohlen)
- Ein USB-Netzteil (5V/3A)
- Optionales Gehäuse
- HDMI-zu-CSI-Bridge oder USB-HDMI-Capture-Dongle (**Bild 11**)
- Ein USB-Y-Kabel (**Bild 12** und **Bild 13**)

Während die ersten drei Komponenten bei den meisten Lesern vorhanden oder leicht zu besorgen sein dürften, sind die letzten drei etwas schwieriger zu beschaffen. Für die HDMI-zu-CSI-Bridge (**Bild 14** zeigt zwei erhältliche Varianten) hingegen muss für eine europäische Bezugsquelle etwas gesucht oder auf eine aus Fernost zurückgegriffen werden. Ein USB-HDMI-Capture-Dongle kann bei einschlägigen Warenversendern für wenige Euro bestellt werden. Auch wenn der Preis dieser Dongles verlockend sein mag, so haben diese einige Einschränkungen, was die Stabilität oder die unterstützten Auflösungen angeht. Die HDMI-zu-CSI-Bridge ist die stabilere und kompatiblere Wahl.

Für die Verbindung zwischen dem Raspberry Pi 4 und dem zu steuernden Rechner wird der USB-C-Anschluss des Raspberry Pi 4 im USB-OTG-Modus verwendet. Damit kann sich der Raspberry Pi 4 gegenüber einem PC wie eine Tastatur, Maus oder Massenspeichergerät verhalten. Hierfür wird der Raspberry Pi 4 mit einem USB-A-zu-USB-C-Kabel mit dem Rechner verbunden und würde darüber auch mit Strom versorgt. Da hier 3 A bei 5 V nötig sind, kann ein stabiler Betrieb nicht garantiert werden, so dass zusätzlich Leistung durch ein Netzteil bereitgestellt werden muss.

Da der USB-C-Anschluss im OTG-Modus verwendet wird, könnte man über die 40-Pin-Stiftleiste 5 V in den Raspberry Pi 4 speisen. Jedoch würden diese 5 V dann auch

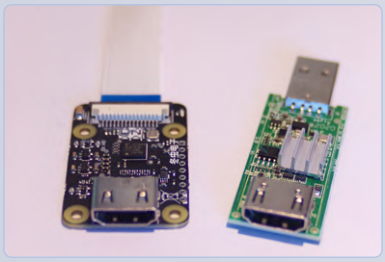


Bild 11. HDMI-zu-CSI-Bridge und USB-HDMI-Capture-Dongle.

Bild 12. USB-A-Y-Kabel.

Bild 13. USB-A-zu-USB-C Kabel.

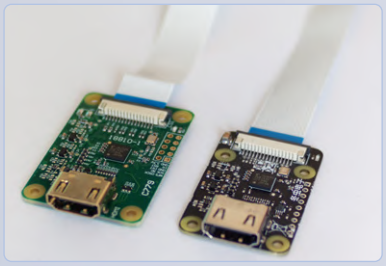


Bild 14. Zwei Varianten der HDMI-zu-CSI-Bridge.

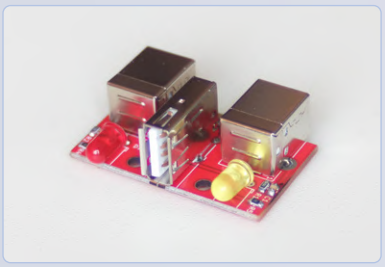


Bild 15. Aufgebaute Adapterplatine.

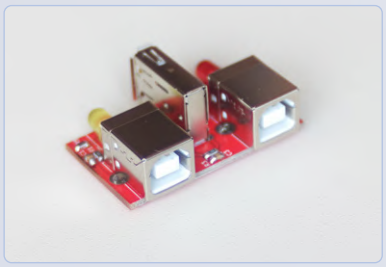


Bild 16. Zwei robuste USB-B-Buchsen.

Bild 17. Micro-SD Karte mit 16 GB Speicher.

Bild 18. Fertiger Aufbau mit HDMI-zu-CSI-Bridge.

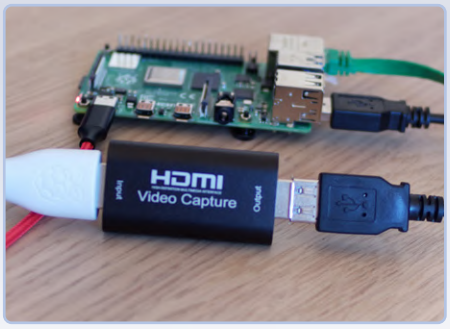


Bild 19. Fertiger Aufbau mit HDMI-USB-Dongle.

den an den USB-C-Port angeschlossenen Rechner mit Strom versorgen. Das ist etwas, das etliche Rechner nicht gerne mögen und mit Fehlfunktionen quittieren. Eine Lösung besteht in einem Y-Kabel, das die Daten und die Spannungsversorgung aufteilt.

Damit nicht zwei USB-Kabel passend zusammengelötet werden müssen, hat das Elektor-Labor eine kleine Platine geroutet (**Bild 15**). Zwei USB-B-Buchsen (**Bild 16**) erlauben einmal den Anschluss an ein Netzteil und einmal den Anschluss an einen Rechner, ohne dass 5 V rückwärts gespeist werden. Beiden Buchsen gemein ist, dass eine LED das Vorhandensein der 5-V-Busspannung signalisiert. Die Platine liegt als KiCAD-Projekt vor und kann aus dem GitHub-Repository von Elektor heruntergeladen werden [1]. Die Platine ist jedoch noch nicht ausgiebig getestet worden, daher erfolgt der Nachbau auf eigene Gefahr (siehe Schaltplan und Stückliste am Ende des Artikels).

Software-Installation

Die Software Pi-KVM kann von der Homepage [2] des Entwicklers heruntergeladen werden. Eine SD-Karte mit 16 GB-Speicher (**Bild 17**) wird für die Installation empfohlen. Wird eine HDMI-zu-CSI-Bridge verwendet, so kann man das *Raspberry Pi 4 v2 Platform Image* für die HDMI-zu-CSI-Bridge nutzen; wird ein USB-HDMI-Dongle eingesetzt, entsprechend das HDMI-USB-Dongle-Image.

Die heruntergeladene Datei muss mit einem Programm wie 7-Zip entpackt und dann mit einem Tool wie dem *Raspberry Pi Imager* auf eine SD-Karte geschrieben werden. Damit ist die Basisinstallation abgeschlossen und es kann mit dem Zusammenbau der Hardware begonnen werden. Wer ein Blick in den Quelltext wagen möchte, kann dieses im GitHub-Repository [3] des Pi-KVM machen.

Hardwarezusammenbau

Wird die CSI-zu-HDMI-Bridge verwendet, gestaltet es sich mit einem passenden Gehäuse leider etwas schwieriger. Entweder hat jemand schon ein solches Gehäuse gebaut oder aber man muss sich mit ein wenig FreeCAD und einem 3D-Drucker selbst behelfen. Für diesen Artikel werden die Teile daher fliegend verbaut.

Für den Aufbau mit HDMI-zu-CSI-Bridge wird eine entsprechende Platine und ein Raspberry-Pi-Kamerakabel benötigt. Eine passend vorbereitete SD-Karte mit dem aktuellen Image für Pi-KVM muss in den Raspberry Pi 4 eingelegt und die HDMI-zu-CSI-Bridge mit dem Kameraport verbunden werden. Das USB-Y-Kabel muss mit dem USB-C-Port des Raspberry-Pi-4 verbunden sein. Damit ist der Pi-KVM einsatzbereit (**Bild 18**).

Für die Variante mit HDMI-USB-Dongle muss dieser mit einem der USB-2.0-Ports des Raspberry Pi 4B verbunden sein. Verwendet man einen der blauen USB-3.0-Ports, so wird der abgebildete USB-Dongle nicht erkannt (**Bild 19**).

Erster Testlauf

Sind alle Teile zusammengebaut und wenn möglich in einem Gehäuse verstaut, steht dem ersten Testlauf nichts mehr im Weg.

Ich selbst habe als Testobjekt einen zweiten Raspberry Pi 4 verwendet, der seit einiger Zeit als Kamera seinen Dienst auf meinem Schreibtisch verrichtet (**Bild 20**). Der Artikel zur Raspberry Pi 4 Kamera ist in ElektorMag 9-10/2021 zu finden [4]). Der Raspberry Pi 4, auf dem Pi-KVM läuft, wird an ein kabelgebundenes Netzwerk angeschlossen.

Nach dem ersten Start des Pi-KVM dauert es einen kleinen Moment, da die Partition an die Größe der SD-Karte angepasst wird. Ist dieser Vorgang abgeschlossen, versucht Pi-KVM per DHCP eine IP-Adresse zu erhalten. Eventuell ist es nötig, im Router oder DHCP-Server nachzusehen, welche IP dem Pi-KVM zugewiesen wurde. Mit einem Webbrowser kann dann die Oberfläche des Pi-KVM aufgerufen werden. Nach dem Booten kann man sich auf der Oberfläche mit dem Benutzernamen *admin* und dem Kennwort *admin* anmelden. Unter dem Punkt *KVM* erscheint dann das Schirmbild des zweiten Raspberry Pi 4 und sowohl Maus als auch Tastatur lassen sich nun aus der Ferne bedienen.

Bild 21 zeigt die Weboberfläche mit dem Desktop des ferngesteuerten Raspberry Pi, aufgenommen mit der HDMI-zu-CSI-Bridge. **Bild 22** zeigt die gleiche Szene, wobei hier ein HDMI-USB-Dongle zum Einsatz kam. Die Bildqualität ist schlechter und es sind grüne Ränder als Artefakte sichtbar.

Für die weitere Konfiguration und die Optionen lohnt sich ein Blick auf die Homepage von Pi-KVM oder in das GitHub-Repository. Neben Maus und Tastatur steht wie schon erwähnt auch ein Massenspeicher bereit. Der Raspberry Pi 4 mit Pi-KVM kann entweder ein optisches Laufwerk oder einen USB-Stick emulieren. Dazu lässt sich das passende Image über die Weboberfläche auf den Raspberry Pi 4 kopieren (**Bild 23**).

Ausblick

Die Pi-KVM Software und ein paar preiswerte Komponenten verwandeln einen Raspberry Pi 4 in eine Fernsteuerung nicht nur für Rechner, sondern auch anderes Equipment, das sich mit Maus und Tastatur bedienen lässt. Wer nicht selbst die Komponenten zusammenbauen möchte, sollte einen Blick auf die PI-KVM Homepage wagen. Dort wird an einem fertigen HAT für den Raspberry Pi 4 gearbeitet und einem dazu passenden Gehäuse. Pi-KVM und seine Entwickler werden momentan mit Hilfe von Spenden durch Patreon oder Paypal finanziert. Sollte Ihnen das Projekt gefallen, würde der Entwickler sich sicherlich über eine kleine Anerkennung freuen.

200523-02

Ein Beitrag von

Entwurf und Text: **Mathias Claußen**
Redaktion: **Jens Nickel**
Layout: **Harmen Heida**

Haben Sie Fragen oder Kommentare?

Haben Sie technische Fragen oder Kommentare zu diesem Artikel? Schicken Sie eine E-Mail an den Autor unter mathias.claussen@elektor.com oder kontaktieren Sie Elektor unter editor@elektor.com.

Bild 20. Raspberry Pi Kamera aus ElektorMag 9-10/2021.

PASSENDE PRODUKTE

- **Raspberry Pi 4 B (4 GB RAM) (SKU 18964**
  www.elektor.de/18964
- **Raspberry Pi HDMI/CSI-2 Adapter (bis 1080p25fps) (SKU 19707)**
  www.elektor.de/19707

WEBLINKS

[1] KiCad-Dateien auf GitHub: https://github.com/ElektorLabs/200523-Raspberry_Pi_4_with_PiKVM
[2] Pi-KVM Homepage: https://www.pikvm.org/
[3] Pi-KVM GitHub Repository: https://github.com/pikvm
[4] Mathias Claußen, „Fotografieren und Video-Streaming mit dem Raspberry Pi 4", ElektorMag 9-10/2021: https://www.elektormagazine.de/200582-02

STÜCKLISTE

Widerstände (0,25 W)

R1,R2,R3 = 100 k

R4,R5 = 1 k

Kondensatoren

C1,C2,C3 = 100 n

Halbleiter

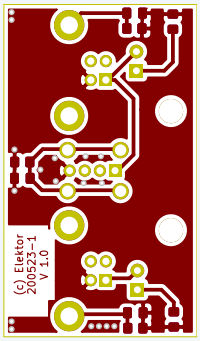
D1,D2 = LED rot 5 mm

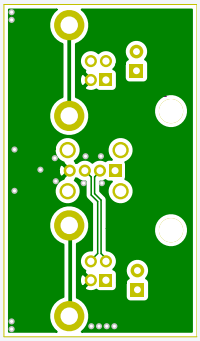
Sonstiges

K1,K2 = USB-B-Buchse für Platinenmontage

K3 = USB-A-Buchse für Platinenmontage

Platine = 200523-1 V1.0





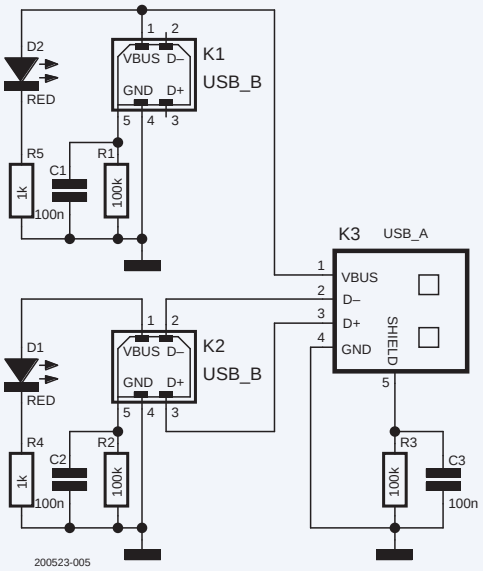


Bild 21. Desktop der Raspberry Pi Kamera mit HDMI-zu-CSI-Bridge aufgenommen.

Bild 22. Desktop der Raspberry Pi Kamera mit HDMI-USB-Dongle aufgenommen.

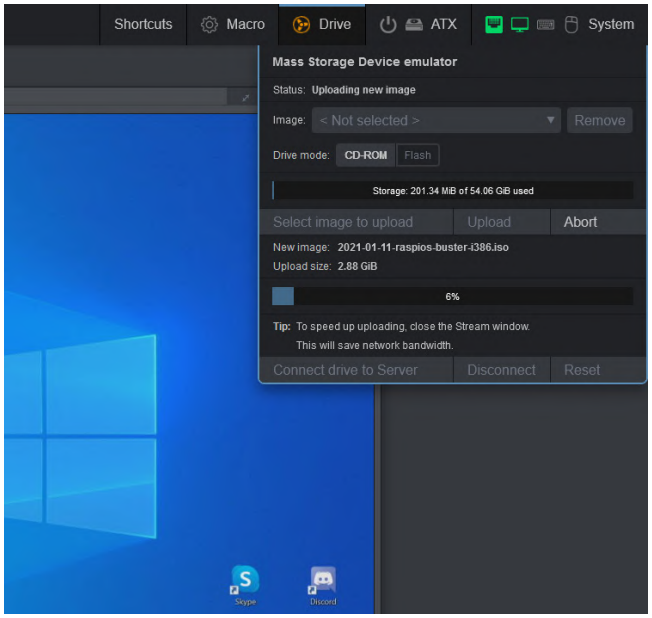


Bild 23. Massenspeicher-Emulation auf der Weboberfläche.

IQaudio **Codec Zero**

Eine Soundkarte für die Raspberry Pi Familie

Von **Mathias Claußen** (Elektor-Labor)

Wer auf der Suche nach einer kleinen und kompakten Soundkarte für einen Raspberry Pi ist, sollte sich den IQaudio Codec Zero einmal näher ansehen, der einen Zero-Formfaktor aufweist.

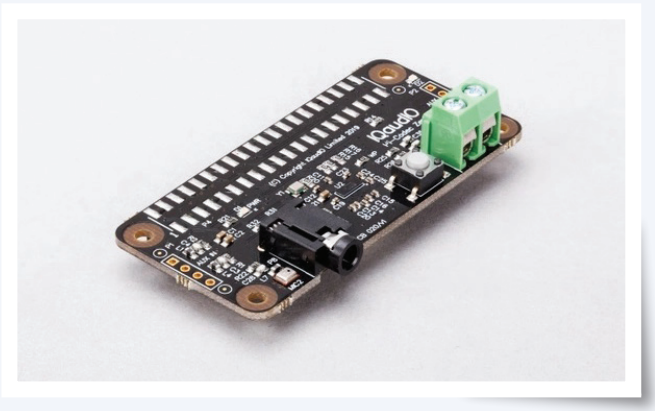


Bild 1. IQaudio Codec Zero (Quelle: Raspberry Pi Foundation).

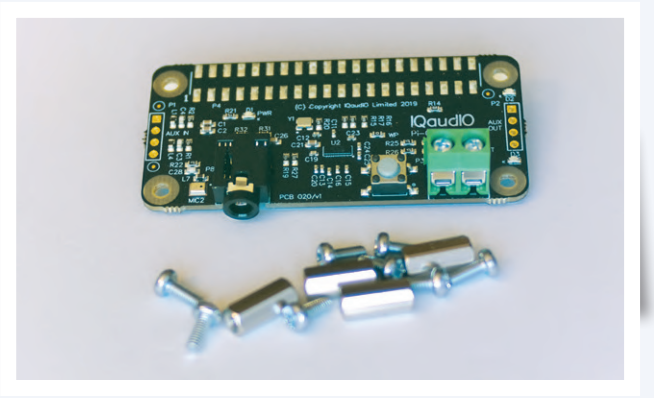


Bild 2. Packungsinhalt des IQaudio Codec Zero.

Wer nur eine digitale Tonausgabe per HDMI benötigt, braucht keine Soundkarte. Eine Soundkarte wird dann interessant, wenn man analogen Ton in guter Qualität ausgeben möchte oder aber auch analoge Signale mit dem Raspberry Pi verarbeiten will. Die A- und B-Modelle des Raspberry Pi besitzen noch einen 3,5-mm-Klinkenanschluss, der zwar analoges Audio ausgeben kann, dessen Qualität jedoch nicht die beste und zum Musikhören nicht wirklich geeignet ist.

Der IQaudio Codec Zero (**Bild 1**) bietet einen integrierten 1,2-W-Verstärker für einen 8-Ω-Lautsprecher, ein integriertes MEMS-Mikrofon sowie einen analogen AUX-Ein- und Ausgang mit Line-Level. Damit sind Projekte wie ein Smart-Speaker, ein VoIP-Telefon oder eine eigene Musikbox recht einfach realisierbar. **Bild 2** zeigt die Hardware und das mitgelieferte Befestigungsmaterial. Herz des IQaudio Codec Zero ist der DA7212 von Dialog Semiconductor. **Bild 3** zeigt die Blöcke und das Signalrouting innerhalb des DA7212. Das IC ist ein 24-Bit-Codec mit bis zu 96 kHz Samplerate, integriert ist ein 5-Band-Equalizer sowie ein ALC und ein Noise-Gate. Die Dokumentation zum IQaudio Codec Zero kann auf der Seite [1] der Raspberry Pi Foundation gefunden werden.

Fast Plug & Play

Der IQaudio Codec Zero besitzt ein EEPROM nach HAT-Spezifikation des Raspberry Pi, so dass dieser automatisch das Board erkennen und passend einrichten sollte. Bei einem Raspberry Pi 3B+ funktioniert dies auch prima, das aktuelle Raspberry Pi OS (32 Bit) erkennt das Board und richtet dieses auch passend ein. **Bild 4** zeigt den IQaudio Codec in den Soundeinstellungen.
Es sind jedoch noch ein paar Schritte nötig, die nicht direkt in der Anleitung zum IQaudio Codec Zero beschrieben sind. Zuerst sollte die Ausgabelautstärke auf ein Minimum reduziert werden. Nachdem der interne Audiocodec des Raspberry Pi laut Anleitung deaktiviert wurde, bleibt nun der IQaudio als Default-Soundkarte im

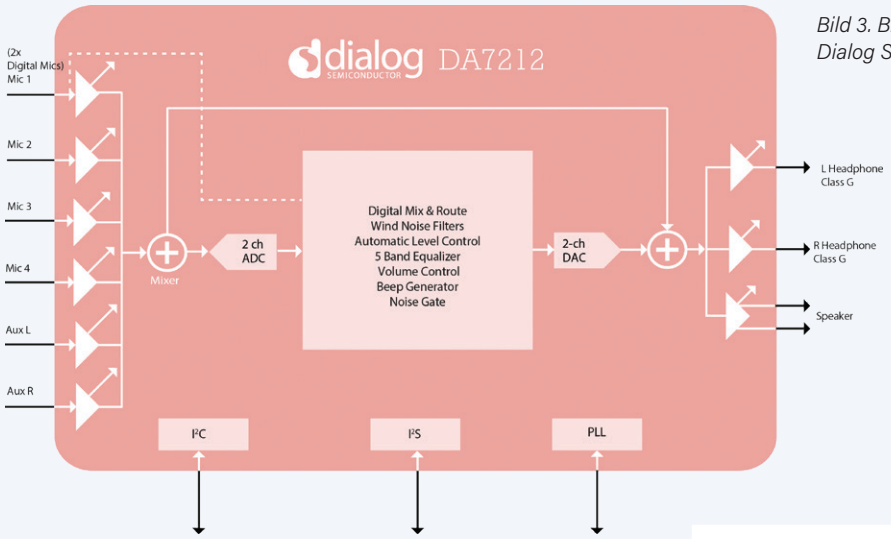


Bild 3. Blockdiagramm des DA7212 (Quelle: Dialog Semiconductor).

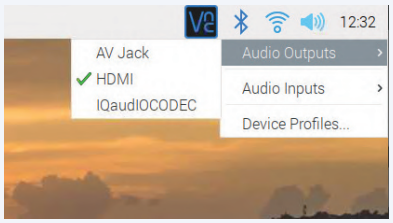


Bild 4. IQaudio Codec Zero Einstellungen.

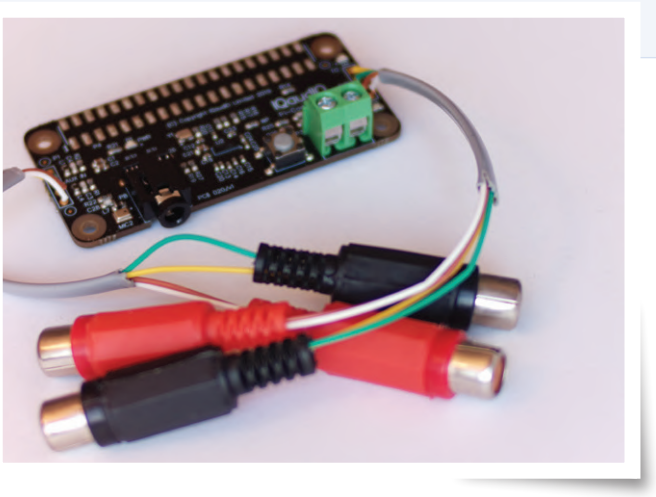


Bild 5. AUX In und Aux Out angeschlossen.

System. Wenn nun ein Terminal geöffnet wird, kann mit `speaker-test -t wav -c 1` ein Lautsprechertest gestartet werden. Der Lautsprecher bleibt jedoch stumm. Es ist nötig, in den Einstellungen der *Advanced Linux Sound Architecture* (*ALSA*) ein paar Parameter anzupassen. Am schnellsten ist das Wiedereinspielen der bereitgestellten Einstellungen von IQaudio. Dazu wird per Terminal ein Git Repository geklont mit `git clone https://github.com/iqaudio/Pi-Codec.git`. Im Terminal wird nun in den erstellten Ordner *Pi-Codec* gewechselt und der Befehlt `sudo alsactl restore -f IQaudIO_Codec_Playback_Only.state` ausgeführt. Danach sollte `speaker-test -t wav -c 1` ein *Front Left* aus dem Lautsprecher ertönen lassen. Ab jetzt werden alle Sounds über den IQaudio Codec Zero ausgegeben. Eine schnelle und einfache Installation!

Mikrofon, AUX-In und AUX-Out

Mit dem eingebauten Mikrofon kann der Raspberry Pi Audio aufzeichnen. So lässt sich ein Smart-Speaker realisieren, zum Beispiel mit voice2json [2]. Oder man kann den Raspberry Pi in Verbindung mit Edge Impulse [3] nutzen, um Daten aufzuzeichnen. Das MEMS-Mikrofon auf der Platine wird keinen Preis in der Kategorie „Aufnahmequalität" gewinnen, erledigt aber seinen Job. Wer mit dem Klang unzufrieden ist, kann ein externes Mikrofon anschließen. Die vordefinierten *.state*-Dateien im geklonten Pi-Codec-Ordner helfen, die passenden Einstellungen für die Aufnahme per Mikrofon zu erhalten. Mit `sudo alsactl restore -f IQaudIO_Codec_OnboardMIC_record_and_SPK_playback.state` wird das Mikrofon als Aufnahmequelle ausgewählt. Für eine Testaufnahme kann im Terminal `arecord --device=hw:1,0 --format S16_LE --rate 44100 -c2 test.wav` verwendet werden, um mit dem Mikrofon eine Testaufzeichnung zu starten und mit *STRG+C* zu beenden. Diese lässt sich dann mit `aplay test.wav` wiedergeben.

Wer eine externe Audioquelle anschließen möchte, kann an den IQaudio Codec Zero passende Buchen anlöten; **Bild 5** zeigt AUX In mit Cinch-Buchsen. Um AUX In als Aufnahmequelle auszuwählen, muss man in einem Terminal `sudo alsactl restore -f IQaudIO_Codec_AUXIN_record_and_HP_playback.state` eingeben. AUX Out ist ebenfalls auf dem Bild zu sehen, mit dem der IQaudio Codec Zero als Zuspieler für einen Verstärker oder aktive Boxen dienen kann.

Neues Leben für ein altes Radio

Etwas, das schon zu lange auf meiner To-Do-Liste steht, ist ein altes Philips-Radio (**Bild 6** und **Bild 7**). Vom Innenleben ist nur noch der Lautsprecher geblieben und es steht Platz für neue Hardware bereit. Im Inneren würde sich ein energiesparender Raspberry Pi gut machen, jedoch wird für den Lautsprecher ein Verstärker benötigt. Gut, dass im IQaudio Codec Zero ein 1,2-W-Verstärker integriert ist, damit lässt sich ein kleines Büro wunderbar beschallen.

Bild 6. Altes Gehäuse eines Radios.

Bild 7. Es ist nur noch ein Lautsprecher übrig.

Für die Musikwiedergabe kann alles verwendet werden, was Audio in Richtung ALSA unter Linux ausgibt. Ob aber der MPD oder eine spezielle Distribution wie Volumio zum Einsatz kommt, steht noch nicht fest. Ein komplett fertiges Konzept für das Radio habe ich also noch nicht, aber eine der Hürden, ein Codec mit kleinem Verstärker wäre schon einmal aus dem Weg. So wird sich mit der Zeit das Innenleben Stück für Stück mit Hardware füllen. Es gibt bei diesem Projekt noch viel zu erledigen und zu planen.

Zusammenfassung

Der IQaudio Codec Zero bietet Aux IN, Aux Out sowie ein integriertes Mikrofon. Die Hardware kann einen kleinen Lautsprecher mit 1,2 W treiben. Für den Preis von etwa 20 Euro ist dies eine kompakte Lösung, die sowohl auf einem Raspberry Pi Zero als auch auf einem Raspberry Pi 3B oder Raspberry Pi 4B eingesetzt werden kann. Für Streaming-Player, VoIP-Telefone oder Smart-Speaker stellt der IQaudio Codec Zero eine solide Basis dar. Wer mit Audio und Streaming auf dem Raspberry Pi und Linux experimentieren möchte, sollte den IQaudio Codec Zero in Betracht ziehen.

210535-02

Sie haben Fragen oder Kommentare?

Haben Sie technische Fragen oder Kommentare zu diesem Artikel? Schicken Sie eine E-Mail an den Autor unter mathias.claussen@elektor.com oder kontaktieren Sie Elektor unter editor@elektor.com.

Ein Beitrag von

Text und Bilder: **Mathias Claußen**
Redaktion: **Jens Nickel**
Layout: **Giel Dols**

WEBLINKS

[1] IQaudio Codec Zero: www.raspberrypi.org/products/iqaudio-codec-zero/
[2] Voice2Json: http://voice2json.org
[3] M. Claussen: Nvidia Jetson Nano - Bildverarbeitung für Einsteiger (Teil 2), Erkennen von Objekten mit Edge Impulse, Elektor 11-12/2021 : www.elektormagazine.de/210318-B-02

PASSENDE PRODUKTE

- **IQaudIO Codec Zero – Sound Card for Raspberry Pi Zero (SKU 19541)**
  www.elektor.de/19541
- **Raspberry Pi Zero WH (mit bestücktem Header) (SKU 18567)**
  www.elektor.de/18567
- **Raspberry Pi Zero 2 W (SKU 19906)**
  www.elektor.de/19906
- **Raspberry Pi 3 B+ (SKU 18452)**
  www.elektor.de/18452
- **Raspberry Pi 400 – Raspberry Pi 4-basierter PC (DE) + GRATIS GPIO-Header (SKU 19430)**
  www.elektor.de/19430

Das PiKVM-Projekt
und seine Lehren

Ein Interview mit Maxim Devaev* (Entwickler von PiKVM)

Von **Mathias Claußen** (Elektor-Labor)

Die PiKVM-Software für den Raspberry Pi 4 ermöglicht den Fernzugriff auf einen PC, Server oder eine andere Maschine über eine Netzwerkverbindung. Wir konnten dem Entwickler von PiKVM ein paar Fragen zu seinem Projekt stellen.

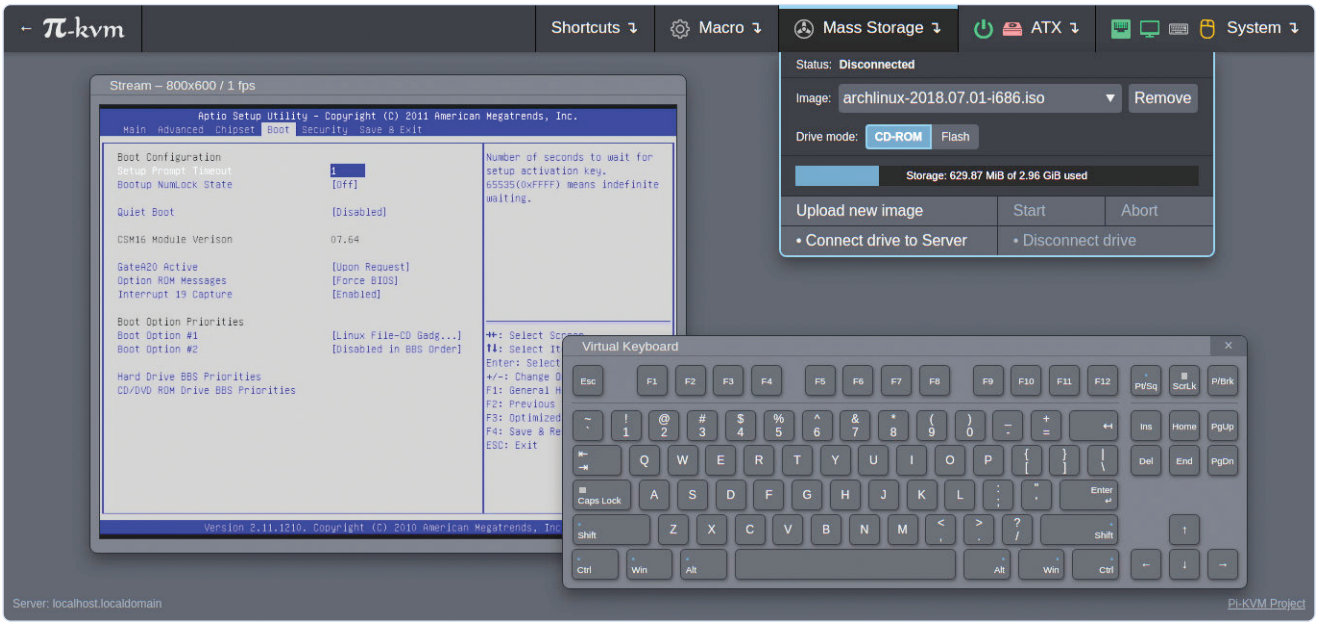


Bild 1. Benutzeroberfläche von PiKVM.

Die PiKVM-Software [1] ermöglicht es, einen Raspberry Pi 4 in eine kostengünstige Fernbedienung für andere Computer zu verwandeln. Der Raspberry Pi wird lokal mit dem zu steuernden Computer verbunden. Er greift auf das Monitorbild zu und emuliert eine USB-Maus, eine Tastatur und einen USB-Stick. Die Signale werden dann von der PiKVM-Software [2], die auf dem Raspberry Pi läuft, über das Internet zugänglich gemacht. Aufseiten des steuernden Computers wird lediglich ein Webbrowser benötigt. Das PiKVM-Projekt (Benutzeroberfläche in **Bild 1**) ist quelloffen und die Kosten für die benötigte Hardware betragen weniger als 100 € für die DIY-Version, ein Bruchteil ähnlicher kommerzieller Produkte. Das wirft einige Fragen auf, die wir Maxim Devaev, dem Entwickler von PiKVM gestellt haben.

* Anmerkung des Übersetzers: Der Name Девaев wird nach korrekter deutscher Transkription Devajew ausgesprochen. Im Internet kursiert allerdings die englische Transliteration Devaev.

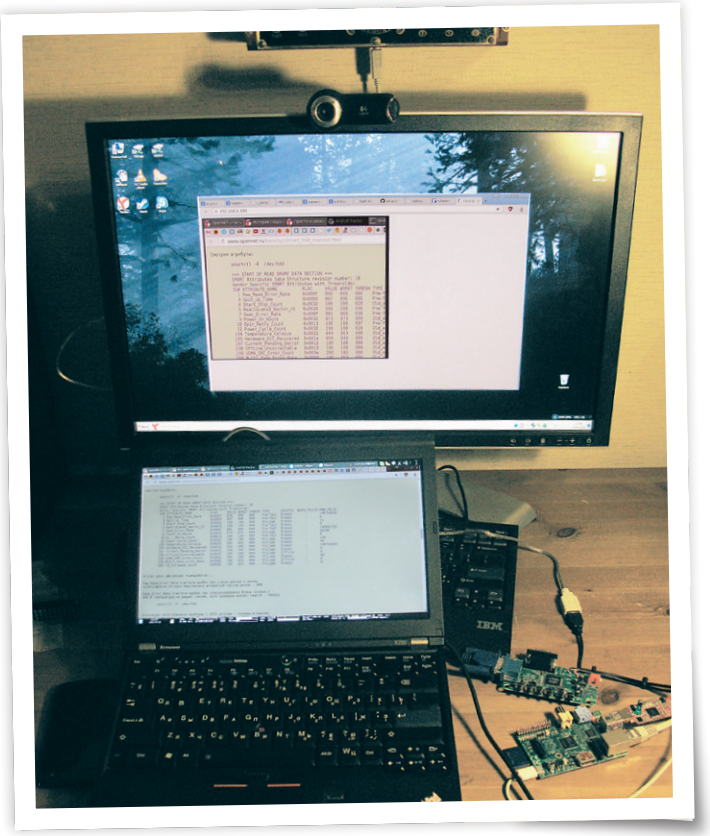


Bild 2. Prototyp von PiKVM auf einem Raspberry Pi 1 (Quelle: Maxim Devaev).

Claußen: Lassen Sie uns mit Ihrem Hintergrund beginnen. Erzählen Sie uns, wie Sie sich für die Elektronik interessiert haben. Haben Sie an der Universität studiert?

Devaev: Sicher, aber es wäre falsch, mich einen Elektronikspezialisten zu nennen. Ich weiß natürlich, wie man einen Lötkolben hält, und ich kann einen einfachen Schaltplan entwerfen und ihn zusammenbauen, ohne das ganze Ding zu verbrennen. Aber in erster Linie bin ich ein Software-Ingenieur. Elektronik ist meine sekundäre und weit weniger entwickelte Fähigkeit.

Ich habe mich schon als Kind für Technik interessiert und seitdem eine Menge wissenschaftlicher und technischer Literatur gelesen. Die Anschaffung meines ersten PCs hatte natürlich große Auswirkungen auf den Inhalt meiner Bücherregale, die sehr informatikbuch-lastig wurden.

Ein paar Jahre später, als ich mich für einen Studiengang an der Universität entscheiden musste, habe ich Informationssicherheit gewählt, brach ihn aber kurz darauf ab, da mich der Lehrplan enttäuschte. Die russische Hochschulbildung ist sehr formal: Es ist oft wichtiger, dass man nicht in den dutzenden humanitären Begleitfächern durchfällt.

Also habe ich mich selbst weitergebildet und einen Job als Softwareentwickler angenommen. Ich arbeitete, arbeitete, arbeitete, verbesserte mich, und einige Zeit später landete ich bei Yandex (eine Art russisches Google). Dort bin ich neun Jahre geblieben und habe in der Zeit Verteilte Systeme entwickelt, löste große Infrastrukturüberwachungs- und -verwaltungsprobleme und all so etwas. Das ist vielleicht nicht das Interessanteste, aber es ist, was es ist.

Claußen: Wie sind Sie auf die Idee gekommen, das PiKVM-Projekt zu starten?

Devaev: Das ist eigentlich eine lustige Geschichte. Ich war einmal zu Besuch bei meinen Eltern, 1400 km von zu Hause entfernt. Mein Heimserver, den ich immer eingeschaltet habe, fiel aus, als ich gerade den Kuchen meiner Mutter aß. Ich war verärgert und musste meinen Freund bitten, zu mir zu fahren und ihn neu zu starten, und beschloss, einen KVM over IP zu kaufen. Ich ging online und sah … 500 Dollar? Ernsthaft? Was ist Wundersames in dem Ding drin, dass es 500 Dollar kostet? Und dann las ich einige Bewertungen und stellte fest, dass man für diese 500 Dollar ein überteuertes Stück Hardware bekommt, gepaart mit schrecklicher, fehlerhafter, proprietärer Software mit Lizenzbeschränkungen. Und die Geräte auf dem Markt waren zudem alle gleich schlecht. Ich weiß nicht, was in mir vorging, aber ich beschloss, diesen Betrag aus Prinzip nicht zu zahlen. Wenn Sie etwas Gutes wollen, machen Sie es selbst. Und der Rest ist Geschichte.

Claußen: Warum haben Sie einen Raspberry Pi für das Projekt verwendet?

Devaev: Der Raspberry Pi war einfach praktisch. Er war neu auf dem Markt und passte perfekt für ein kleines eingebettetes System. Im Grunde hätte es auch jeder andere ARM-Computer sein können, aber der Preis, die Verbreitung und der großartige Herstellersupport spielten eine entscheidende Rolle. Ich wollte keine Zeit damit verbringen, fremde Geräte zu debuggen, eigene Kernel zu erstellen und so weiter. Der Raspberry Pi erlaubte es mir, mich auf die eigentliche Aufgabe zu konzentrieren.

Neben dem Raspberry Pi brauchte ich ein Gerät zur Videoerfassung und eine Möglichkeit, eine Tastatur zu emulieren (an eine Maus dachte ich damals noch nicht). Zu dieser Zeit gab es keine billigen VGA-Grabber auf dem Markt, also entschied ich mich für einen kleinen Hack: Ich verwendete einen Cinch-USB-Dongle, wie er normalerweise zum Digitalisieren alter VHS-Kassetten verwendet wird, und speiste analoges Video über einen billigen VGA-Cinch-Adapter ein. Die Videoqualität war (wie in **Bild 2** zu sehen, Anmerkung des Herausgebers) grauenhaft, aber gut genug, um das BIOS einzurichten und eine Konsole zu benutzen. Für die Tastatur kaufte ich einen PS/2-zu-USB-Wandler und emulierte PS/2-Signale über GPIO. Das war billig, primitiv, aber es funktionierte. Dennoch warf ich letztendlich all diese Hardware weg und entschied mich für eine gängige HDMI-Brücke für Video und tauschte meinen PS/2-Konverter gegen einen Arduino mit selbst entwickelter Software aus.

Claußen: Wie hat das Erscheinen des Raspberry Pi 4 das Projekt beeinflusst?

Devaev: Neben einer besseren Leistung und einer Möglichkeit, die Videoqualität zu verbessern, hat der Pi 4 zwei USB-Controller. Ich konnte einen davon verwenden, um eine Tastatur und eine

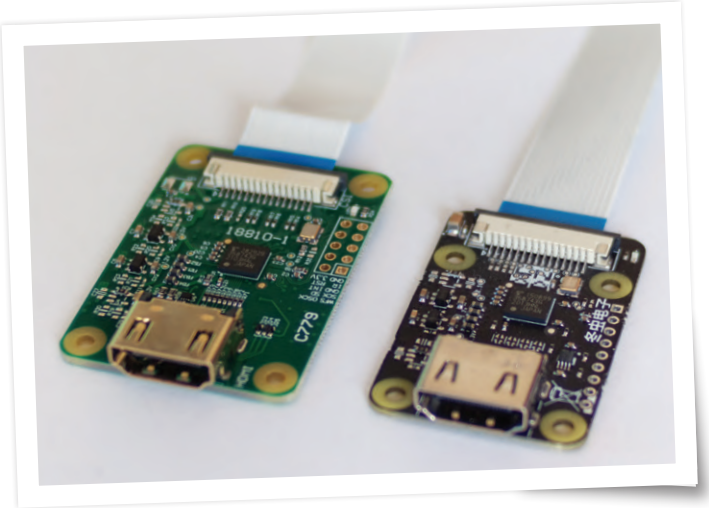


Bild 3. HDMI-zu-CSI-Bridge in zwei Varianten.

Bild 4. HDMI-auf-USB-Dongle.

Maus auf Kernel-Ebene zu emulieren, was es mir ermöglichte, den Arduino wegzulassen. Es gelang mir auch, ein virtuelles CD-Laufwerk zu entwerfen, so dass ich aus der Ferne ein Betriebssystem auf dem Server neu installieren konnte. Damit war der PiKVM den meisten kommerziellen Geräten in Bezug auf Preis und Funktionalität überlegen.

Claußen: Haben Sie PiKVM als „teilzeit-kommerzielles" oder als Hobbyprojekt begonnen?

Devaev: Es war eine Art Hobbyprojekt. Es hat mir Spaß gemacht, meinen Perfektionismus auszuleben. Ich hätte nie gedacht, dass jemand außer mir es brauchen würde, bis ich über ein ähnliches, halb aufgegebenes Projekt auf GitHub (DIY-IPMI) stolperte. Der Autor hatte den gleichen Weg eingeschlagen, aber das Projekt in der Proof-of-Concept-Phase aufgegeben. Ich war begeistert. Ich begann, für mein Projekt zu werben und wie verrückt zu arbeiten, löste ein Problem nach dem anderen, fügte mehr und mehr Funktionalität hinzu und verbrachte Tausende von Stunden mit dem Projekt. Es war großartig zu sehen, wie viele Menschen meine Software zu nutzen begannen. Vorher hatte ich nur kleine Utility-Scripts oder gewichtige Unternehmenssoftware entwickelt.

Claußen: Gibt es Lektionen, die Sie aus diesem Projekt gelernt haben und die Sie gerne mit anderen Entwicklern teilen würden?

Devaev: Okay, hier ist so eine Geschichte. Die Weboberfläche ist der eigentliche Weg, um PiKVM zu nutzen, aber es gibt einen selbstentwickelten Virtual-Network-Computing-Server (VNC) als Alternative. Das ist cool, nicht wahr - einen VNC-Client verwenden, um das BIOS einzurichten! Ich habe MJPEG verwendet, um Videos sowohl für das Web als auch für VNC zu übertragen. Es ist ein einfaches und sehr bandbreitenschonendes Format. Für den Browser konnte ich H.264 und WebRTC verwenden, aber für VNC gab es keine Möglichkeit einer vollwertigen differentiellen Codierung.

Ich habe mich aufgerafft und dem RFBProto (de facto die Beschreibung des VNC-Standards) und dem Betreuer des TigerVNC-Repositorys einen Vorschlag gemacht: Lasst uns eine neue Kodierung erstellen, sie wäre großartig für User mit geringer Bandbreite. Wir diskutierten lange über die Details und kamen schließlich zu einem Format, das uns allen zusagte. Dann ging ich zur IANA und bat sie, diese neue VNC-Erweiterung zu registrieren. Ich fürchtete, dass sie nein sagen würden, da sie bereits ein registriertes Format ohne Beschreibung hatten, aber sie ließen sich darauf ein und erklärten, dass es sich um eine Altlast eines proprietären Clients handelte. Und siehe da, wir entwickeln einen offiziellen TigerVNC-Patch für die allererste quelloffene H.264-über-VNC-Erweiterung. Die Beta-Version läuft reibungslos! Sie ist zwar noch sehr neu, aber einige VNC-Clients und -Server sind bereits daran interessiert, sie zu implementieren.
Die Moral von der Geschichte ist: Scheuen Sie sich nicht, mit anderen Projekten zu kommunizieren, auch wenn sie monumental erscheinen. Und noch etwas: Auch eine einzelne Person kann die Entwicklung des gesamten Protokolls vorantreiben. Man muss nur groß träumen!

Claußen: Hat das Erscheinen des 10-Dollar-USB-HDMI-Dongles geholfen, das PiKVM-Projekt voranzutreiben?

Devaev: Das kommt darauf an. Vor dem USB-Dongle verwendete ich eine HDMI-CSI-Brücke (**Bild 3**), die viele Vorteile hatte. Sicherlich hat der USB-Dongle in **Bild 4** eine große Rolle bei der Verbreitung von PiKVM gespielt, da er sehr billig war, aber ich habe immer davon abgeraten, ihn zu benutzen, da er Hardwareprobleme hat und nicht sehr stabil ist.
Viele Leute haben berichtet, dass sie den USB-Dongle gekauft und es hinterher bereut haben, weil sie oft einen leeren Bildschirm sahen und nichts dagegen tun konnten. Ein USB-Dongle ist eine Blackbox: Man hat keine Ahnung, wie er funktioniert, und wenn

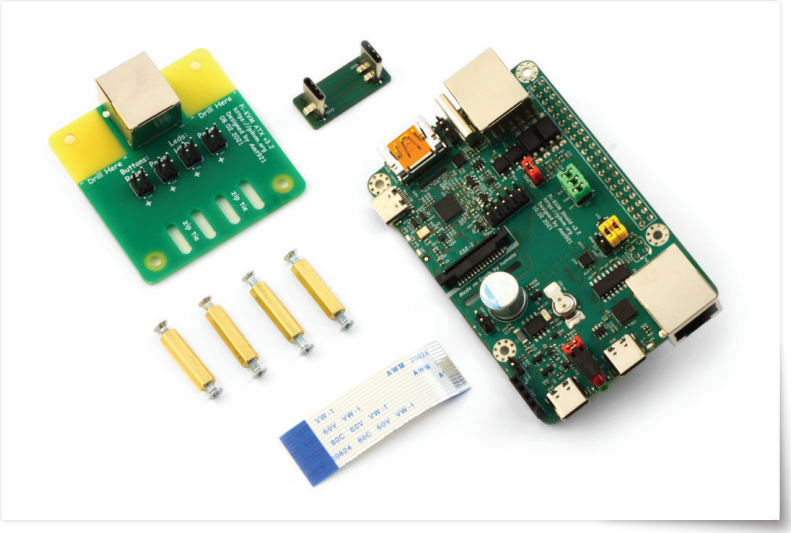


Bild 5. PiKVM v3 HAT (Quelle: Maxim Devaev) [5].

etwas schief geht, kann man es nicht reparieren. Die Bridge hingegen ist viel offener und fast die gesamte Videoverarbeitung wird vom Kernel und meiner Software übernommen.

Claußen: PiKVM ist ein Open-Source-Projekt, das derzeit kostenlos genutzt werden kann. Wie wird seine Entwicklung finanziert? Sie scheinen ja nun Vollzeit an dem Projekt zu arbeiten.

Devaev: Ich würde lügen, wenn ich sagen würde, dass ich eine langfristige Vision, einen Plan und ein Verständnis für den Markt hätte. Alles begann letztes Jahr, als ich meinen Job kündigte, um eine Pause einzulegen und über meine Karriere nachzudenken. Ich setzte die Arbeit an PiKVM als Hobby fort. Die Community wuchs und ich dachte, es wäre toll, wenn PiKVM mein Hauptberuf werden würde, und ich habe einige Open-Source-Projekte gesehen, deren Autoren damit ihren Lebensunterhalt verdienten. Es war eine Win-Win-Situation: Ich würde das tun, was mich wirklich interessiert, und die Benutzer würden für eine kleine Spende qualitativ hochwertige Software erhalten. Als Test richtete ich Patreon ein und kündigte an, dass ich ab sofort Spenden annehmen würde, da ich nun ohne Job war und bereit, PiKVM auf Anfrage zu verbessern (falls eine bestimmte Funktion schnell benötigt wurde). Zu meiner Überraschung begann ich einige Monate später, etwas Gewinn zu machen. Es war weit weniger als mein vorheriges Gehalt, aber wenn man bedenkt, dass ich kein Geld mehr für Fahrten ins Büro ausgeben musste, genug Schlaf bekam und sogar etwas Gewicht verlor, war es ein fairer Handel. Auch die User waren zufrieden, denn es konnte nun nur eine halbe Stunde dauern, bis ein kritischer Fehler gefunden und behoben war, und ich fügte neue Funktionen innerhalb weniger Tage hinzu.

Claußen: Wie unterstützt die Community Ihr Projekt, abgesehen davon, dass sie Ihnen hilft, eine Vollzeitentwicklung zu finanzieren?

Devaev: Ich bin nicht sehr gut darin, Dokumentation zu schreiben, also helfen mir viele Leute dabei. Einige schreiben Anleitungen und Skripte, denken sich neue Wege aus, um einige der PiKVM-Funktionen zu nutzen und beschreiben sie. Einige erstellen Fehlerberichte und geben mir Fernzugriff auf ihre Hardware, damit ich Probleme beheben kann. Und das Wichtigste von allem: Als die Anzahl der Nutzer wirklich groß wurde, wurde es unmöglich, jeder einzelnen Person auf Discord [3] zu antworten, also kümmern sich aktive User um neue Nutzer und wenden sich nur noch an mich, wenn sie ein Problem nicht selbst lösen können. Ich bin diesen Leuten sehr dankbar, da ich mich nun ganz dem Schreiben von Software widmen kann.

Manchmal gibt es Leute, die Patches mit neuen Funktionen oder Fehlerkorrekturen einbringen, aber von diesen Leuten gibt es noch nicht viele, und ich bin im Moment praktisch der einzige PiKVM-Entwickler. Ich plane, einige Leute in Vollzeit einzustellen, die mir bei der Software helfen, wenn wir anfangen, Geräte zu verkaufen. Wenn Sie also unsere Hardware kaufen, unterstützen Sie auch die Entwicklung von Open-Source-Software.

Claußen: Wenn jemand Ihre Arbeit unterstützen möchte, was wäre der beste Weg, dies zu tun?

Devaev: Neben Spenden ist auch Hilfe bei Dingen willkommen, die wir noch nicht allein bewältigen können. Einige dieser Probleme sind in unseren GitHub-Issues [4] beschrieben.

Claußen: Gibt es irgendwelche Pläne, Ihre Arbeit zu kommerzialisieren?

Devaev: Im Moment sind Spenden meine Haupteinnahmequelle. Mir wird oft gesagt, ich könnte eine kommerzielle Lizenz für die PiKVM-Software mit erweiterten Funktionen verkaufen, aber das möchte ich aus zwei Gründen nicht tun. Erstens ermöglicht PiKVM einen direkten Zugriff auf die Hardware. Ich würde so etwas nicht trauen, wenn es nicht Open Source wäre. Und zweitens würde eine Bezahlschranke unweigerlich dazu führen, dass jemand meine offene Software „forkt" und Profi-Funktionen kostenlos entwickelt, um einen Teil der Community für sich zu gewinnen.

Die andere und bald primäre Möglichkeit, wie ich mit meiner Entwicklung Geld verdienen kann, ist der Verkauf meiner eigenen Hardware *PiKVM v3 HAT* [5] (**Bild 5**), die Sie anstelle der DIY-Version verwenden können, um mehr Stabilität und zusätzliche Hardwarefunktionen zu erhalten. Aber ich habe nicht vor, PiKVM DIY aufzugeben und die Benutzer zu zwingen, die v3 zu kaufen. Ich strebe nicht nach Reichtum um jeden Preis, nur nach genug Geld, um mich und meine Katze zu ernähren.

Claußen: Können Sie uns ein wenig über diese Ihre Hardware PiKVM v3 HAT erzählen?

Devaev: Um einen PiKVM zu bauen, muss man seine Hände und eventuell einen Lötkolben benutzen. Ich habe beschlossen, es dem User etwas einfacher zu machen, die Stabilität der Hardware zu verbessern und einige Funktionen hinzuzufügen, die schwer selbst zu machen sind. Obwohl das ursprüngliche Konzept des Geräts

von mir stammt, konnte ich kein komplett eigenes Board entwickeln (das ist eine Sache, die ich nicht beherrsche), also habe ich einen Freund mit Hardwarekenntnissen eingeladen, um mit mir an diesem Projekt zu arbeiten. Unsere PiKVM v3 HAT hat einen eingebauten Server-Power-Controller, eine Video-Capture-Einheit, einen USB-Switch (um das Herausziehen eines Kabels zu emulieren, das ist manchmal nützlich) und einen seriellen Port, um eine Verbindung zu einem Server oder einer Konsole in einem Rack herzustellen. Auch eine Uhr für eine präzise Protokollierung befindet sich auf dem Board. Im Grunde handelt es sich um eine Lösung, die besser und billiger ist als all die 500-Dollar-Geräte, die es auf dem Markt gibt. Sie müssen sich nicht auf mein Wort verlassen, Sie können die v3-Bewertungen auf YouTube nachlesen. Und bald wird es jeder kaufen können.

Claußen: Sie haben auch ein Gehäuse für den Raspberry Pi 4 und Ihr Add-on, das aus Metall gefertigt ist. Gab es einen Grund, keinen Kunststoff zu verwenden? Wir haben auf Discord gelesen, dass es in Russland einfacher ist, etwas aus Metall zu bauen als aus Gummi oder Plastik.

Devaev: Ja, das stimmt. Das Metallgehäuse ist schwerer, stabiler und viel billiger als ein Plastikgehäuse. Diejenigen, die die v3- HAT auf Kickstarter bestellt haben, konnten dieses Gehäuse kostenlos erhalten. Später wird es in Online-Shops verkauft werden. Aber wir haben auch eine kostenlose Zeichnung des Gehäuses für den 3D-Druck, wenn Sie es selber herstellen möchten.

Claußen: Wird Ihr Kit auch in Geschäften außerhalb Russlands erhältlich sein?

Devaev: Die erste Charge sollte in Kürze in den USA erhältlich sein, und die zweite ab Herbst 2021 weltweit.

Claußen: Welche Pläne für den PiKVM haben Sie für die Zukunft?

Devaev: Wir werden sowohl die Software als auch die Hardware weiterentwickeln. Wir planen, die Videoverarbeitung mit 1080p/60 Hz zu implementieren (das ist derzeit aufgrund der Einschränkungen des Pi 4 nicht möglich) und eine sichere Lösung für den Cloud-Zugang für Benutzer ohne offene externe IP-Adresse zu schaffen.

Claußen: Haben Sie noch andere Projekte in der Entwicklung? Können Sie uns ein paar Details darüber verraten, woran Sie gerade arbeiten?

Devaev: uStreamer, denke ich. Es wurde als Aufnahmegerät für PiKVM entwickelt, aber es ist eine völlig separate Komponente, die viele Leute als Server für Webcam-Videos verwenden. Ich habe es von Grund auf als Alternative zum MJPG-Streamer geschrieben, um das Leistungsproblem zu lösen. Jetzt ist uStreamer der schnellste spezialisierte MJPEG/H.264-Videoservice, und darauf bin ich wirklich stolz.

Claußen: Maxim Devaev, vielen Dank für das Gespräch. Für diejenigen, die sich jetzt auf den Weg zur Kickstarter-Kampagne [5] machen, haben wir gute Nachrichten. Die Kampagne war mehr als erfolgreich. Das bedeutet, dass man den PiKVM v3 HAT bald online bestellen kann. Werfen Sie auch einen Blick in den Elektor-Store [6], denn unser Einkaufsteam arbeitet derzeit daran, Ihnen in naher Zukunft auch einige PiKVM v3 HATs anbieten zu können.

210405-02

Haben Sie Fragen oder Kommentare?

Haben Sie technische Fragen oder Kommentare zu diesem Artikel? Schicken Sie eine E-Mail an den Autor unter mathias.claussen@elektor.com oder kontaktieren Sie Elektor unter redaktion@elektor.de.

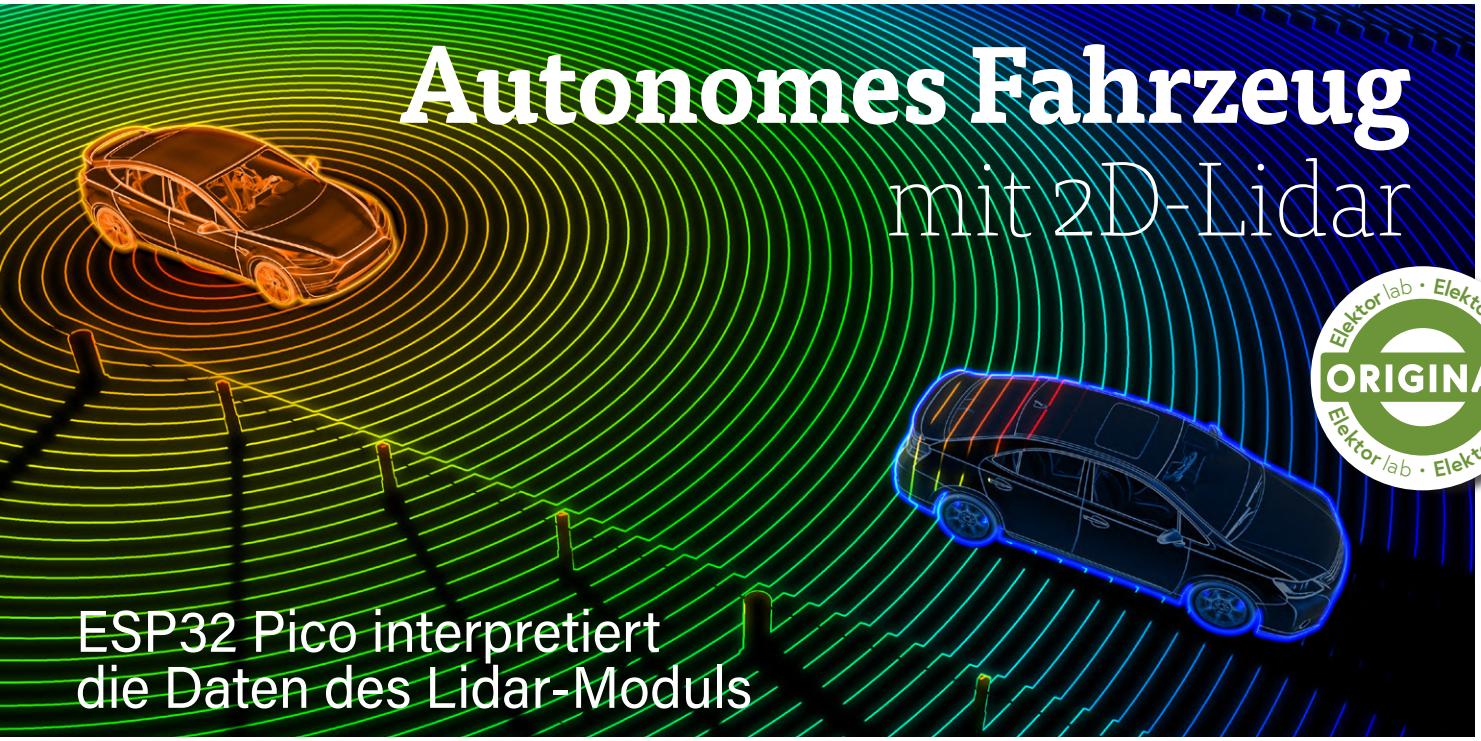
PASSENDE PRODUKTE

- Raspberry Pi HDMI to CSI-2 Adapter Board (supports up to 1080p25fps) (SKU 19707)
  www.elektor.com/19707

WEBLINKS

[1] M. Claußen, „Raspberry Pi als KVM-Fernbedienung“, Elektor 1-2/2022: http://www.elektormagazine.de/200523-02
[2] Pi-KVM-Homepage: https://pikvm.org/
[3] Pi-KVM auf Discord: https://discord.com/invite/bpmXfz5
[4] Pi-KVM GitHu-Issues: https://github.com/pikvm/pikvm/issues
[5] Pi-KVM v3 HAT auf Kickstarter: https://www.kickstarter.com/projects/mdevaev/pikvm-v3-hat
[6] Elektor-Store: https://www.elektor.de/new/new-in-the-store

Autonomes Fahrzeug mit 2D-Lidar



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

Lidar ermöglicht es Robotern und anderen selbstfahrenden Fahrzeugen, Hindernisse in ihrer Umgebung zu erkennen, ohne sie zu berühren, was ihnen viel mehr Bewegungsfreiheit verleiht. Um ein Gefühl für diese Technik zu bekommen, habe ich einen einfachen ferngesteuerten Wagen gebaut und ihn in meinem Wohnzimmer herumfahren lassen.

elektor TV
Sehen Sie sich dieses Projekt im Video an!

Lidar ist ein Akronym für *Light Detection and Ranging*. Lidar ist wie Radar, mit dem Unterschied, dass es Licht statt Radiowellen verwendet. Die Lichtquelle ist ein Laser. Ein Lidar sendet Lichtimpulse aus und misst die Zeit, die eine von einem entfernten Objekt abprallende Reflexion benötigt, um zum Gerät zurückzukehren. Da die Lichtgeschwindigkeit eine bekannte Konstante ist, kann die Entfernung zum Objekt anhand der Laufzeit des Lichtimpulses berechnet werden (**Bild 1**).

Zweidimensionales Lidar

Lidar kann wie etwa bei einem Laser-Entfernungsmesser eindimensional (1D) sein. Es kann auch zweidimensional (2D) sein, so wie die Radargeräte, die auf Schiffen und auf den Towers der Flughäfen eingesetzt werden. Es gibt auch 3D-Lidar, das zum Beispiel in Flugzeugen eingesetzt wird, um eine dreidimensionale Darstellungen der Erdoberfläche unter ihnen zu erstellen. Für dieses Projekt wird ein 2D-Lidar verwendet.

Im Grunde genommen ist ein 2D-Lidar nichts anderes als ein rotierendes 1D-Lidar. Anstatt den Laser und den Detektor zu drehen, ist es oft einfacher, das Licht auf einen rotierenden Spiegel zu werfen. Durch das Aussenden von Lichtimpulsen in regelmäßigen Abständen können 360° abgedeckt und eine Entfernungskarte erstellt werden, mit dem Lidar im Zentrum des Plots.

Dabei ist das Reflexionsvermögen von Objekten wichtig ist. Ein idealer schwarzer Körper kann von einem Lidar nicht gesehen werden, da er überhaupt kein Licht reflektiert.

Anschließen des Lidars

Für meine Experimente habe ich das Lidar X4 der Firma Ydlidar verwendet (**Bild 2**). Es hat eine maximale Reichweite von 10 m und eine Winkelauflösung von 0,5° (für Entfernungen bis 50 cm). Es verfügt über einen Infrarotlaser mit einer Wellenlänge von 785 nm. Bei diesem Gerät wird die Laser- und Detektoreinheit gedreht, ein Drehspiegel kommt nicht zum Einsatz.

Das X4-Lidar gibt Entfernungsdaten als kontinuierlichen seriellen Datenstrom mit 128.000 Baud aus. Über diese serielle Verbindung wird das Gerät auch gesteuert. Sie können mit einfachen Befehlen das Abtasten starten und stoppen, Sie können auch einige Informationen über das Gerät abfragen. Der Motor wird separat über zwei zusätzliche Verbindungen

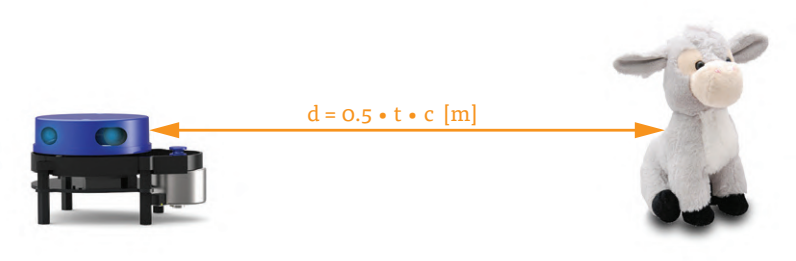


*Bild 1. So berechnet man Entfernungen. **t** ist die Laufzeit des Lichtimpulses in Sekunden und **c** ist die Lichtgeschwindigkeit in m/s. Der Faktor 0,5 korrigiert die Tatsache, dass der Impuls zum Objekt hin und wieder zurück muss, also die doppelte Strecke zurücklegt.*

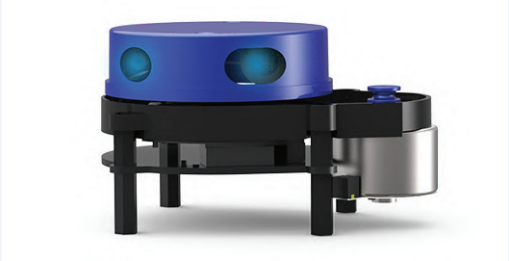


Bild 2. YDLIDAR X4 ist ein kostengünstiges 2D-Lidar mit völlig ausreichender Reichweite und Präzision, um einen Roboter zu bewegen, ohne dass dieser gegen Objekte stößt.

gesteuert, eine für Ein/Aus und eine für die Geschwindigkeit. Das bedeutet, dass dieses Lidar auch im eindimensionalen Modus arbeiten kann, wenn sich der Motor nicht dreht.
Ich habe den seriellen Port und die Motoranschlüsse an ein ESP32-Pico-Kit angeschlossen (**Bild 4**). Um Strom zu sparen, habe ich den Motor des Lidars nur mit seiner niedrigsten Drehzahl betrieben, die bei etwa 400 U/min liegt. Das Lidar zieht dann einen Strom von etwa 400 mA.

Parsing der Lidar-Daten

Zur Programmierung des ESP32-Pico-Kits habe ich die Arduino-IDE verwendet. Nachdem ich eine Funktion zum Parsen der Lidar-Daten geschrieben hatte, musste ich meine Interpretation der Daten auch überprüfen. Das X4-Entwicklungshandbuch stellt nicht sehr klar, wie dies zu bewerkstelligen wäre und gibt zwei Detailstufen an. Die zweite Stufe bietet eine bessere Winkelauflösung, erfordert aber viele Arkustangens-Berechnungen, die sehr rechenintensiv sind. Daher habe ich zunächst die einfache Interpretation ausprobiert.
In einem rechteckigen, geschlossenen Raum stellte ich das Lidar auf einen Tisch und ließ es eine Weile scannen. Danach ließ ich einen gemittelten 360°-Scan über eine serielle Schnittstelle als kommagetrennte Werte (CSV) ausgeben, die ich in Excel lud. Mit der Funktion Netzdiagramm konnte ich es anzeigen (**Bild 4**). Das Ergebnis sah meinem Rechteck sehr ähnlich, und auch die Entfernungen waren einigermaßen korrekt, so dass ich mir nicht die Mühe machte, die Qualität durch Arkustangensberechnungen zu verbessern.

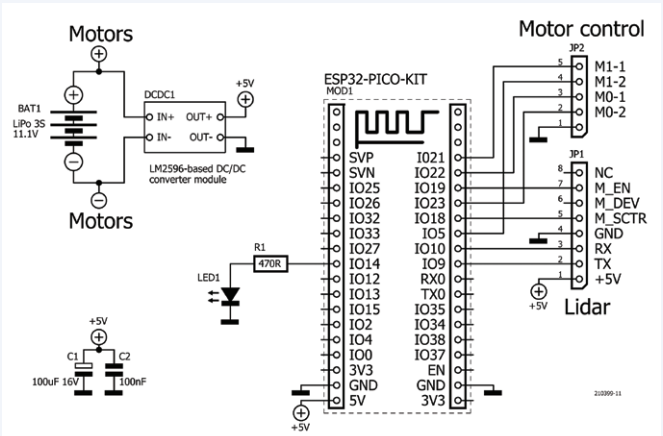


Bild 3. Das ESP32-Pico-Kit ist das Gehirn des autonomen Fahrzeugs, der Motortreiber und der Gleichspannungswandler sind handelsübliche und online erhältliche Module.

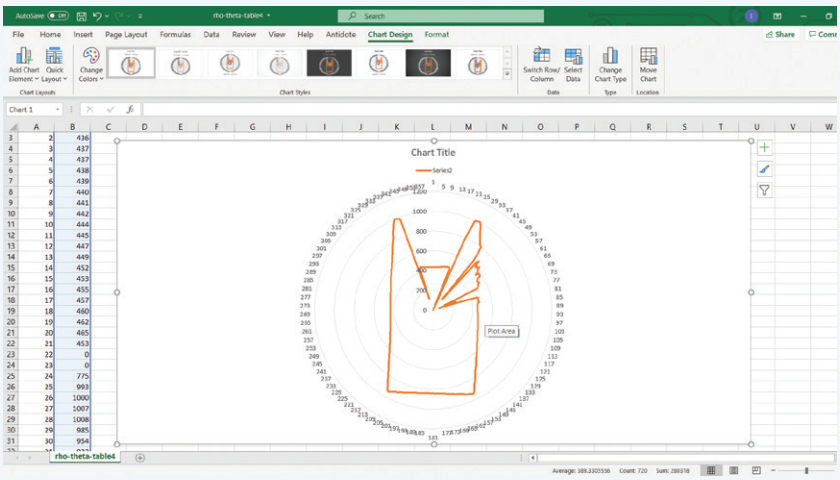


Bild 4. Mit Excel konnte ich meine Interpretation der Lidar-Daten schnell überprüfen. Die kurze horizontale Linie bei 400 mm (und der „Funkschatten" dahinter) sind auf meinen Laptop zurückzuführen, der die dekodierten Lidar-Daten aufzeichnet.

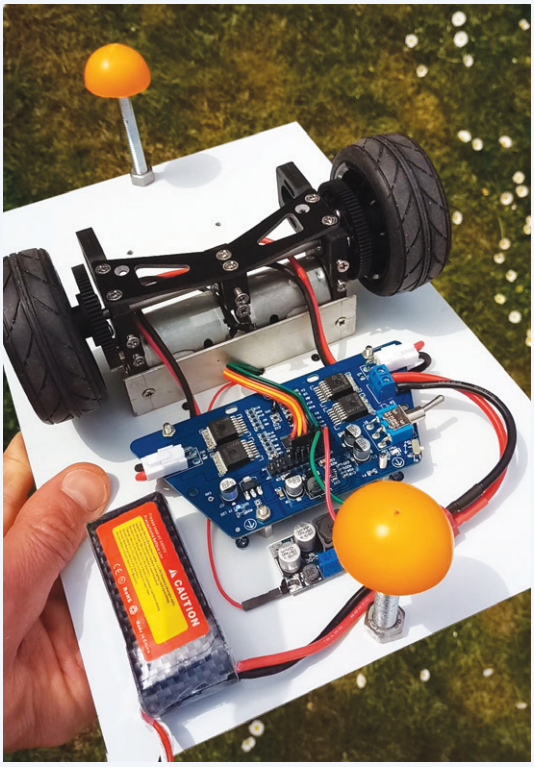
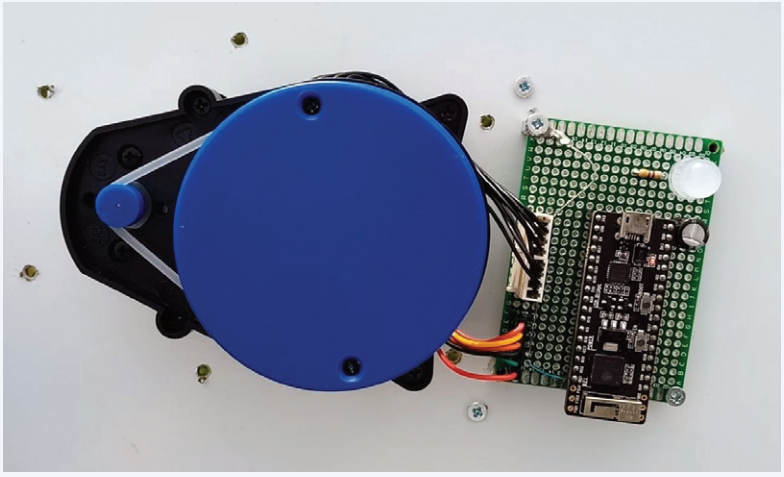


Bild 5. Blick auf den Unterboden des Fahrzeugs. Motor und die Motortreiber wurden einst von Landzo.com bezogen, sind aber nicht mehr erhältlich.

Bild 6. Oberseite des Fahrzeugs mit dem ESP32-Pico-Kit und dem Lidar.



Ein kleiner Roboter entsteht

Als nächsten Schritt baute ich einen einfachen ferngesteuerten Wagen, auf den ich das Lidar montierte. Der Wagen hat zwei angetriebene Räder in der Mitte und je einen Abstandshalter an beiden Enden. Auf diese Abstandshalter habe ich halbierte Tischtennisbälle geklebt, um das Gleitverhalten zu verbessern. Als Bodenblech (und gleichzeitig Chassis) dient eine Platte aus doppelseitig beschichtetem FR4-Platinenmaterial. Die Räder mit den Motoren, die Motortreiberplatine und die Stromversorgung, ein 11,1-V-Li-Po-Akku des Typs 3S, sind auf der Unterseite montiert (**Bild 5**), das ESP32-Modul und das Lidar auf der Oberseite (**Bild 6**). Die Platine ist mit Masse verbunden und schirmt so das ESP32-Modul von den „lauten" Motoren ab. Der Mittelpunkt des Lidars befindet sich in der Mitte der Montageplatte. Auch die Radachse befindet sich in der Mitte. Dieses einfache Auto kann sich drehen und ist ziemlich wendig und manövrierfähig.

Fernsteuerung über Bluetooth hinzufügen

Als Fernsteuerung habe ich die kostenlose und quelloffene Dabble-Bibliothek [1] verwendet, die eine Bluetooth-Steuerung für den ESP32 und Arduino mit einer Smartphone-App mit mehreren Bedienoberflächen verbindet (**Bild 7**). Eine davon ist ein Gamepad, das meiner Anwendung wie angegossen passte. Es ist wirklich einfach zu bedienen und ermöglichte die Steuerung des Wagens mit meinem Handy.

Ein Wegfindungsalgorithmus

Mein Ziel war es, den Wagen so zu programmieren, dass er ganz von selbst herumfährt, ohne gegen Objekte wie Möbel oder andere Gegenstände zu stoßen. Ein beliebter Ansatz ist es, den Wagen herumlaufen zu lassen und zurückzufahren oder wegzulenken, wenn er einem Objekt zu nahe kommt, aber das erfordert, dass der Wagen Entscheidungen trifft. Ich wollte etwas Einfacheres. Es gibt viele Beispiele für einfache Algorithmen, die zu komplexem Verhalten führen, zum Beispiel die Art und Weise, wie ein Vogelschwarm zusammenbleibt [2], und so etwas wollte ich auch.

Meine Idee war, den Wagen immer in die Richtung des größtmöglichen Abstands zu einem Objekt fahren zu lassen, die das Lidar ermittelt. Um zu vermeiden, dass er sich im Kreis dreht, schaut er nur nach vorne, in einem Bereich von -90° bis +90°. Die Umsetzung dieser Regel war recht einfach. Für jeden Scan wird eine Tabelle mit der durchschnittlichen Entfernung für jeder einzelne Winkelgrad aktualisiert. Die Tabelle hat also 360 Einträge. Diese Tabelle wird dann in 10-Grad-Segmente unterteilt (was ziemlich willkürlich ist) und ermittelt, welches dieser Segmente den höchsten gemittelten Abstand aufweist. Die Mitte dieses Segments ist die Richtung, die der Wagen einschlagen soll. Um dies zu erreichen, dreht sich der Wagen so, dass der mittlere Winkelgrad, also die Richtung, zu 0° wird. Wir haben nun einen klassischen Steuerungsalgorithmus, der versucht, einen „Fehler" zu minimieren (**Bild 8**).

Erster Testlauf

Zu meiner großen Überraschung gelang es dem Wagen schon bei seinem ersten Probelauf mit diesem einfachen Algorithmus, sich sicher durch unser Wohnzimmer zu bewegen, ohne gegen Hindernisse zu stoßen (**Bild 9**). Er umrundete unsere Couch und fuhr ohne Schwierigkeiten durch enge Passagen. Der Wagen hatte keinerlei Kenntnisse über seine Umgebung oder sich

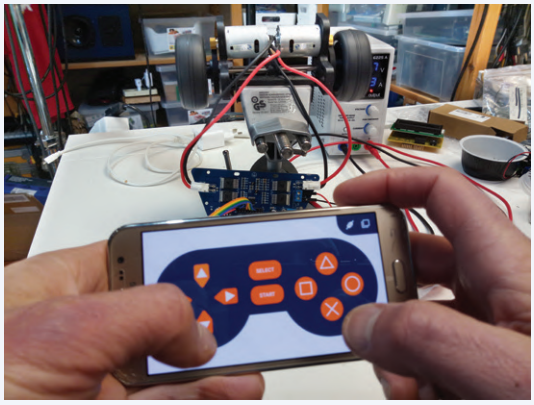


Bild 7. Ausprobieren mit der Smartphone-Daddel-Fernsteuerung auf dem Prüfstand.

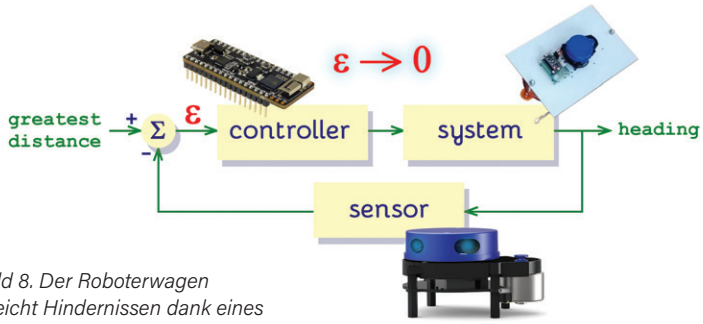


Bild 8. Der Roboterwagen weicht Hindernissen dank eines einfachen, fehlerminimierenden Steuerungsalgorithmus aus.

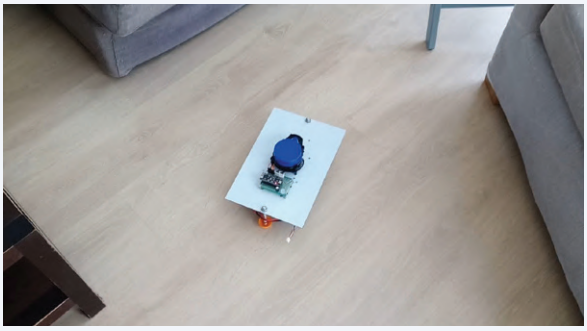


Bild 9. Go cart, go! Sehen Sie im Video [4], wie geschmeidig er sich bewegt.

selbst, etwa seine eigenen Abmessungen. Ich hatte auch nicht versucht, irgendetwas zu optimieren. Alle Parameter wie Vorwärts- und Drehgeschwindigkeit und Suchwinkel wurden einfach auf Werte eingestellt, die ich für vernünftig hielt.

Die Bluetooth-Fernbedienung hat sich als sehr praktisch erwiesen, um den Wagen zu „verwirren" oder ihm aus schwierigen Situationen herauszuhelfen. Außerdem kann man mit ihr die Parameter während der Fahrt ändern. Da ich mich nicht so sehr für die Optimierung interessiere, sondern eher für die Erprobung von Konzepten, habe ich an dieser Stelle aufgehört. Wenn Sie es selbst ausprobieren möchten, finden Sie im Kasten die Links zum Code. Es gibt viele Möglichkeiten, diesen Entwurf zu verbessern, und es ist noch ein weiter Weg bis zu einem autonomen Staubsauger oder Rasenmäher, aber die erzielten Ergebnisse sind sehr ermutigend. Die Software für dieses Projekt, ein Arduino-Sketch für den ESP32, kann unter [3] heruntergeladen werden. Ein Video ist verfügbar unter [4] und ein kleiner Bonus unter [5].

210399-02

Haben Sie Fragen oder Kommentare?

Haben Sie technische Fragen oder Kommentare zu diesem Artikel? Schicken Sie eine E-Mail an den Autor unter clemens.valens@elektor.com oder kontaktieren Sie Elektor unter redaktion@elektor.de.

Ein Beitrag von

Idee, Design, Text und Fotografien: **Clemens Valens**
Redaktion: **Jens Nickel**, **C. J. Abate**
Übersetzung: **Rolf Gerstendorf**
Layout: **Giel Dols**

PRODUKTE

- **ESP32 Pico Kit (SKU 18423)**
  www.elektor.de/18423
- **YDLIDAR X4 (SKU 18601)**
  www.elektor.de/18601

WEBLINKS

[1] Dabble: https://thestempedia.com/product/dabble/
[2] Schwarmverhalten: https://de.wikipedia.org/wiki/Schwarmverhalten
[3] ESP32-Arduino-Sketch für dieses Projekt: https://github.com/ClemensAtElektor/Lidar-controlled-autonomous-vehicle/
[4] Sehen Sie sich dieses Projekt im Video an: https://youtu.be/BmNelv_gR9Q
[5] Lidar-basiertes Parklicht von Rob Reynolds (SparkFun): https://youtu.be/KRfidalgJx8

Raspberry Pi Zero 2 W

Ein erfreuliches und notwendiges Update

Von **Mathias Claußen** (Elektor-Labor)

Der Raspberry Pi Zero mit seinem ARM11-Kern ist inzwischen in die Jahre gekommen und seine Leistung für viele Anwendungen mehr als grenzwertig. Nun wird der Raspberry Pi Zero 2 W mit einem neuen SiP seine Nachfolge antreten. Was bringt uns dieser neue Raspberry Pi?

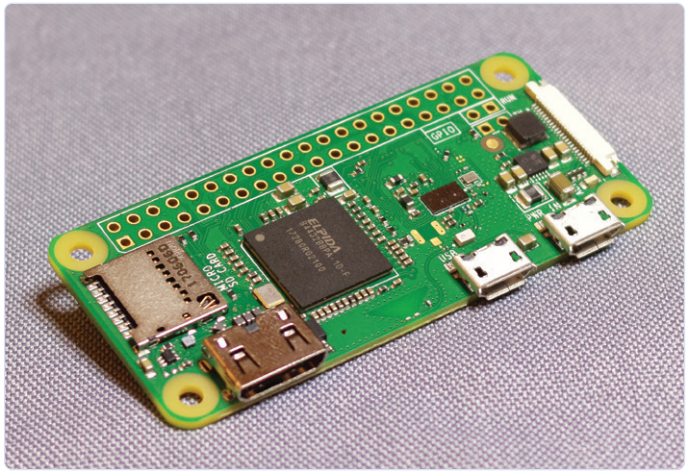


Bild 1. Raspberry Pi Zero

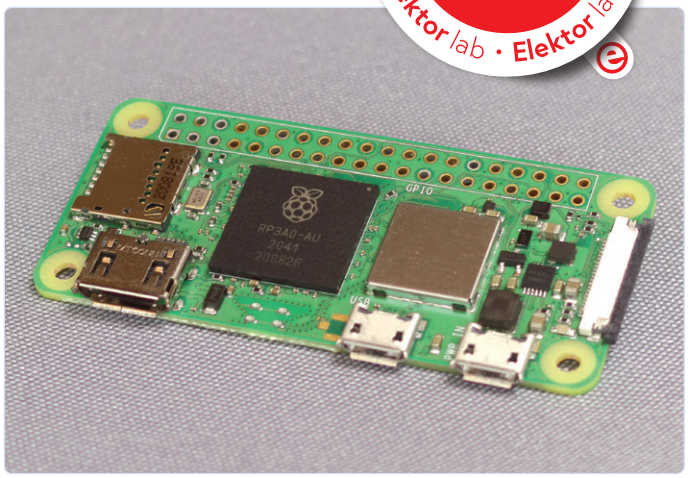


Bild 2. Raspberry Pi Zero 2 W.

Der Raspberry Pi Zero W (**Bild 1**) hat als kleines, nur etwa 10 Euro teures Board in vielen Anwendungen (wie Audioplayer, Kameras oder einfache Steuerungen) seinen Dienst verrichtet. Mit seiner Einführung im Jahr 2017, zum fünften Geburtstag des Raspberry Pi, bekam man einen kleinen Linux-fähigen Rechner, auf Basis des SoC aus dem Raspberry Pi 1. Der nun 10 Jahre alte ARM11-Kern ist inzwischen alles andere als zeitgemäß, und so praktisch Preis und Formfaktor auch sein mögen, der Wunsch der Anwender nach mehr Leistung ist gegeben. Ob es eine verfrühte Überraschung zum 10-jährigen Geburtstag des Raspberry Pi ist oder eine stark verspätete zum neunten, muss jeder selbst entscheiden, aber der Raspberry Pi Zero 2 W (**Bild 2**) ist jetzt da. Elektor hatte bereits Gelegenheit, vorab erste Erfahrungen mit dem Board zu sammeln.

Drop-in Replacement

Die gute Nachricht vorweg: Am Layout der Anschlüsse hat sich nichts geändert. **Bild 3** zeigt einen Raspberry Pi Zero W und einen Raspberry Pi Zero 2 W, Seite an Seite. Alles Zubehör und alle Gehäuse können ohne Sorge weiterverwendet werden. Wie vom Raspberry Pi Zero

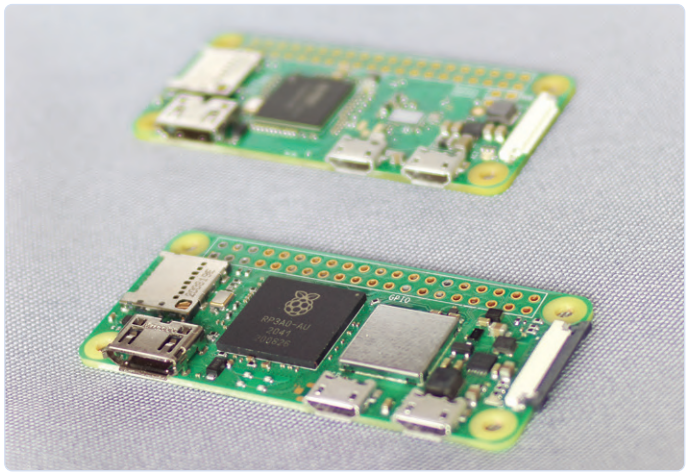


Bild 3. Raspberry Pi Zero 2 W und Raspberry Pi Zero W (Hintergrund).

Tabelle 1: Spezifikationen

- Broadcom BCM2710A1, Cortex-A53 (ARMv8) 64-bit SiP @ 1.0GHz
- 512MB LPDDR2 DRAM
- VideoCore IV GPU
- IEEE 802.11.b/g/n Wireless LAN, Bluetooth 4.2/BLE
- 40-pin GPIO-Header
- Mini-HDMI-Port
- Micro-USB 2.0 OTG Port
- CSI-Kamera-Port
- Micro-SD-Slot
- Micro-USB Power in (5 V DC/2,5 A)
- Composite Video und Reset-Pin als Testpunkte vorhanden

gewohnt, stehen auch hier zwei Micro-USB-Anschlüsse, der gewohnte 40-Pin-Verbinder und ein Mini-HDMI-Ausgang bereit. Bei einem der beiden USB-Anschlüsse handelt es sich auch hier wieder um einen USB-OTG-Port, an den sich per Adapter USB-Zubehör anschließen lässt - oder mit dem der Raspberry Pi Zero 2 selbst zum USB-Gerät werden kann. Zum Booten wird wie gewohnt eine Micro-SD-Karte verwendet, bei der (wie immer) gilt, dass eine höhere Lesegeschwindigkeit die Bedienung des Raspberry Pi Zero 2 W flüssiger macht. **Bild 4** zeigt die Unterseite des Raspberry Pi Zero 2 W, auf der sich keine Komponenten befinden. In **Bild 5** ist die Oberseite des Raspberry Pi Zero 2 W zu sehen, auf der besonders das SiP mit der Beschriftung RP3A0-AU auffällt. Während frühere Modelle alle einen Chip mit der Bezeichnung „BCM" hatten, erinnert diese eher an den RP2040 des Raspberry Pi Pico. So stellt sich die Frage, was ist das für ein Chip und welche Spezifikationen hat der Raspberry Pi Zero 2 W?

Der RP3A0-AU SiP

Wenn man den Namen „RP3A0" auseinandernimmt, so erhält man „RP" für Raspberry Pi, die „3" gibt die SiP-Generation an, die in dem Chip verbaut ist, also vom Kern her ein Raspberry Pi 3. Die genauen Spezifikationen können der **Tabelle 1** entnommen werden: Es handelt sich, vereinfacht gesagt, um einen Raspberry Pi 3 A+ im Formfaktor eines Raspberry Pi Zero. Wer auf einen Raspberry-Pi-4-basierten Zero gehofft hat, sollte bedenken, dass auch im Raspberry Pi Zero W ein Chip verwendet wurde, der schon etwa fünf Jahre im Feld war. Setzt sich dieser Trend fort, dürften wir frühestens 2026 mit dem Einzug des BCM2711 aus dem Raspberry Pi 4 rechnen.

Der Raspberry Pi 3A+ basiert auf einem BCM2837B0 mit einem maximalem Takt von 1,4 GHz. Beim RP3A0 ist dies ein BCM2710A1, der nur mit 1,0 GHz taktet, was dem Energiebedarf und der Wärmeentwicklung zugutekommt. Weitere Optimierungen des Energiebedarfs für das Raspberry-Pi-3-basierte System sind noch zu erwarten, wie uns Eben Upton in einem Interview [1] verraten hat. Der Einsatz eines SiPs aus der Serie des Raspberry Pi 3 bedeutet, dass auf eine stabile und breite Softwarebasis zurückgegriffen werden kann. SD-Karten-Images für den Raspberry Pi 3 booten ohne weitere Anpassungen. Eine der ersten Fragen, die zu klären sind: Wie hat sich die Leistungsaufnahme verändert? Kann ich meine bisherigen Spannungsversorgungen weiter benutzen? Welche neuen Möglichkeiten habe ich und wie schaut es mit der Performance aus?

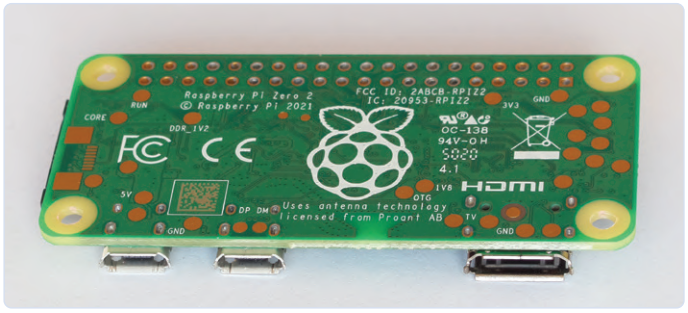


Bild 4. Unterseite des Raspberry Pi Zero 2 W.

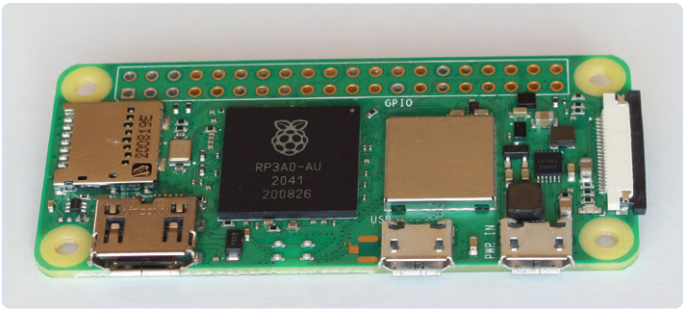


Bild 5. Oberseite des Raspberry Pi Zero 2 W.

Klein und weniger Energiehunger

Durch den minimalistischen Ansatz und die Verwendung von optimierten und lange erprobten Chips zeichnet sich die Raspberry-Pi-Zero-Serie durch einen minimierten Energieverbrauch aus. Beim Raspberry Pi Zero 2 W liegt dieser mit aktivem HDMI-Ausgang und aktiver WLAN-Verbindung bei etwa 2,5 W maximal; das Board kann damit sogar aus einem USB-Port eines Computers gespeist werden. Im Idle-Modus, also bei ruhendem Desktop, sind es aktuell etwa 0,7 W, wobei dies sicherlich noch nicht das untere Ende des Energiebedarfs sein sollte. Ein Raspberry Pi 3B+ zum Vergleich benötigt im Idle-Modus etwa 2 W, also deutlich mehr; unter Last erreicht der Raspberry Pi 3B+ rund 4,4 W bei 1,4 GHz Taktfrequenz.

Die kompakte Baugröße ermöglicht es, den Raspberry Pi Zero in den unterschiedlichsten Applikationen einzusetzen, vom Webradio bis hin zur mobilen Spielekonsole oder zur Automatisierung. Auch als Server für Homeassistant macht sich der neue Raspberry Pi Zero 2 W bestens, durch seinen geringeren Energiebedarf belastet er die Umwelt weniger als die größeren Modelle.

2,4-GHz-WLAN und Bluetooth

Der Raspberry Pi Zero 2 W unterstützt 2,4-GHz-WLAN nach 802.11 b/g/n. Auch Bluetooth und BLE stehen beim Raspberry Pi Zero 2 W aktuell in Version 4.2 bereit. WiFi im 5-GHz-Band bleibt damit den größeren Geschwistern wie dem Raspberry Pi 4 B oder Raspberry Pi 3 B+ vorbehalten.

Erster Boot

Beim Raspberry Pi Zero W waren die Bootzeiten von bis zu mehreren Minuten nicht wirklich erfreulich. Mit dem Raspberry Pi Zero 2 W und einem aktuellen Raspberry Pi OS (32 Bit) erhält man ohne Optimierungen nach 30 Sekunden den Desktop und kann wie gewohnt mit einem Raspberry Pi 3 arbeiten. Wer einen Raspberry Pi Zero verwendet, wird sicherlich schon einen der OTG-Adapter (**Bild 6**) besitzen, um Maus und Tastatur anschließen zu können. Die vier Kerne sorgen hier für eine merkliche Geschwindigkeitssteigerung, auch wenn gegenüber dem Raspberry Pi 3B+ 400 MHz weniger Maximaltakt verfügbar sind.

Bild 6. USB-OTG-Adapter.

Auch die 512 MB RAM bieten noch genug Platz für einige Services, der Raspberry Pi Zero 2 W ist jedoch kein vollwertiger Desktop-Ersatz. Das Öffnen von Webseiten mit 512 MB Ram ist nicht wirklich als schnell zu bezeichnen. Ein Raspberry Pi 4B oder Raspberry Pi 400 bieten deutlich mehr Rechenleistung; die Stärken des Raspberry Pi Zero 2 W liegen in anderen Bereichen.

Ein erfreuliches und notwendiges Update

Der Raspberry Pi Zero 2 W ist ein erfreuliches Update der Zero-Familie. Mit der Wahl des SiPs des Raspberry Pi 3 steht eine bewährte Plattform mit guter Softwareunterstützung auf kleinen Bauraum parat. Ein Raspberry Pi 3 SiP in einem Zero-Formfaktor eröffnet neue Möglichkeiten, sei es in der Automatisierung, beim Retrogaming oder als kleiner energiesparender Mediaplayer. Auch für andere Projekte erhält man eine günstige Basis, wenn ein Raspberry Pi 4 zu teuer und zu energiehungrig ist.

210536-02

Ein Beitrag von

Projekt und Text: **Mathias Claußen**
Redaktion: **Jens Nickel**
Layout: **Giel Dols**

Sie haben Fragen oder Kommentare?

Haben Sie technische Fragen oder Anmerkungen zu diesem Artikel? Schicken Sie eine E-Mail an den Autor mathias.claussen@elektor.com oder kontaktieren Sie Elektor unter editor@elektor.com.

PASSENDE PRODUKTE

- **IQaudIO Codec Zero – Sound Card für Raspberry Pi Zero (SKU 19541)**
  www.elektor.de/19541
- **Retroflag GPi – Game Boy Gehäuse für Raspberry Pi Zero (W) (SKU 19273)**
  www.elektor.de/19273
- **ZeroDock – Prototyping Dock für Raspberry Pi Zero (SKU 19760)**
  www.elektor.de/19760

Projektidee: WLAN-Kamera mit Bewegungserkennung

Raspberry Pi Zero Boards wurden schon immer als WLAN-Kamera eingesetzt. Das funktionierte, aber vor allem in Verbindung mit Bewegungserkennung eher mäßig. Die Bootzeit war lang, und bei der Bewegungserkennung kam die CPU schnell an ihre Grenzen. Auch ein Livestream von der Kamera war nicht immer eine erfreuliche Angelegenheit, die Rechenleistung war dann doch überschaubar und die Bitrate des Streams leider auch. An vielen Stellen haben dann ältere und klobigere Raspberry-Pi-2-basierte Systeme die Aufgabe übernommen.

Mit dem Raspberry Pi Zero 2 W gibt es nun wieder eine gute Basis für solche WLAN-basierten Webcams, die genug Rechenleistung bieten, um eine Webcam mit Bewegungserkennung aufzubauen und einen passenden Livestream zu generieren. Durch die Größe ist ein kompakter Aufbau möglich.

Projektidee: USB-Druck und Scanserver

Mit dem USB-Port des Raspberry Pi lässt sich auch ein älterer Drucker wieder fit für das Netzwerk machen, und das dann sogar kabellos. Je nach Hersteller und Modell kann es ein wenig Konfigurationsarbeit erfordern, doch der Drucker kann im eigenen Netzwerk mit reduzierten Sicherheitslücken weiterbetrieben werden. Das gilt sogar für Scanner und Drucker, die unter aktuellen Windows- oder MacOS-Versionen nicht mehr einsetzbar sind [2].

CUPS und SANE sind hier die klassischen Vertreter für den Druck und das Scannen unter Linux. In Verbindung mit Samba lässt sich eine Drucker-Freigabe einrichten. SANEWinDS ermöglicht es Windows-Clients, auf einen Scanner über das Netzwerk zuzugreifen, so das mehrere Rechner sich den Scanner teilen können.

Auch der Raspberry Pi Zero 2 W selbst kann als Scanner eingesetzt werden - mit den GPIOs und ein paar Skripts lassen sich so auch komplexere Dokumentenverarbeitungen realisieren.

Projektidee: Mediaplayer

Der Raspberry Pi Zero 2 W eignet sich bestens als kleiner Mediaplayer. Vor allem in Verbindung mit dem IQaudio Codec Zero wird aus ihm ein kleines Streaming-Gerät für Audio. Distributionen wie Volumio lassen sich einfach nutzen. Alles wird bestens unterstützt, da es sich im Kern um einen Raspberry Pi 3 handelt. Auch lässt sich der MPD (Music Playing Deamon) auf dem Raspberry Pi Zero 2 W installieren und mit einem der diversen Frontends auf verschiedenen Geräten bedienen. Der Einsatz als WLAN-Stick mit Kodi stellt ebenfalls kein Problem dar.

Projektidee: Debugger für RP2040 und andere MCUs

Etwas, das die Dokumentation der Raspberry Pi Fundation etwas stiefmütterlich behandelt, ist die Option, einen Raspberry Pi als netzwerkfähigen Debugger einzusetzen. Während der Arbeit an den Artikeln zum Release des Raspberry Pi Pico haben wir schon erste Experimente mit einem Raspberry Pi Zero als WLAN-Debugger durchgeführt. Die Einrichtung erfordert zwar etwas Handarbeit und Anpassen der Konfiguration, das Ergebnis ist aber ein günstiger WLAN-fähiger Debugger, der nicht nur auf den Raspberry Pi Pico beschränkt ist, sondern eine breite Palette an Cortex-M-MCUs debuggen kann.

WEBLINKS

[1] 10 Jahre Raspberry Pi, Interview mit Eben Upton, Elektor Industry 03/2021: http://www.elektormagazine.de/210464-02

[2] Web-UI für Scanner im Netzwerk: https://github.com/sbs20/scanservjs

[3] Elektor TV: Der Raspberry Pi Zero 2 W

Impressionen vom World Ethical Electronics Forum 2021

Von **Priscilla Haring-Kuipers** (Niederlande)

Das World Ethical Electronics Forum 2021, das am 18. November 2021 online und live auf der productronica stattfand, inspirierte Innovatoren der Elektronikbranche auf der ganzen Welt mit einer offenen Diskussion über Ethik und Ziele für eine nachhaltige Entwicklung (Sustainable Development Goals, SDG).

Am 18. November 2021 kamen fast 200 Personen virtuell und physisch zusammen, um am ersten World Ethical Electronics Forum (WEEF) während der productronica 2021 in München teilzunehmen, das von *Elektor* und *Elektronik Praxis* organisiert wurde. Die Redner und Teilnehmer dieses Industrie-orientierten Treffens diskutierten über Mittel und Wege, die Elektronik ethischer zu gestalten. In den letzten zehn Jahren haben sich immer mehr Menschen Gedanken über ethische Fragen in der Elektronik gemacht, die verschiedenen Facetten unserer Branche erörtert und ethische Handlungsweisen in ihren eigenen Unternehmen vorangetrieben. Jetzt scheint die Zeit reif zu sein, diese Gedanken und Praktiken mit der gesamten Branche zu teilen und voneinander zu lernen.

Die Redner auf dem WEEF 2021

Werfen wir einen kurzen Blick auf die Themen, die einige der Redner auf dem WEEF 2021 behandelt haben. Sie können alle WEEF-Vorträge online bei Elektor TV [2] ansehen.

- Professor Dr. Stefan Heinemann eröffnete das WEEF mit seiner Einführung „A Focus on SDGs in Electronics“. In diesem ausgezeichneten Vortrag ermutigte er die Elektronikunternehmen, „mehr zu tun“, als es die aktuelle Gesetzgebung verlangt. Er erklärte, dass moralisches Handeln auch ein gutes Geschäft sei und dass man den ewigen Neinsagern keinen Glauben schenken sollte.
- Dr. Paula Palade sprach über ein Regelwerk für die Entwicklung und den Einsatz von selbstfahrenden Autos. Sie arbeitet an einem europäischen Projekt, um die ersten Richtlinien zu entwerfen.
- Amir Sherman (Edge Impulse) zeigte Möglichkeiten auf, maschinelles Lernen in kleiner Hardware einzusetzen, um reale Messwerte mit fortschrittlicher Modellierung zu kombinieren.
- Margot Cooijmans von der Philips-Stiftung erörterte, wie Medizintechnik in unterversorgten Regionen bereitgestellt werden kann. Dies sollte aber nicht durch die Überlassung gespendeter Technik ohne dauerhafte Auswirkungen auf die Gesundheitssysteme geschehen.
- Dirk Akerman von SEGGER sprach über einen Ansatz des „friendly licensing“, bei dem Produkte für private und Bildungszwecke sowie zur Erstellung von Prototypen kostenlos genutzt werden können.
- Felix Plitzko von AISLER strebt danach, „Hardware weniger hart zu machen“ und versprach, in Zukunft eine kohlenstoffneutrale Leiterplattenproduktion zu ermöglichen. Er erforscht dies derzeit zusammen mit dem Fraunhofer-Institut [3].
- Johannes und Markus Brenner sprachen darüber, wie man „Achtsamkeit“ zum zentralen Thema seines Unternehmens machen kann, anstatt nur dem Profit hinterherzujagen, und wie man sich gut um die Menschen in seinem Unternehmen kümmern kann.

Mehr WEEF

Natürlich dürfen wir uns nicht mit Reden zufrieden geben. Wenn wir in der Elektronik ethisch sein wollen, müssen wir handeln! Dies war erst die erste Ausgabe des WEEF, und es soll eine jährliche Veranstaltung werden. Das zweite WEEF ist bereits für November 2022 geplant. Bis dahin werden wir an einem WEEF-Index mit den „ethischsten" Personen/Unternehmen in der Elektronikbranche sowie an einem WEEF-Manifest arbeiten. Wir würden uns freuen, wenn Sie uns Ihre Meinung zu den folgenden und weiteren Fragen mitteilen würden:

- Wen möchten Sie in den WEEF-Index aufnehmen?
- Was sollte in das WEEF-Manifest aufgenommen werden?
- Welches sind die wichtigsten ethischen Fragen in Ihrer Elektronik-Praxis?

Bitte verfolgen Sie die Diskussion und beteiligen Sie sich unter **www.elektormagazine.com/weef**.

210645-02

WEBLINKS

[1] WEEF-Webseite: https://www.elektormagazine.com/weef
[2] ElektorTV, „World Ethical Electronics Forum 2021": https://youtu.be/ELpxR6SuLb8
[3] AISLER, „AISLER Plans to Ship the First World-Wide Carbon Neutral Circuit Board in 2022", 18. November 2021: https://www.elektormagazine.com/news/aisler-plans-to-ship-first-world-wide-carbon-neutral-circuit-board-in-2022

MOTORsteuerung

Wie die Motorsteuerung weniger kompliziert wird

Motor-Steuerung

Von **Stuart Cording** (Elektor)

Der Elektromotor erfindet sich ständig neu, um neue Anwendungsanforderungen zu erfüllen - von elektronischem Spielzeug und Haushaltsgeräten bis hin zu Elektrowerkzeugen und Elektrofahrzeugen. Ebenso wichtig sind jedoch die vielfältigen elektronischen Steuerungen, die mit dem Motor Hand in Hand gehen, um die verschiedenen Anforderungen zu erfüllen. Die Integration ist der Schlüssel dazu, die Leistungsfähigkeit eines Motors voll auszuschöpfen. In den 1970er Jahren revolutionierte die integrierte H-Brücke die Heimelektronik, zum Beispiel beim Videorekorder. Heute sorgen hoch optimierte Mikrocontroller mit speziellen Motorsteuerungsfunktionen für einen effizienten Betrieb bürstenloser Motoren.

Kleine bürstenbehaftete Permanentmagnet-Gleichstrommotoren sind fast überall zu finden. Die meisten dieser Motoren, die oft als **F**ractional **H**orse**P**ower (FHP) bezeichnet werden, dienen der Bewegungssteuerung in Automobilen und Haushaltsgeräten. Dank des zunehmenden Komforts werden in Autos nicht nur Scheibenwischer und Lüftungsgebläse, sondern auch Spiegel, Fensteröffner, Schiebedächer, Türen und verstellbare Sitze elektromotorisch gesteuert. Die dabei verwendeten Gleichstrommotoren werden aus elektrischer Sicht auf einfache Weise, aber wenig effizient angesteuert. Deshalb werden solche Motoren nach wie vor häufig eingesetzt, wenn es sich nicht um Anwendungen handelt, die einen Dauerbetrieb erfordern, wie bei Pumpen und Lüftern. Hier kommen meist effizientere bürstenlose Motoren zum Einsatz.

Steuerung einfacher bürstenbehafteter Gleichstrommotoren

Bei einfachen Gleichstrommotoren mit Permanentmagneten werden Rotor und Stator durch ein Spaltrohr getrennt, in dem sich normalerweise zwei Magnete befinden. In der Mitte ist ein Anker auf einer Rotorwelle montiert, auf der sich die Wicklungen befinden, die wiederum mit einem Kommutator auf der Rotorwelle verbunden sind. Kohlebürsten, die in der Regel Teil der Kappe auf dem Spaltrohr des Stators sind, ermöglichen die Zuführung elektrischer Energie zu den Wicklungen (**Bild 1**). Die Position des Kommutators auf der Welle ist entscheidend für einen optimalen Wirkungsgrad. Im Idealfall muss die Position der Kommutatorbürste, die so genannte Kommutierungsebene, dafür sorgen, dass das von den Rotorwicklungen erzeugte Magnetfeld im rechten Winkel zum Magnetfeld des Stators steht. Während des Betriebs „verläuft" diese Ebene jedoch aufgrund von Störungen im Magnetfeld des Stators. Der Einfachheit halber wird deshalb eine Kommutierungsebene gewählt, die bei einer bestimmten Drehzahl den höchsten Wirkungsgrad erzielt.

Da die Kommutierung des Motors mechanisch gelöst ist, müssen nur noch Drehzahl und Drehrichtung elektrisch umgesetzt werden. Ist nur eine Drehzahlanpassung erforderlich, genügt ein pulsweitenmoduliertes Signal (PWM), das ein entsprechend dimensioniertes Leistungsbauteil wie einen MOSFET ansteuert (**Bild 2**). Die meisten Mikrocontroller (MCU) verfügen über einen Timer, der, einmal konfiguriert, ein PWM-Signal ohne regelmäßige Interaktion seitens des Prozessors erzeugen kann. In solchen Fällen wird ein n-Kanal-MOSFET als Low-Side-Schalter bevorzugt, da das Gate in der Regel bei Anwendungen mit geringem Strombedarf direkt angesteuert werden kann. Wird doch ein High-Side-Schalter benötigt, muss ein n-Kanal-MOSFET zusammen mit einem MOSFET-Treiber eingesetzt

werden, um die erforderliche Gate-Spannung zu erzeugen [1].

Drehzahlüberwachung und Richtungssteuerung

Für die Drehzahlregelung benötigt der verwendete Regelkreis die aktuelle Drehzahl der Motorwelle. Ein Ansatz ist die Verwendung eines Sensors, zum Beispiel eines optischen Detektors, und ein am Rotor befestigtes Schlitzrad. In einem digitalen Steuersystem kann die Periode zwischen den Signalen des optischen Sensors in Motorumdrehungen pro Minute (RPM) übersetzt werden. Die meisten MCUs verfügen über einen Eingangsblock, der Zeitgeberwerte anhand der steigenden und fallenden Flanke eines Signals erfassen kann. Die Auflösung hängt von der Taktrate des Timers ab und sollte hoch genug sein, um auch bei höchsten Drehzahlen brauchbare Daten zu liefern. Der Ausgang dieses Blocks wird in einen PID-Regelalgorithmus (Proportion-Integral-Differential) [2] eingespeist, der das Puls/Pausen-Verhältnis der PWM ändert, um den Sollwert der Motordrehzahl zu erhalten (**Bild 3**).

Solche auf dem Rotor montierten Sensoren erweisen sich aber oft als zu teuer für die geplante Anwendung. Stattdessen kann die Gegen-EMK des Motors, die proportional zu seiner Drehzahl ist, verwendet werden. Sie bietet bei den meisten Anwendungen mehr als genug Genauigkeit. Während der Phasen der PWM-Steuerung, in denen der Leistungsschalter ausgeschaltet ist, kann die Gegen-EMK direkt an den Klemmen des Motors gemessen werden. Da die meisten Mikrocontroller auch über einen Analog-Digital-Wandler (ADC) verfügen, ist dies sehr wirtschaftlich. Es sind in der Hardware nur ein paar Widerstände zur Begrenzung der Spannung auf den Eingangsbereich des ADC erforderlich. Komplexer ist die Software-Implementierung. Die ADC-Messung muss mit dem Abschalten des MOSFETs synchronisiert werden und erfordert eine zusätzliche kurze Verzögerung, damit sich die Spannung einpendeln kann (**Bild 4**). Bei MCUs, die auf Motorsteuerung spezialisiert sind, kann wahrscheinlich eine Hardwareverbindung zwischen dem PWM-Ausgang und dem ADC-Trigger sowie ein Verzögerungsblock eingerichtet werden, so dass dieser Messansatz automatisiert werden kann.

Bild 1. Die Bestandteile eines Gleichstrommotors mit kleiner Leistung. Der Kommutator ist deutlich auf der Rotorwelle neben den Ankerspulen zu sehen. (Quelle: ShutterStock/Pixel Enforcer)

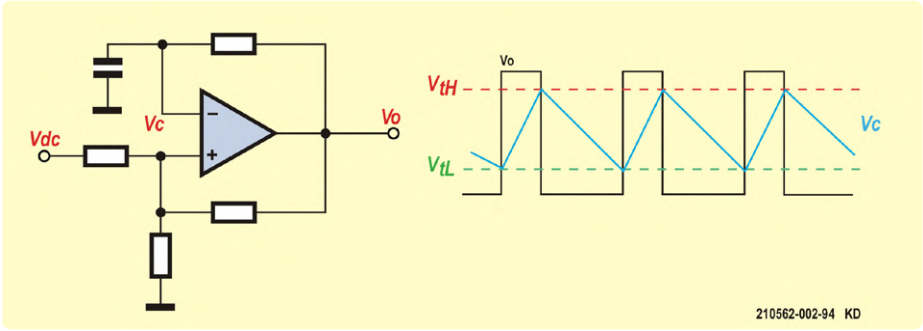


Bild 2. PWM-Generator auf Grundlage eines Operationsverstärkers.

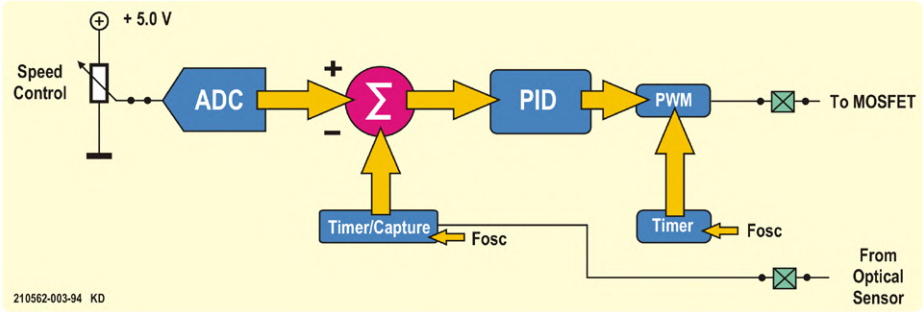


Bild 3. Das Ausgangssignal eines am Rotor des Motors angebrachten Sensors kann zur Erfassung von Zeitgeberwerten verwendet werden. Diese werden mit dem Sollwert der Drehzahl kombiniert und an den PID-Regler weitergeleitet, um die Motordrehzahl zu regeln. (Quelle: Microchip Technology)

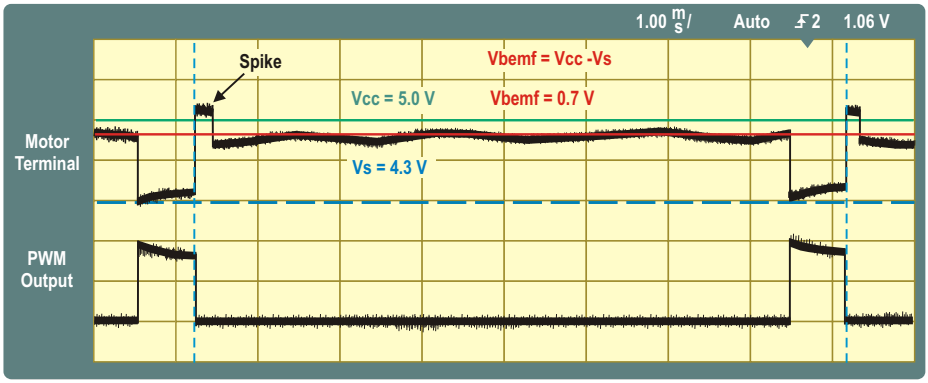
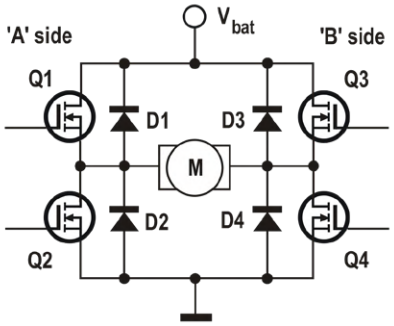


Bild 4. Nach dem Abschalten des MOSFET erscheint eine kleine Spitze an den Motorklemmen. Die ADC-Messungen müssen warten, bis die Gegen-EMK einen stabilen Zustand erreicht hat. (Quelle: Precision Microdrives)

Damit haben wir schon die Zutaten für eine Steuerung der Motordrehzahl; es muss nur noch die Drehrichtung geregelt werden. Die H-Brücke (**Bild 5**) verwendet vier Leistungsschalter, mit denen sich die Polarität der an den Motorklemmen anliegenden Versorgung vertauschen lässt (entweder Q2 und Q3 oder Q1 und Q4). Das Kurzschließen der Motorklemmen über das obere oder untere Schalterpaar kann als elektrische Bremse zur Verringerung der Rotordrehzahl verwendet werden. Die Drehzahlregelung erfolgt wiederum über ein PWM-Signal für einen oder beide MOSFETs. Es ist jedoch zu beachten, dass es zu einem Kurzschluss kommen kann, wenn sowohl High- als auch Low-Side-MOSFETs gleichzeitig eingeschaltet sind.

Die meisten großen Halbleiterhersteller bieten hoch integrierte H-Brückenlösungen an, die einfach zu verwenden sind und eine Vielzahl von Schutz- und Diagnosefunktionen bieten. Der IFX9201SG [3] von Infineon ist ein solcher Baustein, der für Ströme von bis zu 6 A für Motorsteuerungen in industriellen Anwendungen geeignet ist. Der Baustein kann von Allzweck-I/Os und einem PWM-Signal angesteuert werden und ermöglicht so die aktive Kontrolle von Motordrehzahl und -richtung. Parallel dazu ermöglicht eine SPI-Schnittstelle den Zugriff auf die Diagnoseregister, die Rückmeldungen über Kurzschlüsse zur Versorgungsschiene oder Masse, offene Lastbedingungen oder Unterspannung enthalten. Falls gewünscht, können Drehzahl und Richtung statt über Richtungs- und PWM-Eingangspins auch über SPI gesteuert werden. Das Risiko eines unbeabsichtigten Kurzschlusses der H-Brücke ist damit ausgeschlossen, und weitere Schutzfunktionen wie Übertemperatur und Überstrom sind integriert. Wer schnell einen Prototyp entwickeln möchte, kann das H-Bridge-Kit 2Go [4] erwerben, das sich in die Software-Entwicklungsumgebung DAVE von Infineon integrieren lässt.



	Q1 (CTRL1)	Q2 (CTRL2)	Q3 (CTRL3)	Q4 (CTRL4)
Forward	on	off	off	on
Reverse	off	on	on	off
Coast	off	off	off	off
Break	off	on	off	on

210562-005-94 KD

Bild 5. Eine H-Brückenschaltung mit einer Tabelle, die beschreibt, wie der Schalterbetrieb die Richtung steuert. Durch Kurzschließen der Motorklemmen lässt sich auch eine elektrische Bremse realisieren. (Quelle: Microchip Technology)

Optimierung des PID-Reglers

Die Suche nach einer MCU, die dem gewünschten Preisniveau für die Anwendung entspricht, führt oft zu einem 8-Bit-Controller. Leider sind solche MCUs nicht gerade für ihre mathematische Leistung berühmt, was es schwierig macht, einen PID-Regler zu implementieren, der prompt auf Änderungen des Drehzahlsollwerts oder Lastschwankungen reagiert. Um dieses Problem zu lösen, verfügen einige PIC-MCUs wie der PIC16F1619 von Microchip über einen integrierten PID-Accelerator-Block (**Bild 6**) mit einem 35-Bit-Akkumulator. Mit der üblichen PID-Regelgleichung werden drei Konstanten aus Verstärkung K_p, Nachstellzeit T_n und Vorhaltzeit T_v in die Register des Moduls geschrieben, bevor der gewünschte Sollwert eingegeben und das PID-Berechnungsmodul aktiviert wird. Das PID-Modul reduziert die Anzahl der erforderlichen Taktzyklen von etwa 1.000 bei einem Software-PID auf der gleichen MCU auf neun [5].

Die Vorteile von BLDC und PMSM

Obwohl FHP-Gleichstrommotoren attraktiv einfach sind, rücken zunehmend auch andere Eigenschaften wie

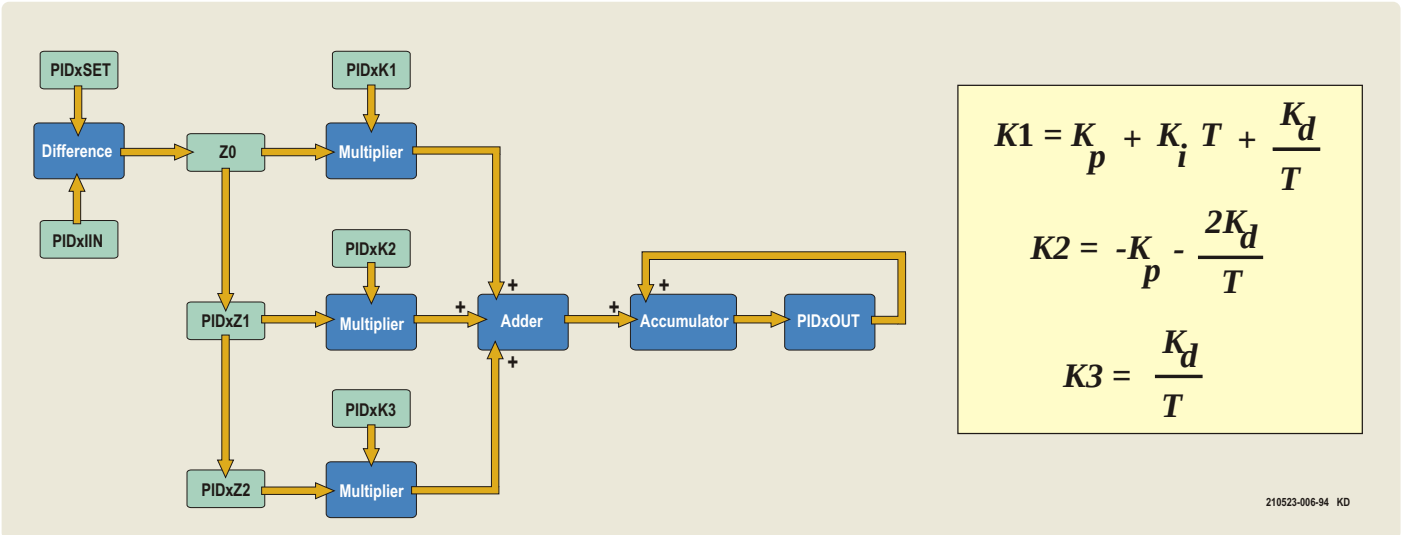


Bild 6. Der PID-Block des PIC16F1619 zusammen mit den Gleichungen zur Berechnung der drei verwendeten Standardkonstanten. (Quelle: Microchip Technology)

Effizienz, Geräuschentwicklung und höhere Leistungsdichte bürstenlose Gleichstrommotoren (BLDC) und Permanentmagnet-Synchronmotoren (PMSM) in den Blickpunkt. Bei optimaler Ansteuerung sind BLDC- und PMSM-Motoren wesentlich effizienter, und da sich die elektrischen Wicklungen im Stator befinden, ist die Wärmeabfuhr einfacher.

BLDC- und PMSM-Motoren können als von innen nach außen umgewendete FHP-Gleichstrommotoren betrachtet werden. Die für die Kommutierung verwendeten Spulen sind auf den Stator, also das Gehäuse des Motors, gewickelt, während der Magnet um den Rotor gewickelt ist. Dies führt zu leichten Rotoren mit geringer Trägheit. Allerdings gibt es keine Mechanik für die Umsetzung der Kommutierung, so dass diese Aufgabe dem Entwicklungsingenieur überlassen bleibt. Der Hauptunterschied zwischen den beiden Motortypen besteht darin, dass BLDC-Motoren eine trapezförmige Gegen-EMK erzeugen und einfachere Kommutierungsanforderungen aufweisen als PMSM-Motoren, die eine sinusförmige Kommutierung verwenden und eine sinusförmige Gegen-EMK erzeugen. PMSM-Motoren sind auch leiser im Betrieb.

Die Wicklungen sind in der Regel sternförmig angeordnet. Es erfordert sechs Halbbrücken, um die Spulen eines BLDC-Motors in einem sechsstufigen Muster zu erregen, das zwei der drei Spulen mit Strom versorgt (**Bild 7**). Es entsteht ein Magnetfeld, das sich um den Stator dreht, ein Feld, dem der Rotor einfach folgen muss und das zu der gewünschten mechanischen Drehung führt. Die Umkehrung der Kommutierungsreihenfolge bewirkt, dass der Rotor seine Richtung ändert. Während das abgebildete einfache Modell der BLDC-Funktionalität den Betrieb verdeutlicht, sind die Motoren in der Realität in der Regel allerdings so gewickelt, dass mehrere elektrische Umdrehungen erforderlich sind, bevor eine vollständige mechanische Drehung erreicht wird.

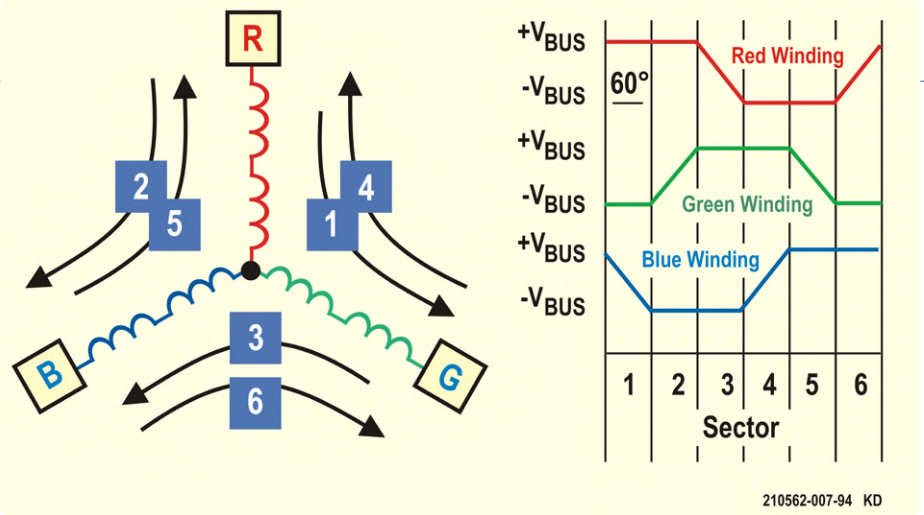


Bild 7. BLDC-Motor mit sternförmigen Wicklungen und den Kommutierungsschritten für eine einzige elektrische Umdrehung. Die mechanische Drehung kann mehrere elektrische Umdrehungen erfordern. (Quelle: Microchip Technology)

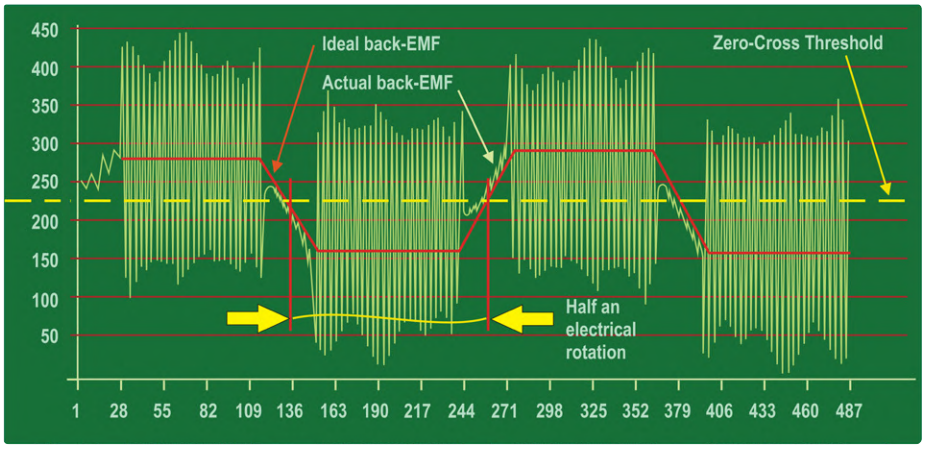


Bild 8. Die Gegen-EMK der offenen Phase kann gemessen und zur Bestimmung des Drehwinkels verwendet werden, wodurch ein Hall- oder anderer Sensor überflüssig wird. (Quelle: Microchip Technology)

Bestimmung des Rotorwinkels ohne Sensor

Die nächste Aufgabe besteht darin, die Position des Rotors zu bestimmen, damit der nächste Kommutierungspunkt zum richtigen Zeitpunkt ausgegeben wird. In vielen BLDC-Motoren sind Hallsensoren eingebaut, die den Rotorwinkel mit einer angemessenen Genauigkeit anzeigen. Die steigenden und fallenden Flanken solcher Sensoren lassen sich bei vielen MCUs mit flanken-getriggerten Interrupts verknüpfen, so dass sich die korrekte Ansteuerung der Motorspulen einfach in Interrupt-Routinen realisieren lässt. Solche Signale können auch von einem Timer/Counter erfasst werden, um die Drehzahl zu bestimmen und den Eingang eines PID-Reglers zu speisen. Die Motordrehzahl wird auch über PWM-Ausgänge gesteuert. Auch wenn die Kommutierung solcher Motoren in der Verantwortung des Entwicklungsingenieurs liegt, unterscheidet sich der Ansatz nicht so sehr von der Verwendung ihrer Gleichstrom-Pendants mit Bürsten.

Wie bereits erwähnt, erhöhen aber die Sensoren die Kosten und es kann auch sein, dass einfach kein Platz für sie vorhanden ist. Wie bürstenbehaftete Gleichstrommotoren erzeugen aber auch BLDC-Motoren eine Gegen-EMK, die zur Bestimmung der Rotorposition verwendet werden kann. Wenn die Gegen-EMK der nicht angetriebenen Phase die Nulldurchgangsschwelle (die Hälfte der Versorgungsspannung) durchquert, kann der Rotorwinkel durch Messung der Zeit zwischen diesen Nulldurchgangsereignissen bestimmt werden (**Bild 8**).

Das erscheint auf den ersten Blick einfach, ist aber praktisch äußerst schwierig umzusetzen. Zunächst einmal erzeugen Motoren im Stillstand keine Gegen-EMK,

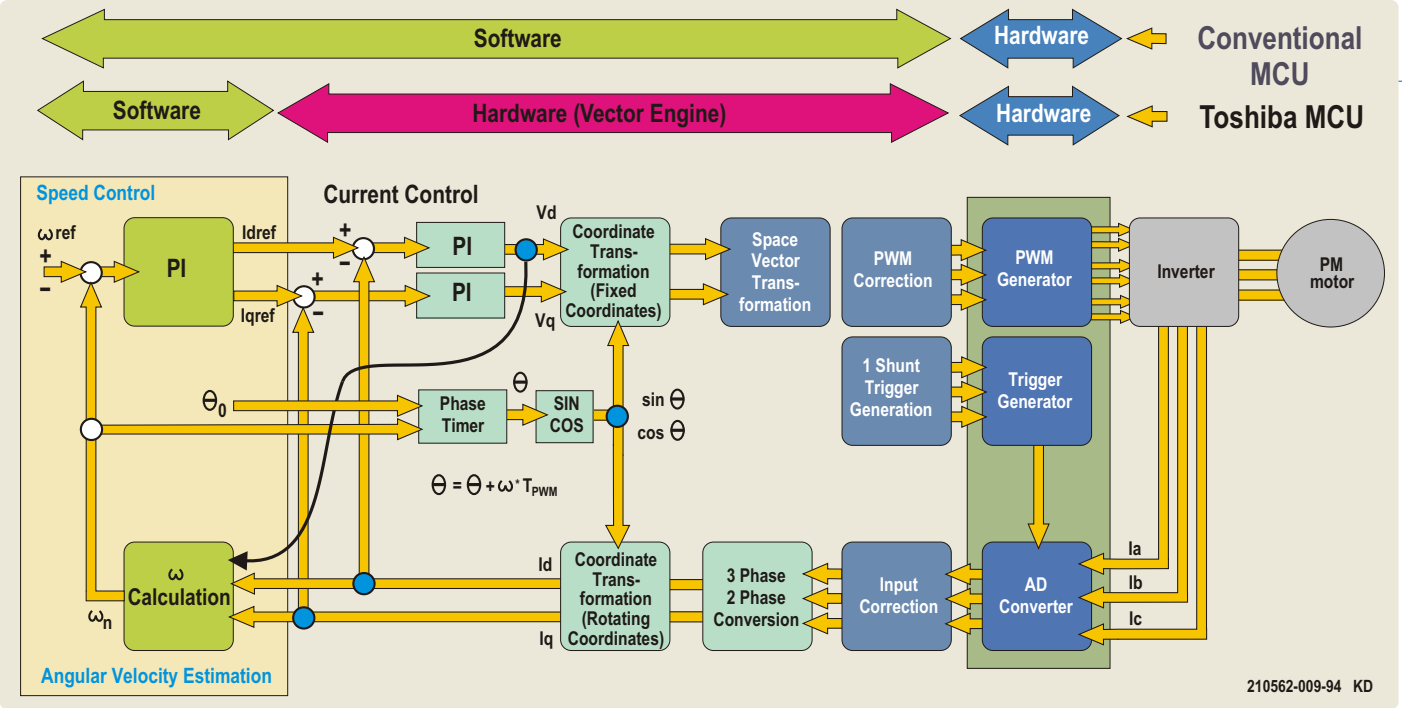


Bild 9. Hoch integrierte Motorsteuerungsmodule wie die Advanced Vector Engine Plus von Toshiba ermöglichen eine quasi-autonome Steuerung von Permanentmagnetmotoren. (Quelle: Toshiba Electronics Europe)

so dass sie zunächst auf (ausreichende) Touren gebracht werden müssen, bei der ein Algorithmus genügend Gegen-EMK-Signal zur Messung erhält. Außerdem kann der Code, der für die Auswertung der Gegen-EMF-Daten und die Anwendung der Filterung erforderlich ist, eine erhebliche Ausführungszeit aufweisen. Dies kann bedeuten, dass zwischen verschiedenen Algorithmen und Kommutierungsansätzen gewechselt werden muss, wenn der Motor zwischen niedrigeren und höheren Drehzahlen wechselt. Anbieter von intelligenten BLDC-Motorsteuerungslösungen haben viele dieser Probleme gelöst: Ein neuer Motor muss nur mehr an eine Motorsteuerung angeschlossen werden, die automatisch den Motor „kennenlernt". Mehr hat der Ingenieur nicht zu tun!

Implementierung einer sinusförmigen Motorsteuerung

Die Algorithmen zur Steuerung von PMSM-Motoren weisen eine zusätzliche Komplexitätsebene auf. Die PWMs der MCU müssen nämlich kontinuierlich moduliert werden, um ein sinusförmiges Ausgangssignal zu erzeugen, das die Motorwicklungen erregt. Solche Ansätze verwenden eine feldorientierte Regelung (FOC), bei der die Ströme der Dreiphasenwicklungen zur Bestimmung des aktuellen Rotorwinkels verwendet werden. Mit Hilfe von Clarke- und Park-Transformationen sorgt diese Regelungsmethode dafür, dass jederzeit ein Winkel von 90° zwischen dem Rotor- und dem Statormagnetfeld aufrechterhalten und so ein maximales Drehmoment erreicht wird.

Führende Unternehmen im Bereich der Motorsteuerung konzentrieren sich die auf die Vereinfachung ihrer Lösungen, damit sich die Entwickler auf die Alleinstellungsmerkmale ihrer Anwendungen konzentrieren können.

Zwar lässt sich eine solch komplexe Software auch in einigen Standard-MCUs implementieren, viele Hersteller haben aber spezielle Hardwaremodule entwickelt, die einen Großteil der Komplexität automatisieren. Eine solche Familie von Bausteinen ist die TXZ4A+-Reihe von Toshiba [6]. Diese Arm-Cortex-M4-MCUs verfügen über ein *Advanced Vector Engine Plus (A-VE+)*-Modul, das eng mit den PWM- und ADC-Modulen gekoppelt ist (**Bild 9**). Innerhalb der Hardware des A-VE+-Moduls werden die kompletten Dreiphasen-zu-Zweiphasen- und inversen Clarke- und Park-Transformationen durchgeführt, so dass neben einigen „Verwaltungsaufgaben" in der Software nur ein PI-Regler und eine Abschätzung der Winkelgeschwindigkeit erforderlich sind. Die restliche Zeit arbeiten die Module quasi-autonom zusammen.

Innovation bei der Motorsteuerung

Angesichts der hohen Komplexität konzentrieren sich die führenden Unternehmen im Bereich der Motorsteuerung auf die Vereinfachung ihrer Lösungen, damit sich die Entwickler auf die besonderen Funktionen ihrer Anwendungen konzentrieren können. Nach Ansicht von Jonas P. Proeger, *Director Business Management* bei Trinamic (jetzt Analog Devices) (**Bild 10**), sind die Kernthemen die Verbesserung der Effizienz, die Verringerung der Geräusche und die Erhöhung der Intelligenz. Effizienzverbesserungen sind notwendig, um die vor allem in der Logistik wachsende Zahl

Bild 10. Jonas P. Proeger, Director Business Management bei Trinamic (jetzt Analog Devices). (Quelle: Analog Devices)

batteriebetriebener autonomer Fahrzeuge wie *Autonomous Mobile Robots* (AMR) zu unterstützen. Auch die verwendeten Motoren ändern sich: Höhere Leistungsdichten und Gewichtseinsparungen führen zu Motoren mit geringerer Induktivität. Dies hat zur Folge, dass sich die Algorithmen zur Motorsteuerung ändern müssen, wobei die Regelkreise mit bis zu 100 kHz arbeiten. Bausteine wie der TMC4671 [7] erreichen dies durch den Einsatz dedizierter Hardware und verdrängen damit alternative Ansätze mit DSPs oder FPGAs.

Ein weiteres wachsendes Bedürfnis ist der geräuscharme Betrieb, der im Vergleich zu alternativen Produkten mit ihren Schleifmotoren und Getrieben den Eindruck höherer Qualität vermittelt. Hier hebt Jonas seine StealthChop-Technologie hervor, die eine spannungsgesteuerte Chopping-Technik einsetzt, um hörbare Geräusche bei einem bipolaren Schrittmotor um bis zu 10 dB zu reduzieren [8]. Die daraus resultierende Reduzierung des Ripple-Stroms senkt zudem die Verlustleistung des Motors. Der letzte Wachstumsbereich sind intelligente, „selbstlernende" Motoren für Industrieanlagen. Solche Motorlösungen tragen zum zunehmenden Einsatz von künstlicher Intelligenz auf Systemebene bei, indem sie Echtzeitdaten direkt aus der Fabrikhalle liefern.

Leistung ist nichts ohne Kontrolle

Man kann leicht den Eindruck gewinnen, dass sich alles auf BLDC- und PMSM-Motoren verlagert. Dies ist jedoch nicht der Fall. Es gibt nach wie vor zahlreiche Anwendungen, bei denen herkömmliche FHP-Gleichstrommotoren ihren Zweck erfüllen. Und angesichts der Fülle von H-Brücken-Motortreibern und Low-End-MCUs mit cleverer Peripherie ist klar, dass auch die Halbleiterindustrie noch viel Potenzial sieht. Da die Anwendungen jedoch immer höhere Wirkungsgrade verlangen, auf Batteriebetrieb umstellen oder einen nahezu geräuschlosen Betrieb erfordern, müssen sich die Ingenieure mit BLDC- und PMSM-Motoren auseinandersetzen. Zwar gelten viele der für die Steuerung von Gleichstrommotoren verwendeten Techniken nach wie vor, doch gibt es bei den Feinheiten der Reglerimplementierung zahlreiche Fallstricke. Eine solide theoretische Grundlage ist deshalb hilfreich und ermöglicht die Fehlersuche bei Motorsteuerungslösungen, um jedoch Zeit zu sparen, ist es sinnvoll, auf die Erfahrung von Experten in diesem dynamischen Bereich zurückzugreifen und die hochintegrierten Hardwarelösungen zu prüfen, die der Markt zu bieten hat.

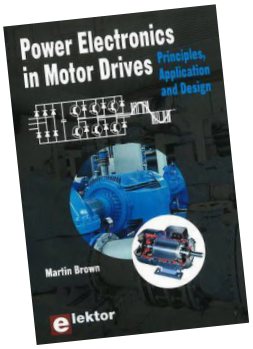
210562-02

Haben Sie Fragen oder Kommentare?

Haben Sie technische Fragen oder Kommentare zu diesem Artikel? Schreiben Sie dem Autor eine E-Mail an stuart.cording@elektor.com.

Ein Beitrag von

Gestaltung und Text: **Stuart Cording**
Redaktionelle Bearbeitung: **C. J. Abate**
Illustrationen: **Patrick Wielders**
Übersetzung: **Rolf Gerstendorf**
Layout: **Harmen Heida**



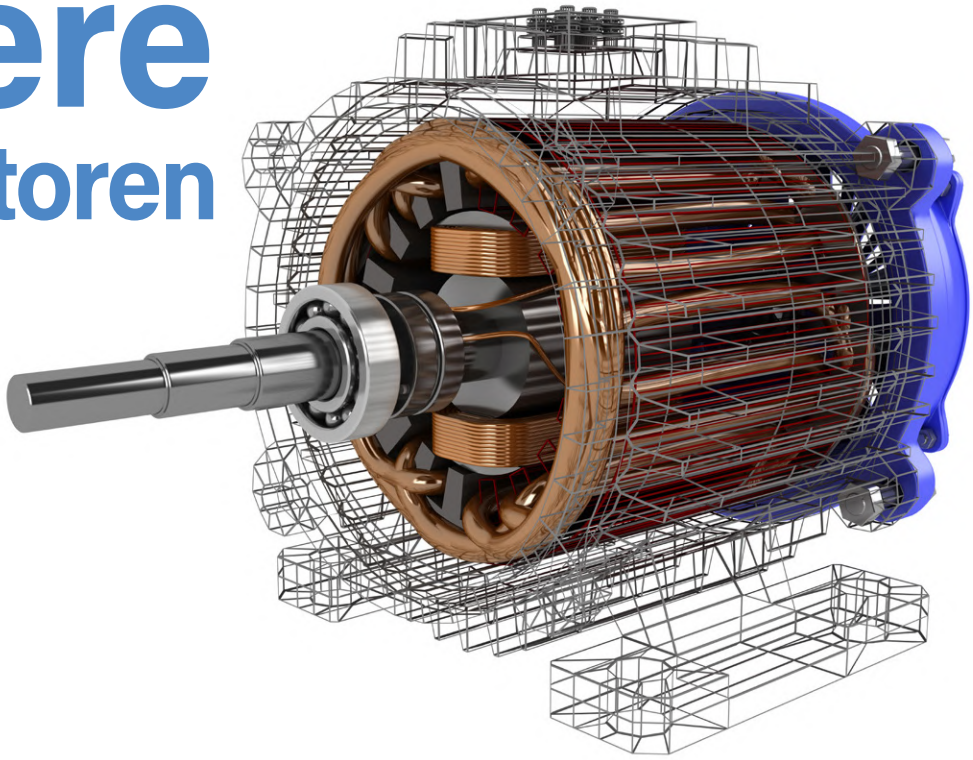
PASSENDE PRODUKTE

› E-book: Power Electronics in Motor Drives (SKU 18517)
www.elektor.de/18517

WEBLINKS

[1] C. Valens, „How to Choose Between High-side and Low-side Switching", Elektor, August 2019: https://bit.ly/3H38uA5
[2] „Understanding PID Control, Part 1: What Is PID Control?", MathWorks, Mai 2018: https://bit.ly/3CZfcor
[3] Infineon-Produktseite IFX9201SG: https://bit.ly/3CZfEmD
[4] Infineon-Produktseite H-Bridge-Kit 2GO: https://bit.ly/3ww91FZ
[5] D. Hou, „PID Control on PIC16F161X by using a PID Peripheral", Microchip Technology Inc. 2015: https://bit.ly/3DbGCHy
[6] Toshiba-MCUs der TXZ4A+-Reihe: https://bit.ly/3bXUSId
[7] Trinamic-Produktseite TMC4671: https://bit.ly/3bWO2CA
[8] B. Dwersteg, „stealthChop Performance (qualitative)", TRINAMIC Motion Control GmbH & Co. KG: https://bit.ly/3CZgxvt

Größere Elektromotoren



Prinzipien und Wissenswertes

Motor Control

Von **Dr. Thomas Scherer**

Elektromotoren haben unterschiedliche Charakteristiken und werden schon viele Jahrzehnte in den unterschiedlichsten Anwendungen eingesetzt. Nach dem nun abzusehenden und weitgehenden Ersatz von Verbrennermotoren in Fahrzeugen durch elektrische Antriebe werden sie wohl als einzige Maschinen für mechanische Bewegungen übrigbleiben. Grund genug, sich einen Überblick zu verschaffen.

Elektromotoren sind überall da zu finden, wo elektrische in irgendeine Art mechanische Energie umgewandelt werden soll. Sie haben viele Vorteile gegenüber Verbrennermotoren und anderen nichtelektrischen Verfahren der Erzeugung mechanischer Energie (zum Beispiel Wasser- und Windmühlen). Die wichtigsten sind kleiner Raumbedarf, geringes Gewicht, lange Lebensdauer und hohe Wirkungsgrade, die gerade bei größeren Exemplaren weit über 90 % liegen. Die kleinsten Exemplare findet man in nichtmechanischen Luxusuhren. Im Leistungsbereich unter 100 W gibt es zwar auch gewöhnliche, sogenannte Universalmotoren, doch dominieren viele Spezialvarianten wie etwa Schrittmotoren, Servos, Spaltpolmotoren, kleine Asynchronmotoren und Synchronmotoren sowie zunehmend BLDC-Motoren. Mathias Claußen hat in einem Beitrag [1] an einer anderen Stelle dieser Elektor-Ausgabe die Ansteuerung kleiner Motoren beleuchtet; von mir war vor drei Jahren ein Artikel [2] über generelle Ansteuermethoden zu lesen. Nachfolgend werden daher vor allem die Prinzipien größerer Motoren beschrieben, wie sie vielfach nicht nur bei Staubsaugern und Waschmaschinen, sondern auch in elektrisch angetriebenen Fahrzeugen und vor allem auch in der Industrie, beispielsweise in Werkzeugmaschinen zum Einsatz kommen.

Motortypen und Pi x Daumen

Wie schon erwähnt, geht es in diesem Beitrag nicht um Motoren kleiner Leistung, weshalb die in diesem Bereich häufig vorkommenden Typen (vom Spaltpol- über Kondensator- und Schrittmotor bis zu Servos) nicht behandelt werden. Ebenfalls nicht auf dem Radar sind Linearmotoren, obwohl sie häufig in Festplatten zur Kopfpositionierung und mit etwas höherer Leistung in Stellantrieben Verwendung finden. Auch Höchstleistungslinearmotoren im MW-Bereich sind ein anderes Thema, da deren militärische Nutzung etwa in Railguns aufgrund des Zwecks der Beschleunigung von Projektilen gänzlich andere Ansteuerungen erfordert. Zudem stellen Railguns ein eher exotisches Thema dar.
Nachfolgend wird daher nur auf die vier Typen von **Tabelle 1** eingegangen, die in der Praxis sicherlich mehr als 90 % aller Anwendungsfälle von unter 1 kW bis in den mehrstelligen MW-Bereich abdecken. Prinzipiell lässt sich jeder Motortyp sowohl in klein

Tabelle 1. Typen größerer elektrischer Motoren.	
Bezeichnung	**Typisches/Besonderheiten**
Universalmotor	Mechanischer Kommutator erzeugt Wechselfeld, zahlreiche Varianten, Reihen- und Nebenschluss, hohes Anlaufdrehmoment
Drehstrom-Asynchronmotor	Passiver Läufer (kein Kollektor), Rotor läuft Statorfeld nach, Steuerung über Schütze oder Frequenzumrichter (Drehzahl), hoher Anlaufstrom (Stern/Dreieck-Umschaltung)
Drehstrom-Synchronmotor	Erregerfeld notwendig (permanent/elektrisch), Rotor synchron zum Drehfeld, Frequenzumrichter (Drehzahl), auch Generator, Außen- / Innenpolmaschine
BLDC-Motor	Bürstenloser Gleichstrom-Motor = Variante des Synchronmotors mit elektronischer Kommutierung, für Antriebe, sehr zuverlässig, bei kleineren und mittleren Leistungen oft mit Permanentmagneten

als auch in ganz groß bauen, aufgrund unterschiedlicher elektrischer und mechanischer Eigenschaften eignen sie sich aber für bestimmte Anwendungsbereiche, wobei sich breite Überschneidungen ergeben.

Interessant sind bei Elektromotoren auch Mythen, die bei nicht auf Motoren spezialisierten Technikern nach wie vor verbreitet sind. So zum Beispiel die überidealisierte Vorstellung, dass Elektromotoren beim Start – also bei Drehzahl = 0 U/min – über ein „unendliches" Drehmoment verfügen würden. Der Konjunktiv ist berechtigt, denn das würde nur bei extremen Strömen, extremen Magnetfeldern, winzigsten Wicklungswiderständen, hyperstabilen Metallen und utopischer Steuerelektronik möglich sein. Da all diese Parameter aber irdischen Begrenzungen unterliegen, bleibt vom Mythos nur die Faustregel, dass (die meisten Typen von) Elektromotoren tatsächlich über ein hohes Anlaufdrehmoment verfügen, wenn sie selbst und ihre Ansteuerung sowie die Versorgung entsprechend ausgelegt sind. Gerade in Fahrzeugen hat diese Eigenschaft gegenüber Verbrenner-Motoren den Vorteil, dass man sich ein Schaltgetriebe mit Gängen (oder einer Automatik) weitgehend sparen kann, was der Zuverlässigkeit, der Einfachheit und dem geringen Gewicht des Antriebs zugute kommt – selbst wenn letzterer Aspekt aufgrund des hohen Gewichts von Akkupaketen weit mehr als aufgefressen wird.

Wer bei Motoren nicht so bewandert ist, für den ist wohl auch eine weitere Faustregel hilfreich: Bei Elektromotoren korrelieren Drehmoment und Betriebsstrom bei gegebener Last recht linear, denn der fließende Strom korreliert mit dem erzeugten Magnetfeldern der Wicklungen und diese mit den daraus resultierenden Anziehungs- oder Abstoßungskräften, welche ja das Drehmoment bewirken.

Noch eine, allerdings noch gröbere Faustregel wäre, dass die maximal mögliche Leerlaufdrehzahl – abgesehen von mechanischen Erwägungen – von der Versorgungsspannung abhängig ist. Warum? Sich drehende Elektromotoren erzeugen eine sogenannte Gegen-EMK, die abgesehen von konstruktiven Aspekten weitgehend von der Drehzahl abhängt. Unter dem historischen Begriff elektromotorische Kraft = EMK wird eine motorisch generierte Spannung verstanden. Bei der Gegen-EMK handelt es sich um die durch das Drehen induzierte Spannung, die der Versorgungsspannung entgegensteht. Sieht man von Induktivitäten und Verlusten ab, ist die resultierende und damit wirksame Spannung an den Wicklungen die Differenz aus angelegter Spannung und Gegen-EMK, und der resultierende Strom hängt fast ausschließlich von dieser Differenz und dem ohmschen Widerstand der Wicklungen ab. Geht die Differenz gegen 0 V, ist die Maximaldrehzahl erreicht, da der nun kleine noch fließende Strom lediglich die Reibungsverluste kompensiert.

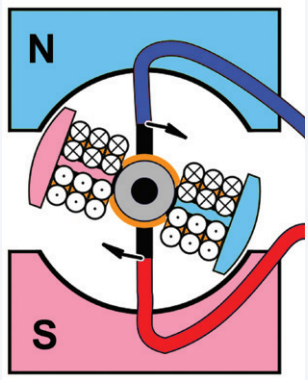


Bild 1. Funktionsprinzip und Aufbau eines permanent erregten Einfach-Elektromotors. Im Stator sitzt oben der Nordpol und unten der Südpol. Der Rotor ist ein Elektromagnet, der über einen Kommutator aus Kupferschleifring und Kohlebürsten versorgt wird (Bild nach Michael Frey, CC 3.0 [3]).

Funktionsweisen

Der historisch früheste und wohl auch einfachste Motor ist der permanent mit konventionellen Dauermagneten erregte Motor von **Bild 1**. Der Rotor besteht aus einem drehbar gelagerten Elektromagneten. Der schwarze Punkt in der Mitte symbolisiert die Achse des Lagers und der graue Ring das Lager. Bei Versorgung mit Gleichspannung wird sein Magnetfeld über einen Kommutator, einem Schleifring mit Kupferflächen und Abgreifkontakten aus Kohle selbstgesteuert und passend zur Position umgeschaltet. Polt man die Versorgungsspannung um, ändert sich auch die Drehrichtung. Es ist evident, dass dieser Motor zum Generator wird, wenn sein Rotor mechanisch gedreht wird. Dann liegt an seinen Klemmen eine pulsierende Gleichspannung an. Bei Verzicht auf den Kommutator hätte man einen Wechselstromgenerator vor sich, und beim Anlegen einer Wechselspannung einen nicht wirklich praktikablen einphasigen Synchronmotor mit Anlaufproblemen.

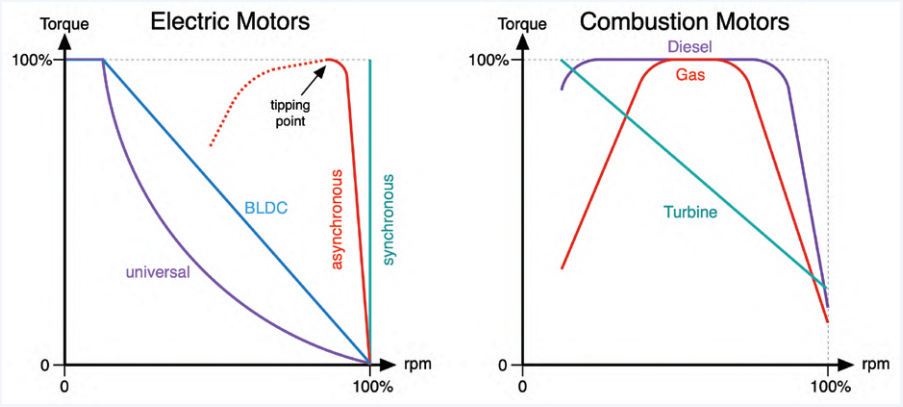


Bild 2. Der idealtypische Verlauf des Drehmoments in Abhängigkeit von der Drehzahl bei verschiedenen Elektromotoren im Vergleich mit drei Verbrennungsmotoren (Bild: Dr. Thomas Scherer).

Aus diesem Einfachmotor lassen sich fast alle anderen Motoren ableiten. Ersetzt man den Permanentmagneten des Stators durch eine Wicklung beziehungsweise einen weiteren Elektromagneten, dann ist man schon beim sogenannten Universalmotor angelangt. Macht man den Motor mehrphasig, landet man bei den Asynchron-, Synchron und BLDC-Motoren, die mit einem elektrischen Drehfeld arbeiten. Der Einfachmotor hat spannungsfrei einen schlechten Freilauf, da die Feldlinien des Permanentmagneten immer noch Haltekräfte bewirken. Fremderregte Motoren hingegen trifft dieses Problem weniger.

Die unterschiedlichen prinzipiellen Charakteristiken dieser Motorenarten sind in **Bild 2** auch im Vergleich zu gängigen Verbrennungsmotoren in idealisierter Form zu sehen. Der Verlauf des Drehmoments in Abhängigkeit von der Drehzahl unterscheidet sich massiv. Allerdings sind die Unterschiede damit noch bei weitem nicht erschöpft und diese anderen Aspekte haben ebenfalls großen Einfluss auf die Eignung in verschiedenen Einsatzszenarien.

Universalmotor

In **Bild 3** ist mit dem Universalmotor die nächste Abwandlung des Einfachmotors von Bild 1 zu sehen, der schon vor über 117 Jahren erfunden wurde. Hierbei wurde schlicht der Permanentmagnet durch eine weitere Wicklung, also durch einen Elektromagneten ersetzt. Der Vorteil zeigt sich vor allen Dingen bei größeren Leistungen, denn hier würden Permanentmagnete nicht nur teuer, sondern auch sehr schwer und platzraubend. Beim klassischen Universalmotor ist die Erregerwicklung des Stators in Reihe mit der Rotorwicklung geschaltet, und somit handelt es sich dabei um einen sogenannten Reihenschlussmotor. Gegenüber dem nicht abgebildeten und weniger gebräuchlichen Nebenschlussmotor, bei dem die Statorwicklung parallel zur Rotorwicklung geschaltet ist, hat der Universalmotor unter anderem den Vorteil eines sehr hohen Anlaufdrehmoments.

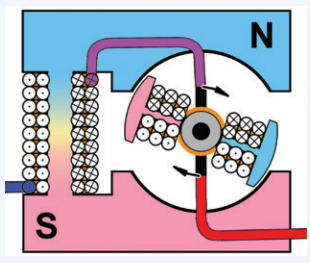


Bild 3. Funktionsprinzip und Aufbau eines Universalmotors, bei dem gegenüber dem Einfach-Motor der Permanent- durch einen Elektromagneten ersetzt wurde (Erweiterungen gegenüber Bild 1: Dr. Thomas Scherer).

Beide Varianten arbeiten sowohl an Gleich- als auch an Wechselspannung. Bei einer Eignung für letztere und für hohe Drehzahlen sind die Kerne von Stator und Rotor aus einem Stapel gegeneinander isolierter Bleche aufgebaut, um Wirbelstrom- und Ummagnetisierungsverluste zu begrenzen. Beiden Varianten ist eigen, dass eine Umpolung beim Betrieb mit Gleichspannung nicht die Drehrichtung ändert. Will man das erreichen, muss man entweder nur die Stator- oder nur die Rotorwicklung umpolen, aber eben nicht beide zugleich.

Aufgrund des hohen Anlaufstroms gerade bei größeren Universalmotoren im kW-Bereich wird aus Gründen des Schutzes des Motors selbst (sowie um das Auslösen von Sicherungen zu verhindern) eine Anlaufsteuerung verwendet, die im einfachsten Fall den Motor beim Einschalten zunächst über einen Lastwiderstand mit geringerem Strom anlaufen lässt. Erst nach einiger Zeit beziehungsweise nach ausreichender Drehzahl wird die Versorgung direkt an die Motorklemmen gelegt. Nicht nur der Widerstandswert muss an die konkreten Bedingungen angepasst sein, sondern auch die Belastbarkeit muss dem Integral des Verlaufs der Stromkurve in der Anlaufzeit gewachsen sein. Die Anlaufwiderstände können also durchaus voluminös ausfallen, weshalb in speziellen Konstruktionen sogar wasserbasierte Anlaufwiderstände eingesetzt werden. Bei richtig großen Universalmotoren reicht manchmal ein einzelner Widerstand nicht aus. Hier werden dann mehrere, in Reihe geschaltete Vorwiderstände nacheinander, beziehungsweise bei Erreichen bestimmter Drehzahlen kurzgeschlossen. Eine wie in **Bild 4** gezeigte Zeitverzögerung ist die einfachste Art des Anlaufschutzes. Intelligenter ist die Erfassung der Drehzahl und das drehzahlabhängige Kurzschließen der Vorwiderstände und/oder das langsame Hochfahren der Spannung über eine Elektronik. Wie schon erwähnt, lassen sich Universalmotoren an Gleich- und

Wechselstrom betreiben. Die Leerlauf- und die Arbeitsdrehzahl unter Last hängen von der Konstruktion und der Höhe der angelegten Spannung ab. Die Abgabeleistung unter Last und die Drehzahlen lassen sich folglich auch durch PWM-Steuerungen einstellen. Wenn sowieso schon eine aufwändigere PWM-Ansteuerung vorhanden ist, lässt sich mit dem dann ohnehin vorhandenen Mikrocontroller leicht ein Anlaufschutz mittels PWM erreichen, indem zunächst mit kleinem Puls/Pausen-Verhältnis gestartet und dieses dann langsam hochgefahren wird. Da bei einem eventuellen Blockieren durch eine übermäßige Last oder schon bei deutlicher Überlast sehr hohe und gegebenenfalls zerstörerische Ströme auftreten können, empfiehlt sich gerade bei größeren Exemplaren neben der Überwachung der Drehzahl die Überwachung des fließenden Stroms. Dies wird in einfachen Fällen durch einen thermisch operierenden Überstromschalter realisiert. Beim Unterschreiten einer Mindestdrehzahl oder beim Überschreiten eines Maximalstroms wird dann der Motor abgeschaltet und so der Motor selbst, seine Elektronik und die mit dem Motor angetriebene Maschine geschützt. Damit lässt sich ein Stromausfall auch für andere Maschinen durch Auslösen der Sicherung in der Elektroinstallation vermeiden.

Universalmotoren werden, wie der Name schon impliziert, in einem weiten Feld von unter 100 W bei Bohrmaschinen und im gesamten Werkzeugbereich sowie bei Haushaltsgeräten bis hin zu Motoren in Lokomotiven und Triebwagen im MW-Bereich an Gleich- und Wechselspannungen eingesetzt. Aufgrund des ungleichmäßigen Drehmomentverlaufs über die Rotation finden sich selbstverständlich kaum so einfache Varianten wie die von Bild 3, sondern häufig einphasige Mehrpolausführungen mit unterschiedlichen Stator- und Rotorformen (**Bild 5**) sowie mit unterschiedlichen Kompensationswicklungen, die ebenfalls den Drehmomentverlauf, das Ausmaß des sogenannten Bürstenfeuers und somit auch die Zuverlässigkeit des Kommutators und sonstige Aspekte beeinflussen. Wenn man den Zusammenhang von Drehmoment und Drehzahl in Bild 2 betrachtet, fällt auf, dass hier die Drehzahl bei gegebener Spannung von der Last abhängt. Eine Drehzahlstabilisierung erfordert daher eine aktive Regelung mit Drehzahlerfassung.

Asynchronmotor

Der Asynchronmotor, dessen prinzipieller Aufbau in **Bild 6** gezeigt wird, hat gegenüber dem Universalmotor entscheidende Vorteile. Der wichtigste ist das völlige Fehlen eines Kommutators, was ihn – abgesehen von den Lagern – verschleißfrei und somit extrem zuverlässig macht. Die größeren Varianten sind fast immer als dreiphasige Drehstromvariante realisiert. Mehrpolige Ausführungen dienen hier einem gleichmäßigeren Drehmomentverlauf über die Rotation. Die abgebildete häufig vorkommende Innenläufer-Version nutzt das magnetische Drehfeld des Stators, das bei Belastung durch den Nachlauf des Rotors in dessen Leiterkäfig einen Strom und somit ein Magnetfeld induziert, dessen reduzierte Gegen-EMK wiederum zum erhöhten Stromfluss durch den Rotor führt. Aus diesem Grund wird der Asynchronmotor im englischen Sprachraum als Induction Motor bezeichnet. **Bild 7** zeigt einen aufgeschnittenen Asynchronmotor im unteren kW-Bereich. Die Polzahl eines solchen Motors ist für die Nenndrehzahl ebenfalls

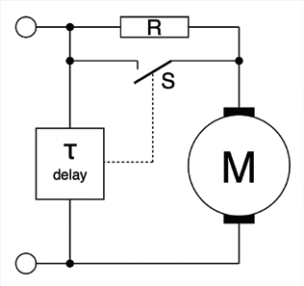


Bild 4. Eine einfache Anlaufschutzschaltung mit Vorwiderstand zur Vermeidung zu großer Anlaufströme.

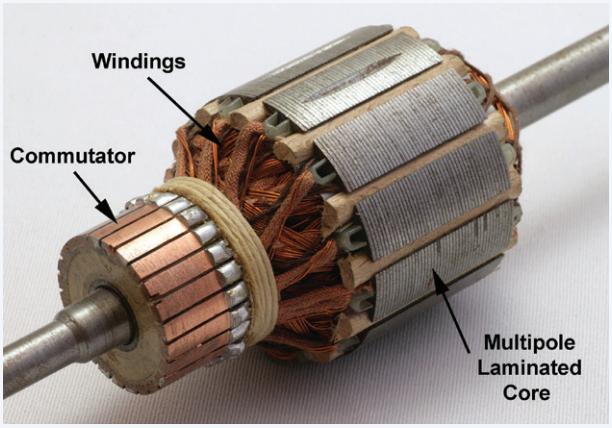


Bild 5. Rotor eines typischen einphasigen Universalmotors. Die Abgriffe am Kommutator korrespondieren mit den Wicklungen und der Polzahl (Bild nach Sebastian Stabinger, CC 3.0 [4]).

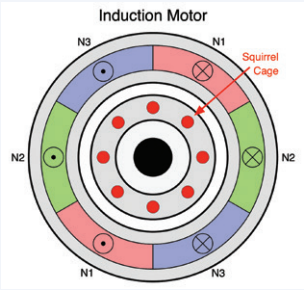


Bild 6. Prinzipieller Aufbau eines Drehstrom-Asynchronmotors (Innenläufer). Der Rotor besteht aus einem Leiterkäfig mit kurzgeschlossenen Windungen (Bild: Dr. Thomas Scherer).

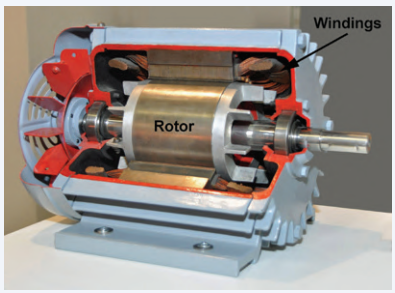


Bild 7. Ein typischer Asynchronmotor mit etwa 1 kW im Schnitt. Man kann die Wicklungen des Stators und den als Kurzschlussläufer ausgeführten Rotor gut sehen (Bild nach S. J. de Waard, CC 3.0 [5]).

Bild 8. Ein kleiner Frequenzumrichter für Asynchronmotoren im unteren kW-Bereich. Diese elektronischen Vorschaltgeräte erlauben die Verstellung der Drehzahl und es gibt sie auch mit mehr Funktionen in größeren Ausführungen (Bild nach C. J. Cowie, CC 3.0 [6]).

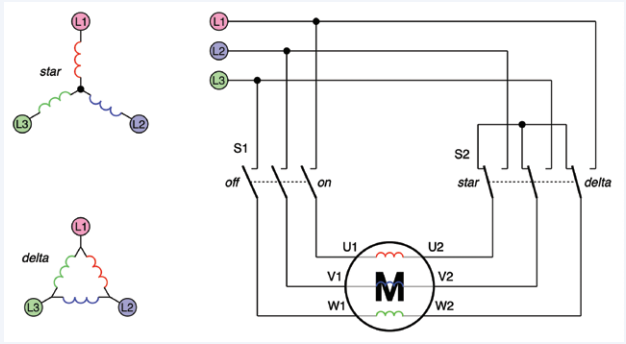


Bild 9. Stern/Dreieckumschaltung per Schütz als einfache Anlaufhilfe zur Vermeidung hoher Anlaufströme.

relevant: Verdoppelt sich die Polzahl, halbiert sich die Drehzahl des Rotors bei gegebener Frequenz der dreiphasigen Wechselspannung. Wie man schon in der Kennlinie von Bild 2 sehen kann, hat ein Asynchronmotor eine recht steile Funktion und der Rotor dreht immer knapp unter der Frequenz des anliegenden Drehfelds. Seine Drehzahl ist aufgrund des Schlupfs bei Belastung immer etwas kleiner als die des Drehfelds und lastabhängig, aber dennoch noch recht starr an die Frequenz des Drehfelds beziehungsweise der speisenden Wechselspannung gebunden. Bei Überlastung wird irgendwann ein Kipppunkt erreicht, ab dem das Drehmoment wieder abnimmt – in diesen Fällen bleibt der Motor stehen und der Strom steigt stark an. Der nutzbare Drehzahlbereich lässt sich durch das Vorschalten eines Frequenzumrichters (**Bild 8**) um einige Faktoren variieren (reduzieren und erhöhen), was vor allen Dingen beim Einsatz in industriellen Maschinen genutzt wird.

Beim Anlauf benötigen größere Asynchronmotoren große Ströme, die durchaus bis zum Faktor 10 des Nennstroms erreichen können. Um den Anlaufstrom zu reduzieren, existieren verschiedene Maßnahmen. Im einfachsten Fall wird im Bereich bis etwa 11 kW häufig eine Stern/Dreieckumschaltung (**Bild 9**) vorgenommen, da der Strom respektive die Leistung bei Sternschaltung nur ein Drittel relativ zur Dreieckschaltung beträgt. Die Umschaltung von Stern- auf Dreieckschaltung wird typischerweise erst bei mehr als 75 % der Nenndrehzahl vorgenommen. Dabei muss darauf geachtet werden, dass die Belastung während des Anlaufs nicht zu groß wird, damit der Kipppunkt nicht erreicht wird. Bei Frequenzumrichtern ist das nicht erforderlich, da hier die Drehzahl beziehungsweise die Frequenz der abgegebenen Drehspannung langsam hochgefahren werden kann.

Aufgrund der extremen Robustheit und des hohen Wirkungsgrads dieses Motors findet man ihn nicht nur bei Maschinen aller Art in der industriellen Produktion, sondern durchaus auch für Antriebe von elektrischen Fahrzeugen. Selbst Tesla nutzt in etlichen aktuellen Modellen hochdrehende und räumlich klein gebaute Asynchronmotoren mit Untersetzungsgetriebe und erreicht die hier nötige, sehr große Drehzahlbandbreite durch eine ausgeklügelte Motorsteuerung mit Strom- und Drehzahlerfassung. Mit Elektronik kann man also den größten Nachteil der starren Kopplung an die Frequenz der versorgenden Wechselspannung weitgehend kompensieren. Dreht sich der Rotor schneller als das Drehfeld, fungiert die Asynchronmaschine als Generator. Es ergibt sich bei Nutzung als Antriebsmotor eine Rekuperation, was für Fahrzeuge und größere stationäre Maschinen nicht unerheblich ist.

Synchron- und BLDC-Motor

Ein Synchronmotor ist im Prinzip ein an Drehstrom betriebener dreiphasiger Universalmotor. In einer häufigen Variante wird die Rotorwicklung über getrennte Schleifringe gespeist, die aber anders als beim Universalmotor keine Polwender darstellen und somit geschlossen sind und wenig Verschleiß ermöglichen. Anders als beim Innenläufer von **Bild 10** dargestellt, kann man auch den Rotor mit Permanentmagneten ausstatten, was Schleifringe erübrigt. Bei größeren stationären Motoren ist die abgebildete fremderregte Variante preiswerter und thermisch sicherer, denn Permanentmagnete können schon bei kurzfristiger Übertemperatur Schaden nehmen. Es existieren auch Außenläufer, bei denen der Innenteil statisch fixiert ist und Schleifringe bei Permanentmagneten auf dem Außenläufer entfallen, was diese Variante sehr langlebig und wartungsfrei macht. Wie aus dem zugehörigen Diagramm des Drehmoments in Abhängigkeit von der Drehzahl zu ersehen, bleibt die Drehzahl konstant und starr an das Drehfeld gekoppelt. Der

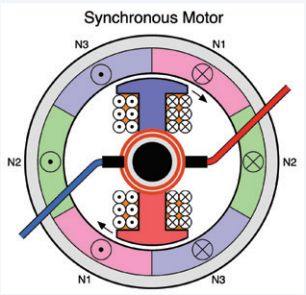


Bild 10. Der prinzipielle Aufbau eines einfachen Drehstrom-Synchronmotors (Innenläufer). Der Rotor kann zur Vermeidung des Kommutators auch mit Permanentmagneten bestückt sein (Bild: Dr. Thomas Scherer).

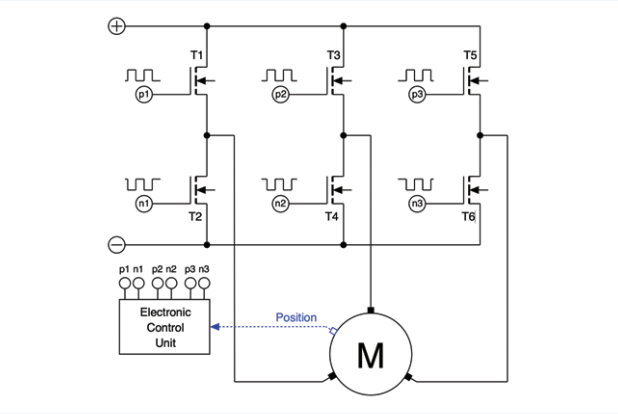


Bild 11. Prinzip der elektronischen Kommutation über eine im Gegentakt phasenversetzt angesteuerte Dreifach-Halbbrücke bei einem BLDC-Motor.

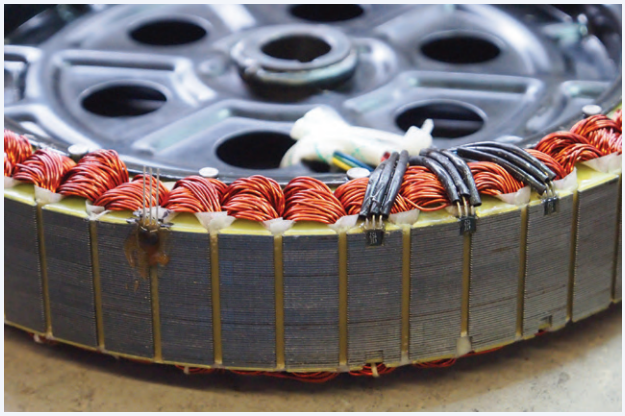


Bild 12. Stator bei einem 2-kW-BLDC-Motor in Radnabenausführung (Außenläufer). Man kann nicht nur die vielpoligen Wicklungen samt Blechpaket gut sehen, sondern auch die Positionierung der Hallsensoren zur Erfassung der Winkelposition des Rotors (Bild: Dr. Thomas Scherer).

Rotor hat also bei Belastung keinen Schlupf, sondern sein Phasenwinkel eilt belastungsabhängig dem Drehfeld nach. Dieser Motor kann wie die anderen auch als Generator betrieben werden. Für einen gleichmäßigeren Drehmomentverlauf über die Rotation werden Synchronmotoren vielpolig gebaut, wobei sich auch hier die Polzahl umgekehrt proportional zur Drehzahl verhält.

Der große Vorteil des Synchronmotors steckt schon in seiner Bezeichnung: Man kann damit ohne eigene Regelung sehr präzise und konstante sowie belastungsunabhängige Drehzahlen erreichen. Da die Schleifringe anders als beim Universalmotor nicht unterbrochen sind und die Erregerleistung relativ klein ist, unterliegen die Kohlebürsten nur sehr geringem Verschleiß. Beim Einsatz von Permanentmagneten im Rotor erhält man ebenfalls extrem robuste Motoren mit kleinen Formfaktoren. Von Nachteil ist der etwas schwierigere Selbstanlauf, weshalb in größeren Exemplaren im Rotor gelegentlich zusätzliche Kurzschlusskäfige integriert werden, um sie dann quasi als Asynchronmotor hochzufahren. Bezüglich des Anlaufs sind die gleichen Schutzmaßnahmen möglich und gegebenenfalls sogar erforderlich wie beim Asynchronmotor – inklusive moderner Frequenzumrichter mit Zusatzfunktionen.

Aus einem Synchronmotor wird ein BLDC-Motor oder bürstenloser Gleichspannungsmotor, wenn das Drehfeld in den Wicklungen des Stators nicht etwa mit einer dreiphasigen Wechselspannung generiert wird, sondern die Wicklungen über drei Halbbrücken (**Bild 11**) an Gleichstrom betrieben werden und auf diese Weise die nötige dreiphasige Wechselspannung (übrigens wie bei einem Frequenzumrichter) elektronisch generiert wird. Hinzu kommt, dass die Winkelposition des Rotors erfasst und mit diesen Informationen die drei Halbbrücken getaktet werden. Resultat ist ein Quasi-Synchronmotor, der sein Drehfeld selbst steuert und dessen Drehzahl von der Versorgungsspannung und der Belastung abhängt. Gegenüber dem Universalmotor ergibt sich nicht nur ein unkomplizierterer Bau und bei Verwendung von Permanentmagneten nahezu Wartungsfreiheit, sondern auch eine optimalere Charakteristik von Drehmoment in Abhängigkeit von der Drehzahl. Fügt man noch eine PWM-Steuerung hinzu, lässt sich die Leerlaufdrehzahl über das Puls/Pausenverhältnis einstellen. Regelt man die PWM mit dem fließenden Strom als Bezugsgröße, korreliert der Strom recht linear mit dem Drehmoment bei gegebener Last und Drehzahl.

Beide Varianten werden in stationären Maschinen in unterschiedlichen Leistungsklassen verbaut. Für Fahrzeugantriebe eignen sich allerdings BLDC-Motoren besser, weil man auch hier kommutatorfreie Lösungen bauen kann und komplexe Steuerungen ohnehin erforderlich sind. Ein Spezialfall davon ist der sogenannte Radnabenmotor (**Bild 12**), wie er oft bei E-Bikes zum Einsatz kommt und bei Antrieben mit kleineren bis mittleren Leistungen verwendet wird. Hier handelt es sich um einen Außenläufer, dessen äußerer Rotor oft mit Permanentmagneten bestückt ist. Durch eine hohe Polzahl erreicht man niedrige Drehzahlen und hohe Drehmomente bei gleichmäßigem Drehmomentverlauf sowie extreme Zuverlässigkeit. Permanentmagnete sind aber gerade bei größeren Leistungen nicht nur teuer, sondern auch schwer. Gerade beim Radnabenmotor handelt man sich damit eine hohe ungefederte Masse ein, die den Fahrkomfort verschlechtert. Aus diesem Grund setzt man im Fahrzeugbau eher auf schnelllaufende und kompakte Asynchronmotoren, welche die Räder über bewegliche Wellen antreiben.

Außerdem...

Auch wenn Elektromotoren nicht nur die Zukunft gehört, sondern sie absehbar so gut wie alle anderen Motorenarten verdrängen werden, gibt es doch einiges bei ihrem Einsatz zu beachten. Heute sind Elektromotoren weitgehend ausentwickelt und große Neuheiten sind nicht mehr zu erwarten. Womit man aber rechnen kann, das sind evolutionäre Optimierungen. Dies betrifft nicht nur die zur Verfügung stehenden Materialien und die Konstruktions- und Fertigungstechniken, sondern vor allem auch die intelligente

Tabelle 2. Die 20 größten Hersteller von Elektromotoren (Stand 2019) [8].

Ranking	Hersteller	Sitz
1	Siemens	Deutschland
2	Toshiba	USA
3	ABB	Schweiz
4	Nidec Motor	Japan
5	Rockwell Automation	USA
6	Ametek	USA
7	Regal Beloit	USA
8	Johnson Electric	Hongkong
9	Franklin Electric	Indien
10	Allied Motion	USA
11	Faulhaber	Deutschland
12	General Electric	USA
13	Danaher Motion	USA
14	WEG	Brasilien
15	maxon motor	Schweiz
16	TECO Westinghouse	Taiwan
17	Hitachi	Japan
18	Lincoln Electric	USA
19	Piela Electric	USA
20	Dumore Corporation	USA

elektronische Steuerung. Nicht nur werden Leistungshalbleiter immer besser (schneller, effizienter), sondern gerade die Rechenleistung kostet immer weniger, was komplexere Funktionen und sogar immer noch eine etwas bessere Effizienz des Gesamtpakets aus elektromechanischem Motor und elektronischer Steuerung möglich macht.

Letztlich ist die Auswahl eines Elektromotors für eine spezielle Anwendung immer eine Suche nach dem optimalen Kompromiss aus recht vielen Faktoren. Es braucht dazu viel Know-How, doch als Kunde bekommt man heute recht gute Unterstützung durch die Spezialisten der Hersteller von Elektromotoren. Es gibt hunderte Hersteller, und die Übersicht ist nicht ganz so einfach zu bekommen. Der Gesamtumsatz aller Hersteller lag im Jahr 2020 übrigens bei immerhin fast 150 Milliarden $ [7]; in den nächsten Jahren erwarten Analysten eine Umsatzsteigerung von gut 6 % pro Jahr. In **Tabelle 2** sind die 20 größten Hersteller aufgelistet. Man sieht die deutliche Dominanz der USA, auch wenn der Markführer in Deutschland sitzt. Suchen Sie allerdings etwas Spezielles, reicht dieses Wissen nicht aus und nicht nur eine tiefe Recherche, sondern auch eine Einarbeitung in die Materie ist unabdingbar!

210555-02

Ein Beitrag von

Autor: **Dr. Thomas Scherer**
Redaktion: **Jens Nickel**
Layout: **Giel Dols**

Sie haben Fragen oder Kommentare?

Haben Sie technische Fragen oder Anmerkungen zu diesem Artikel? Schicken Sie eine E-Mail an die Elektor-Redaktion unter editor@elektor.com.

PASSENDE PRODUKTE

- **SmartPi 2 – Smart Meter für Raspberry Pi (inkl. 3 Stromklemmen) (SKU 18165)**
  www.elektor.de/18165
- **Stromklemme für SmartPi 2 (SKU 18167)**
  www.elektor.de/18167
- **PeakTech Stromzange 4350 (SKU 18161)**
  www.elektor.de/18161

WEBLINKS

[1] M. Claussen, „Motorsteuerung mit H-Brücken“, ElektorMag Januar/Februar 2022: www.elektormagazine.de/210491-02
[2] T. Scherer, „Steuerung von Elektromotoren“, Elektor Januar/Februar 2019: www.elektormagazine.de/magazine/elektor-69/42253
[3] Einfach-Elektromotor: https://tinyurl.com/yb98ug89
[4] Rotor eines Universalmotors: https://tinyurl.com/shjy66sj
[5] Schnitt eines Asynchronmotors: https://tinyurl.com/4f89exsh
[6] Frequenzumrichter: https://tinyurl.com/rpycby87
[7] Markt für Elektromotoren: https://tinyurl.com/a7suakfp
[8] Ranking der Hersteller von Elektromotoren: https://tinyurl.com/yjrjju5v

ESP32-C3:
32-Bit-RISC-Einkerner
Ein erstes Hands-on im Elektor-Labor

Bild 1. ESP32-C3-DevKitC-02.

Von **Mathias Claußen** (Elektor-Labor)

Mit Spannung erwartet wurde der ESP32-C3 aus dem Hause Espressif: Statt zwei Kernen wie der „große" ESP32 besitzt der neue Controller nur einen, und zwar mit der offenen RISC-V-Architektur. Im Elektor-Labor haben wir den ESP32-C3 und das ESP32-C3-DevKitC-02 einem ersten Praxistest unterzogen.

Wer dem Autor auf Twitter [1] folgt, wird bereits gesehen haben, dass bei ihm ein paar ESP32-C3-DevKits auf dem Tisch liegen. Und diejenigen Leser, welche unsere *Lab Notes* vom Juni [2] gelesen haben, werden bereits wissen, dass der ESP32-C3 softwareseitig noch stetige Verbesserungen erfährt.
Vorab ein paar Hinweise zu diesem Artikel: Unsere DevKits haben noch einen ESP32-C3-Chip in der Revision 2 verbaut. Die DevKits werden noch mit einem beidseitig bedrucken A4-Blatt geliefert, das auf die Probleme dieser Revision hinweist. Unter anderem sind das eine hohe Stromaufnahme in Deep-Sleep-Mode und ein Totalausfall des USB/JTAG-Adapters im Chip. Die kommende Revision 3 soll dieses beheben, wie aus dem Tweet unter [3] ersichtlich ist.
Auch die IDF wird laufend für den ESP32-C3 angepasst und kann noch Bugs enthalten. Da die Arduino-Unterstützung drauf aufbaut, können diese Bugs auch dort wiedergefunden werden. Zu der Zeit, als dieser Artikel geschrieben wurde, musste der Development-Release des Arduino-Support-Packages installiert werden.
Dazu muss der folgende Link unter *Preference* in die *Additional Boards Manager URL*s eingetragen werden:

https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_dev_index.json.

Damit wird momentan die Version *2.0.0-rc1* installiert.
Wer nun ein ESP32-C3 basiertes Kit kauft, sollte eine Revision 3 des Chips erhalten, wie aus dem Forumseintrag von ESP32.com [4] zu sehen ist. Die Revision 2 wurde nur in geringen Stückzahlen für die ersten DevKits verwendet.

Das ESP32-C3-DevKitC-02

Zuerst sollen der ESP32-C3 und das DevKitC-02 (**Bild 1**) vorgestellt werden. Bei dem ESP32-C3 handelt es sich um einen Nachfolger für den ESP8266. Auch hier wird ein einzelner Prozessorkern verwendet, der mit bis zu 160 MHz getaktet werden kann und WiFi im 2,4-GHz-Band nach BGN beherrscht. An der Stelle hören die Gemeinsamkeiten mit dem ESP8266 auch schon auf. Der ESP32-C3 hat von seinem größeren Verwandten, dem ESP32, einen Großteil der Peripherie bekommen. Damit beherrscht der Chip neben WiFi auch BLE 5.0 und Bluetooth Mesh. Zusätzlich bringt er die liebgewonnene

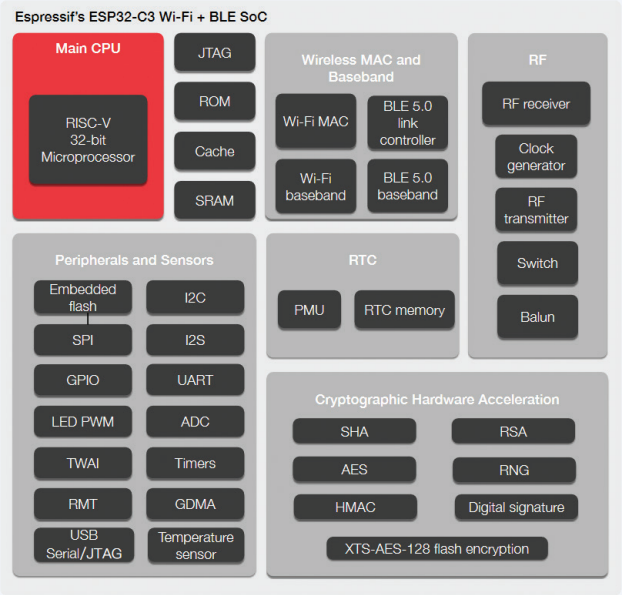


Bild 2. ESP32-C3 Blockdiagramm.

- RISC-V CPU mit 160 MHz
- 400 KB SRAM (16 KB als Flash Cache)
- Integriertes 2,4-GHz-WLAN (BGN)
- Bluetooth LE 5.0
- Kryptographische Hardware-Beschleunigung
- 22 programmierbare GPIOs
- 2 x 12-Bit-SAR ADC
- 3 x SPI (unterstützt SPI, Dual SPI, Quad SPI und QPI)
- 2 x UART (unterstützt RS232, RS485, IrDA mit bis zu 5 MBd)
- 1 x I²C (bis zu 800 kbit/s)
- 1 x I²S
- RMT (Fernsteuerungs-Peripheriegeräte)
- TWAI (CAN 2.0 b kompatibel / ISO 11898-1)
- PWM
- Interner USB/JTAG-Adapter

Tabelle 1. Spezifikationen des ESP32-C3

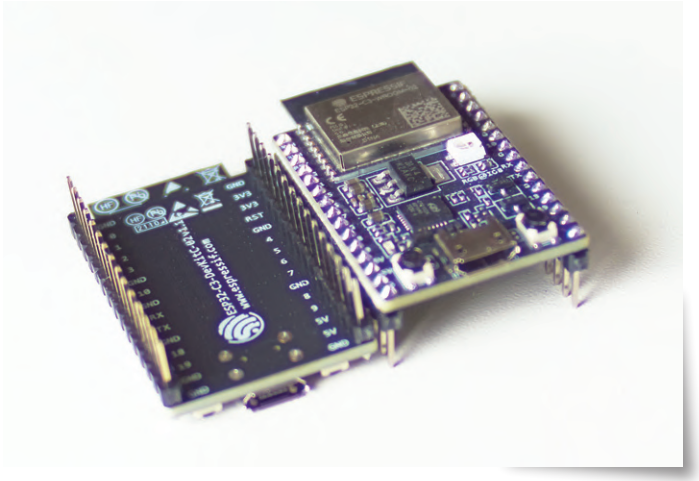


Bild 3. Ober- und Unterseite des ESP32-C3-DevKitC-02.

I/O-Matrix mit, so dass wir fast jede Funktion mit fast jedem Pin nutzen können. **Bild 2** zeigt das Blockdiagramm des ESP32-C3, inklusive des USB-Seriell/JTAG-Adapters.
Auch bei den anderen Funktionen im Chip ist klar zu erkennen, dass diese aus dem ESP32 stammen. **Tabelle 1** zeigt eine Übersicht der Funktionen. Mit seinen 384 kB RAM bietet der ESP32-C3 etwa fünf Mal mehr RAM als der ESP8266 (80 kB). Etwas, was den ESP32-C3 von allen anderen ESP32- oder ESP8266-Chips unterscheidet, ist der verwendete Prozessorkern. Während alle Vorgänger auf einem Tensilica L106 oder LX6/LX7 [5] basieren, steckt im ESP32-C3 ein Prozessorkern mit RISC-V-Befehlssatz. Das bedeutet, dass bestehende Compiler für diesen Kern aus dem RISC-V-Ökosystem verwendet werden können. Verbesserungen dieser Compiler und der entsprechenden Tools stehen damit dem ESP32-C3 offen. Es muss nicht erst wie beim ESP8266 eine Toolchain durch die Community geschaffen werden.
Auf dem DevKitC-02 selbst ist gezwungenenmaßen ein USB-Seriell-Wandler verbaut sowie eine WS2812-kompatible RGB-LED. Eine genaue Übersicht der Komponenten kann dem Schaltplan [6] für das Board entnommen werden. **Bild 3** zeigt das ESP32-C3-DevKitC-02 von der Oberseite und der Unterseite.

Das etwas andere Blinky-Beispiel

Wie schon erwähnt, ist eine WS2812-kompatible RGB-LED verbaut, so dass es mit dem Setzen eines I/O-Pins nicht getan ist, um die LED zum Leuchten zu bringen. In Verbindung mit der Arduino-IDE kann aber der NeoPixel-Bibliothek von Adafruit verwendet werden. Der Code in **Listing 1** lässt die RGB-LED rot blinken, besondere Anpassungen für den ESP32-C3 scheinen nicht nötig zu sein. Auch der Upload erfolgt genauso einfach wie bei einem ESP32. Durch die I/O-Matrix können auch an anderen Pins WS2812-LEDs betrieben werden.

Chip-Typ und Version auslesen

Bei den ESP32-Chips lassen sich grundlegende Eigenschaften des Chips auslesen. Das **Listing 2** gibt die Informationen auf der seriellen Schnittstelle bei 115.200 Baud aus (mit 9.600 Baud hat der Chip in Revision 2 und mit dem verbauten USB-Seriell-Wandler verstümmelte Zeichen ausgegeben). Wer sich ein ESP32-C3-Kit zulegt, kann so sehen, welche Chiprevision vorliegt. Vor allem wer einen Blick in die Errata eines Chips wirft, wird froh sein, wenn er sein Exemplar klar den Bugs zuordnen kann.

Portierung von ESP32-Projekten

Mit die interessantesten Funktionen des ESP32-C3 sind sicherlich WiFi und BLE. Viele werden bei ihren WiFi-Projekten auch SPIFFS oder littlefs verwenden, um Webseiten oder andere Daten auf dem ESP32 zu verwalten.

Listing 1. Blinky

```
/* This sample needs the Adafruit NeoPixel library */
#include <Adafruit_NeoPixel.h>
 #define PIN        8
 #define NUMPIXELS  1
Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);

void setup() {
     pixels.begin(); // INITIALIZE NeoPixel strip object (REQUIRED)
     pixels.clear(); // Set all pixel colors to 'off'
 }

void loop() {
     delay(1000);                          // wait for a second
         pixels.setPixelColor(0, pixels.Color(255, 0, 0));
         pixels.show();
     delay(1000);                          // wait for a second
         pixels.setPixelColor(0, pixels.Color(0, 0, 0));
         pixels.show();
 }
```

Wer mit der Arduino-IDE 1.X unterwegs ist, sollte sich eine gepatchte Version [7] für den Uploader installieren, da die Originalversion von me-no-dev [8] nicht mit dem ESP32-C3 zusammenarbeitet.

Als Beispiel haben wir den Code [9] des ESP32-Mini-NTP-Servers [10] verwendet. Dieser kann im eigenen Netzwerk die Uhrzeit per NTP bereitstellen und ist für das Arduino-Framework geschrieben. Nach ein paar Anpassungen der Pins lässt sich der Code ohne Probleme kompilieren und auf den ESP32-C3 hochladen. Damit lässt sich viel Code und Wissen vom ESP32 auf dem ESP32-C3 transportieren, und auch Neulinge brauchen vor diesem Chip keine Angst haben. Die meisten Beispiele für den ESP32 laufen auch auf dem ESP32-C3, und da auch hier ein FreeRTOS im Hintergrund werkelt, kann wie bei einem ESP32 mit allen Vor- und Nachteilen des FreeRTOS gearbeitet werden.

Da der ESP32-C3 im Moment nur durch den Developer-Zweig des Arduino-Frameworks für die ESP32-Chips unterstützt wird, haben noch nicht alle IDEs eine volle Unterstützung für den Chip. Mit dem Fortschreiten des Frameworks für den ESP32 wird diese aber früher oder später Einzug halten.

Zusammenfassung

Mit dem ESP32-C3 steht eine kostengünstige Alternative zum ESP8266 bereit, die einen Großteil an Peripherie des ESP32 erhält. Mit dem integrierten USB/Seriell- und JTAG-Adapter kann der Chip über USB Daten austauchen. Es lässt sich sogar das Debuggen von Code starten (sollten in der Revision 3 nicht noch unvorhergesehene Dinge auftreten...).

WEBLINKS

[1] @ElektorMathias bei Twitter : https://twitter.com/ElektorMathias/status/1386623477604626433
[2] Elektor Lab Notes Juni 2021, Elektormagazine.de: https://www.elektormagazine.de/news/elektor-lab-notes-june-2021-de
[3] Tweet über USB Serial / JTAG Unterstützung beim ESP32-C3: https://twitter.com/ElektorMathias/status/1392475706647584776
[4] Forum-Thread zu den ESP32 C3 Revisionen: https://esp32.com/viewtopic.php?t=21040
[5] Wikipedia - Cadence Tensilica Produkte: https://de.wikipedia.org/wiki/Tensilica
[6] Schaltplan ESP32 C3 DevKitC-02: https://dl.espressif.com/dl/schematics/SCH_ESP32-C3-DEVKITC-02_V1_1_20210126A.pdf
[7] ESP32 Data-Uploader für Arduino IDE (gepatchte Version): https://github.com/lorol/arduino-esp32fs-plugin/releases
[8] ESP32 Data-Uploader für Arduino IDE: https://github.com/me-no-dev/arduino-esp32fs-plugin
[9] Github-Repository Mini NTP Server mit GPS : https://github.com/ElektorLabs/180662-mini-NTP-ESP32
[10] Mini-NTP-Server mit GPS auf Elektor Labs: https://www.elektormagazine.de/labs/mini-ntp-server-with-gps
[11] Material zu diesem Artikel: https://bit.ly/3CD6iNs

Bestehenden Code kann man dank des Arduino-Frameworks mit dem ESP32-C3 verwenden. Die Menge an RAM und Flash erlaubt es auch, größere Projekte mit dem Chip umzusetzen. Wir sind gespannt, wann die ersten ESP32-C3 Boards von Drittanbietern in den europäischen Shops erscheinen und welche Funktionen diese bieten werden. Wer schon ein ESP32-DevKitC-02 zuhause hat, kann bis dahin schon einmal anfangen, Code zu schreiben und zu testen. Die Arduino-Sketche zu den Listings finden Sie auf unserer GitHub-Seite [11].

210466-02

Sie haben Fragen oder Kommentare?

Haben Sie technische Fragen oder Kommentare zu diesem Artikel? Schicken Sie eine E-Mail an den Autor unter mathias.claussen@elektor.com oder kontaktieren Sie Elektor unter redaktion@elektor.de.

Listing 2. Auslesen der Chiprevision

```
/* From the IDF documentation at https://github.com/espressif/esp-idf/components/esp_hw_support/include/
esp_chip_info.h */
 /*
  typedef enum {
     CHIP_ESP32  = 1, //!< ESP32
     CHIP_ESP32S2 = 2, //!< ESP32-S2
     CHIP_ESP32S3 = 4, //!< ESP32-S3
     CHIP_ESP32C3 = 5, //!< ESP32-C3
     CHIP_ESP32H2 = 6, //!< ESP32-H2
 } esp_chip_model_t;
// Chip feature flags, used in esp_chip_info_t
 #define CHIP_FEATURE_EMB_FLASH      BIT(0)      //!< Chip has embedded flash memory
 #define CHIP_FEATURE_WIFI_BGN       BIT(1)      //!< Chip has 2.4GHz WiFi
 #define CHIP_FEATURE_BLE            BIT(4)      //!< Chip has Bluetooth LE
 #define CHIP_FEATURE_BT             BIT(5)      //!< Chip has Bluetooth Classic
 #define CHIP_FEATURE_IEEE802154     BIT(6)      //!< Chip has IEEE 802.15.4
typedef struct {
     esp_chip_model_t model;  //!< chip model, one of esp_chip_model_t
     uint32_t features;       //!< bit mask of CHIP_FEATURE_x feature flags
     uint8_t cores;           //!< number of CPU cores
     uint8_t revision;        //!< chip revision number
 } esp_chip_info_t;
*/

void setup() {
     Serial.begin(115200);
 }

void loop() {
     delay(5000);
     Serial.println(“esp_chip_info()”);
     Serial.println(“----------------------------------------------------------------”);
     esp_chip_info_t info;
     esp_chip_info(&info);
     Serial.print(“Chip Model: “);
     switch(info.model){
         case 1:{
             Serial.println(“ESP32”);
         }break;
        case 2:{
             Serial.println(“ESP32-S2”);
         }break;
```

Ein Beitrag von
Text und Bilder: **Mathias Claußen**
Redaktion: **Jens Nickel**
Layout: **Giel Dols**

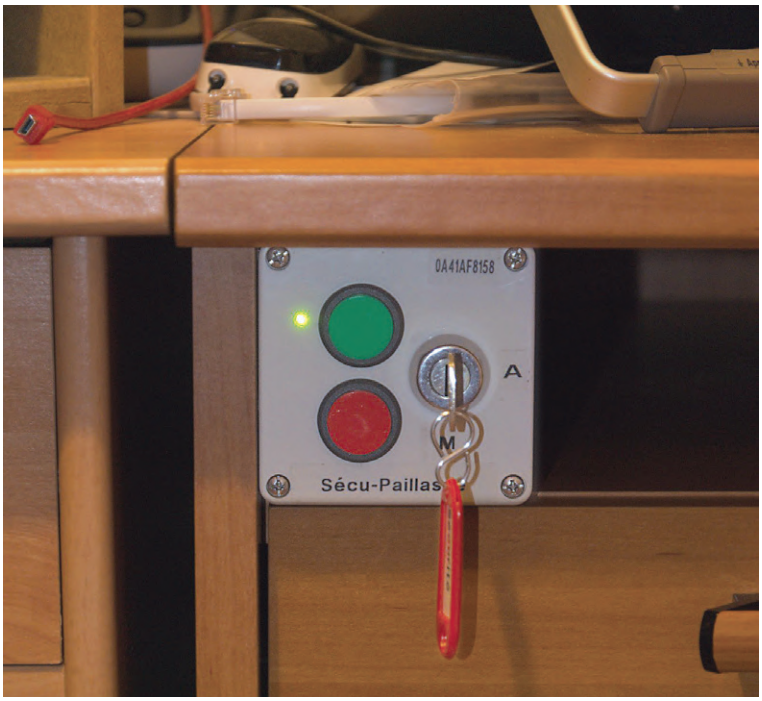
PASSENDE PRODUKTE

- ESP-C3-12F-Kit Development Board mit 4 MB Flash (SKU 19855)
  www.elektor.de/19855

```
        case 4:{
             Serial.println("ESP32-S3");
         }break;
        case 5:{
             Serial.println("ESP32-C3");
         }break;
        case 6:{
             Serial.println("ESP32-H2");
         } break;
        default:{
             Serial.print("Unknown Chipmodel");
             Serial.println(info.model);
         }
     }

    Serial.print("Featues :");
     if(info.features&CHIP_FEATURE_EMB_FLASH){
         Serial.print(" Embedded Flash ");
     }
     if(info.features&CHIP_FEATURE_WIFI_BGN){
         Serial.print(" WiFi(BGN) ");
     }
     if(info.features&CHIP_FEATURE_BLE){
         Serial.print(" BLE ");
     }

     if(info.features&CHIP_FEATURE_BT){
         Serial.print(" BT CLassic ");
     }
    if(info.features&CHIP_FEATURE_IEEE802154){
         Serial.print(" IEEE802.155.4 ");
     }
     Serial.println("");
     Serial.print("Cores: ");
     Serial.println(info.cores);
     Serial.print("Chip Revision: ");
     Serial.println(info.revision);
     Serial.println("-----------------------------------------------------------------");
     Serial.println();
}
```

Schützen Sie sich selbst und andere!

Hauptnetzschalter für den Labortisch selbst gebaut



Von **Philippe Le Guen** (Frankreich)

Ist Ihr Elektronik-Arbeitsplatz für Besucher auch sicher, wenn er unbeaufsichtigt ist? Sie wissen vielleicht, was Sie tun, aber was ist mit Familienmitgliedern und Freunden? Riskieren sie einen Stromschlag, wenn sie versehentlich etwas anfassen? Vermeiden Sie sicherheitshalber alle Risiken mit diesem „Master Bench Power Switch“!

Warum ich es für notwendig befunden habe, meinen Labortisch sicherer zu machen? Dafür gibt es mehrere Gründe, von denen einige ganz persönlicher Natur sind. Zunächst einmal wollte ich in der Lage sein, alle Geräte auf dem Labortisch - sowohl Messinstrumente als auch Geräte, die sich in der Entwicklung oder in der Reparatur befinden - mit einem einzigen Tastendruck ein- und auszuschalten. Außerdem möchte ich nicht, dass der „Prüfstand“ nach einem Ausfall oder einer (versehentlichen) Unterbrechung der Stromversorgung automatisch wieder eingeschaltet wird. In solchen Fällen ist es sicherer, wenn die Stromversorgung manuell wiederhergestellt werden muss.

Der dritte Grund ist, dass ich, nachdem mein Sohn sein Elternhaus verlassen hat, mein kleines Labor in seinem ehemaligen Schlafzimmer installiert habe, wo es für die Schar meiner Enkelkinder zugänglich ist. Da ihre kleinen Hände alles erforschen und anfassen, was in ihrer Reichweite ist, wollte ich das Tischnetzteil mit einem Schlüsselschalter schützen, so dass es für Unbefugte unmöglich ist, es einzuschalten.

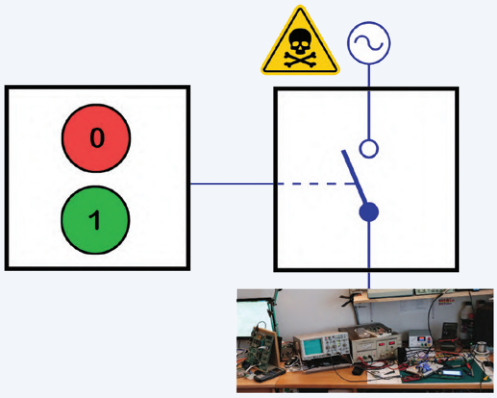


Bild 1. Die Schaltung ist in zwei separate Geräte aufgeteilt, die Steuerbox (links) und die Leistungsstufe (rechts). Der zusätzliche Schlüssel-Hauptschalter wird in die Steuerbox eingebaut.

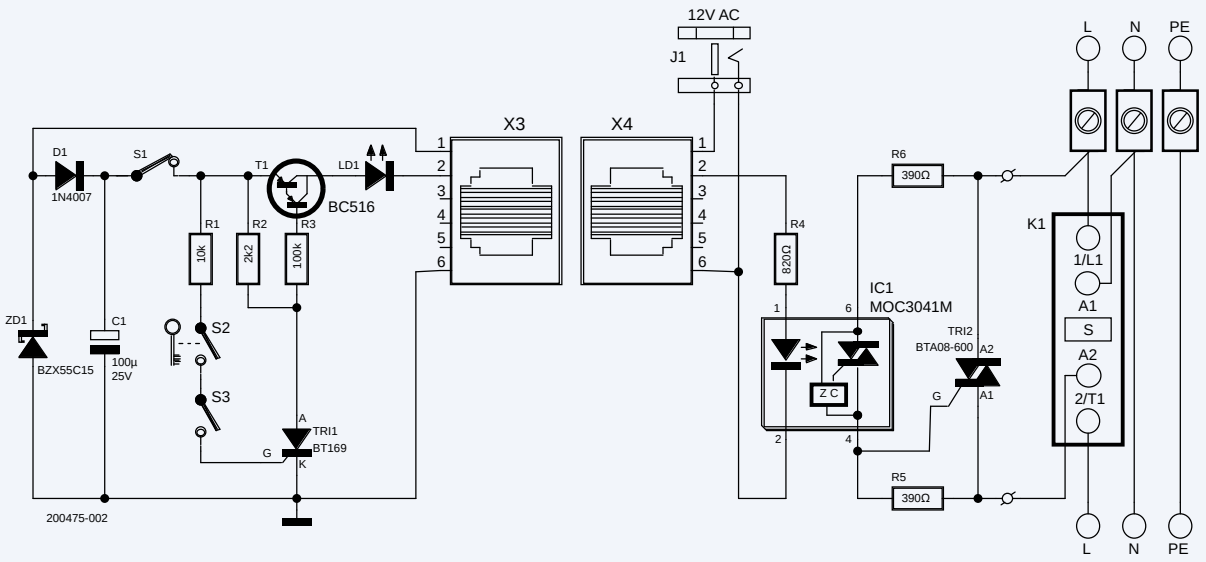


Bild 2. Der linke Teil der Schaltung, die Steuerbox, wird von der rechts dargestellten Leistungsschaltstufe versorgt.

Anmerkung der Redaktion: Ein Abschalten des N-Leiters zur Netztrennung ist nicht in allen Fällen sinnvoll und auch nicht statthaft, da die in einem abgeschalteten (zum Beispiel induktiven) Verbraucher gespeicherte Energie nicht über den N-Leiter abfließen kann. Aus diesem Grund ist im Schaltplan auch ein nur einpoliges Schütz eingezeichnet. Möchte man den N-Leiter dennoch abschalten, sollte ein Schütz mit vor/nacheilendem N-Kontakt eingesetzt werden.

Spezifikationen

Um eine zusätzliche Sicherheit zu erreichen, wollte ich den Steuerkreis mit Niederspannung (12 V) versorgen und die Leistungsstufe durch einen Optokoppler galvanisch trennen. Das Ganze sollte ein elektronisches System sein, das dank der Verwendung von Standardkomponenten leicht selber zu bauen und zu reparieren ist.

Ja, ich hätte durchaus einen dieser handelsüblichen NVR-Sicherheitsschalter (no volt release) verwenden können, wie sie bei Werkzeugmaschinen eingesetzt werden, aber ich wollte ihn – selbst ist der Mann – in Eigenregie entwerfen, auch damit er den kleinen Sicherungskasten passt, den ich aus diesem Grund in meinem Labor installiert habe.

Ein Entwurf in zwei Teilen

Ich habe ein einfaches elektronisches System entworfen, das nur Standardbauteile verwendet (**Bild 1**). Die grüne und die rote Drucktaste zum Ein- und Ausschalten des Labortisches sind zusammen mit dem Schlüsselschalter in einem kleinen Polycarbonat-Gehäuse untergebracht. Ich habe mich für dieses Gehäuse entschieden, weil ich es bereits hatte (oder weil man es sich notfalls problemlos in jedem gut sortierten Baumarkt besorgen kann) und weil es perfekt in mein Labor passt. Den Schlüsselschalter, der in Reihe mit dem Ein-Taster geschaltet ist, habe ich in meiner „Reserveteilekiste" gefunden, man kann irgendein nichtmomentanes Exemplar verwenden, bei dem der Schlüssel in beiden Positionen abgezogen werden kann. Zur Inspiration können Sie sich an [1] orientieren.

Der eigentliche Schalter ist ein zweipoliges (für L1 und N) Installations- oder Schwachstromschütz mit NO-Kontakten, das 20 A schalten kann und das ebenfalls aus meiner Bastelkiste stammt. Ein Schütz ist einem gewöhnlichen Relais ähnlich, beim Schalten größerer (und induktiver) Lasten wegen seiner Funkenlöschkammern aber robuster. Außerdem besitzt mein Schütz einen mechanischen Schalter, mit dem man es permanent ein- oder ausschalten kann, was praktisch ist, wenn der Steuerkreis mal ausfällt (man weiß ja nie, was passiert).

Die Leistungsstufe ist in einem DIN-Hutschienengehäuse mit 1 TE untergebracht, das auf der Schiene des Sicherungskastens neben dem Schütz montiert wird. Die vollständige Schaltung ist in **Bild 2** zu sehen. Die Steuerbox befindet sich am Labortisch in einiger Entfernung vom Leistungsrelais und seiner Schnittstelle. Beide Teile sind durch Kabel mit RJ11-Steckern an beiden Enden verbunden. Ich habe dafür ein gewöhnliches Cat5-Ethernet-Kabel verwendet.

Die Niederspannungsversorgung

Das Kabel transportiert die 12-V-Versorgungswechselspannung vom Leistungsteil zur Steuerbox und deren Ausgangssignal in die umgekehrte Richtung. Die Wechselspannung wird von D1 gleichgerichtet und durch C1 geglättet. Es spricht natürlich nichts dagegen, den Wert von C1 zu erhöhen, um die Schaltung unempfindlicher gegen Rauschen und Störungen zu machen. Eine 15-V-Zener-Diode (ZD1) schützt vor Überspannungen.

Das 12-V-Wechselspannungs-Netzteil, das

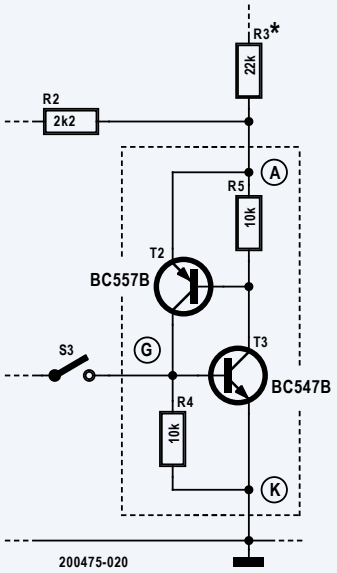


Bild 3. Der Thyristor kann durch ein PNP/NPN-Transistorpaar ersetzt werden.

an der Leistungsschaltung angeschlossen ist, war früher ein Gleichspannungsnetzteil, bis sein Gleichrichter starb, so dass es heute nichts weiter ist als ein Transformator mit Isolierung der Klasse II. Das ist löblich hinsichtlich des Recycling-Gedankens, aber Sie können natürlich auch ein Steckernetzteil mit Gleichspannungsausgang verwenden. In diesem Fall würde D1 als Schutz vor Verpolung fungieren.

Ein Thyristor mit Gedächtnis

Ein Druck auf den grünen Ein-Tasters S3 (ein Schließer – NO) schaltet den Thyristor TRI1 in den leitenden Zustand, so dass die Basis des Darlington-Transistors T1 auf 0 V gezogen wird und dieser Transistor ebenfalls leitet. Über die LED LD1 fließt ein Strom, der von R4 im Leistungsteil auf etwa 10 mA begrenzt wird, durch die LED im Optokoppler IC1. Der Optokoppler schaltet den Triac TRI2 ein, der seinerseits das Relais im Schütz aktiviert.
Der Zustand des Thyristors TRI1 bleibt auch dann erhalten, wenn man den Taster S3 loslässt, und zwar solange der Laststrom durch R2 fließen kann. Um den Zustand zu beenden, kann man entweder einen negativen Impuls an das Gate legen oder, wie es hier geschieht, mit dem roten Aus-Taster S1 (ein Öffner – NC) kurz die Spannungsversorgung des Thyristors unterbrechen. TRI1 blockiert sofort, wodurch auch T1 sperrt, kein Strom zu IC1 fließt, der Triac TRI2 sperrt, wodurch letztendlich das Schütz deaktiviert wird.
Der Optokoppler IC1 schaltet übrigens lobenswerterweise nur im Nulldurchgang der Netzwechselspannung, wenn der Strom gleich Null ist. Die beiden Widerstände R5 und R6 haben die im Datenblatt empfohlenen Werte.

Keine Snubber-Schaltung

Die Verwendung eines Triacs mit einer induktiven Last wie einem Motor oder einem Relais erfordert normalerweise eine Schutzschaltung, die in der Regel aus einem RC-Netzwerk, dem so genannten Snubber-Glied, zwischen den beiden Anoden des Triacs besteht. Dies ist hier aber nicht erforderlich, da es sich beim TRI2 um einen so genannten

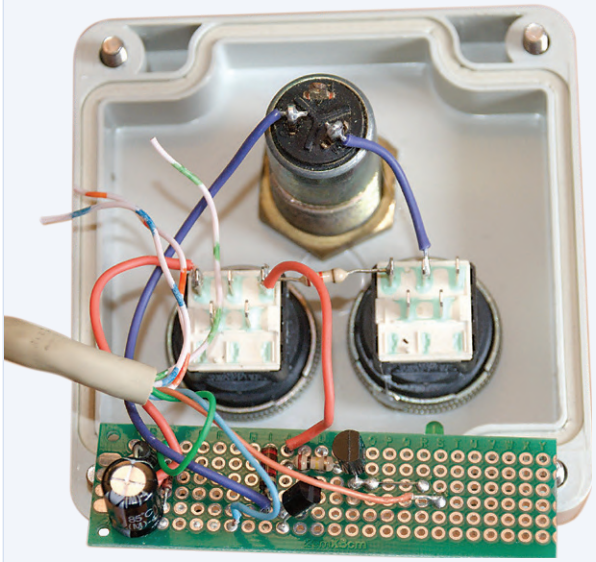


Bild 4. Das Innere der Schaltbox. Die kleine Platine kann neben den Drucktasten mit einem Tropfen Heißkleber befestigt werden.

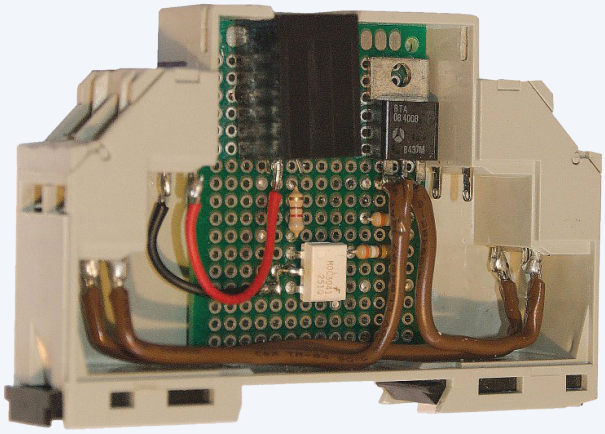


Bild 5. Die Leistungsstufe ist im DIN-Hutschienengehäuse. Das Schwarze ist die RJ11-Buchse, an der die Steuerelektronik angeschlossen wird.

Snubberless-Typ handelt.
Dem geschulten Auge dürfte das Fehlen von Sicherungen und Schutzschaltern aufgefallen sein. In der Realität ist dem Eingang der Leistungsstufe ein 30-mA-Differenzstrom-Schutzschalter, dem Ausgang Schützes ein Leitungsschutzschalter (aka Sicherungsautomat) nachgeschaltet.

Transistoren statt Thyristor

Ich schlage auch eine zweite Version meines Projekts vor, die nicht einen Thyristor TRI1, sondern stattdessen ein Paar PNP/NPN-Transistoren (BC557B und BC547B) verwendet. Zusammen mit zwei 10-kΩ-Widerständen bilden diese beiden Transistoren einen Pseudo-Thyristor (**Bild 3**).
Beim Einschalten sperren T2 und T3 (genau wie der Darlington T1 natürlich). Es fließt kein Strom zum Optokoppler IC1. Wenn der Ein-Taster gedrückt wird, beginnt T3 zu leiten und zieht die Basis von T2 fast auf 0 V. Dann beginnt auch T2 zu leiten und legt eine positive Spannung an die Basis von T3, so dass beide Transistoren auch beim Loslassen des Tasters durchgeschaltet bleiben. Der Strom wird durch R2 und R5 begrenzt. Wie zuvor fließt ein Strom von etwa 10 mA über T1 zum Optokoppler, so dass der Triac ebenfalls leitet. Das Ausschalten erfolgt wieder durch Drücken des Aus-Tasters.
Diese zweite Schaltung funktioniert genauso wie die erste mit dem echten Thyristor. Ich persönlich bevorzuge den Thyristor wegen des einfachen Aufbaus. Andererseits kann man bei der zweiten Version auf den Darlington-Transistor für T1 verzichten und ihn durch einen BC557B ersetzen. Verringern Sie in diesem Fall den Wert seines Basiswiderstandes auf 22 kΩ. Ich habe einen BC516 verwendet, weil ich den Beutel mit BC557s in meinen Schubladen nicht finden konnte, obwohl dort eigentlich eine ziemliche Ordnung herrscht.

PASSENDE PRODUKTE

- **PeakTech 2240 AC Power Source (SKU 19315)**
  www.elektor.de/19315
- **Fumetractor with LED Light (SKU 19092)**
  www.elektor.de/19092

Aufbau

Wie bereits erwähnt, habe ich zwei Gehäuse für das System verwendet.

- In dem fast quadratischen Polycarbonat-Gehäuse mit den Abmessungen 82 x 80 x 55 mm (Bopla M210, siehe **Bild 4**) befinden sich die beiden Drucktasten und der Schlüsselschalter zusammen mit der Niederspannungs-Steuerelektronik. Der Schaltkasten passt gut unter einen meiner beiden Labortische. Die Drucktasten sind von jedem der beiden Tische aus jederzeit gut zugänglich.
- Ein 17,5 mm breites DIN-Schienen-Gehäuse (1 TE) zur Aufnahme der Leistungsstufe einschließlich des Optokopplers und des Triacs (**Bild 5**). Es stellt auch die 12-V-Stromversorgung für die Steuerbox über die RJ11-Buchse bereit. Wegen des geringen Stroms habe ich keinen Kühlkörper für den Triac verwenden müssen.

Aufgrund der wenigen Bauteile in beiden Teilschaltungen habe ich keine Platinen entworfen, sondern die Elektronik auf Lochpunktplatinen aufgebaut, die ich dann mit Heißkleber in ihren jeweiligen Gehäusen fixiert habe. Denken Sie immer daran, dass Sie am 230-V-Netz arbeiten, also seien Sie bitte sehr vorsichtig, die Anschlüsse des Triacs liegen recht nahe beieinander! Die Materialliste kann von [2] heruntergeladen werden.

200475-02

Ein Beitrag von

Idee, Design, Text und Illustrationen: **Philippe Le Guen**
Schaltpläne: **Patrick Wielders**
Redaktion: **Clemens Valens**
Übersetzung: **Rolf Gerstendorf**
Layout: **Giel Dols**

Haben Sie Fragen oder Kommentare?

Haben Sie technische Fragen oder Kommentare zu diesem Artikel? Kontaktieren Sie Elektor unter redaktion@elektor.de.

WEBLINKS

[1] Beispiele von Schlüsselschaltern: https://www.mouser.com/datasheet/2/140/KO-345961.pdf
[2] Konstruktionsdateien, Datenblätter und Stückliste: https://www.elektormagazine.de/labs/4209
[3] Website des Autors: https://www.pleguen.fr/index.php/securiser-sa-paillasse

Teil 02

Programmieren in Python

Nickname-Generator mit grafischer Benutzeroberfläche.

MAKER

Laura Sach

Laura leitet das A-Level-Team bei der *Raspberry Pi Foundation* und erstellt Ressourcen für Schüler zum Thema Computerwissenschaften.

@CodeBoom

MAKER

Martin O'Hanlon

Martin arbeitet im Lernteam der *Raspberry Pi Foundation*, wo er Onlinekurse, Projekte und Lernressourcen erstellt.

@martinohanlon

In Teil 1 dieses Tutorials haben Sie gelernt, wie Sie Ihre grafische Oberfläche mit einer Vielzahl von Optionen ausstatten können. Jetzt ist es Zeit, in den wirklich interaktiven Teil einzusteigen und eine Anwendung zu erstellen, die auf Ihre Eingaben reagiert. Wie wäre es zum Beispiel mit einem Button, der beim Anklicken einen Nickname erzeugt? Solche Phantasienamen werden häufig verlangt, wenn Sie sich irgendwo im Internet anmelden müssen – und oft fällt einem ausgerechnet dann kein brauchbarer Name ein.

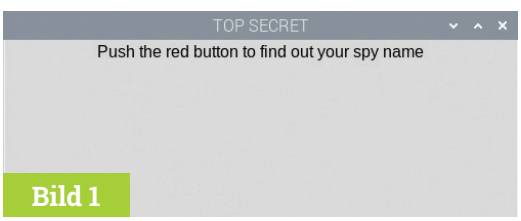
Zum Programmieren der entsprechenden Software muss man nur ein einfaches Fenster erstellen und etwas Text hinzufügen. Der hier gezeigte Code enthält auch einige Kommentare; Zeilen, die mit einem # beginnen und den Umgang mit dem Programm vereinfachen.

Führen Sie den folgen Code aus. Sie sollten das in **Bild 1** gezeigte Fenster sehen

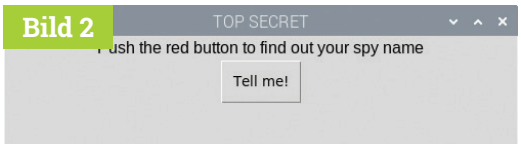
```
# Imports ---------------
from guizero import App, Text

# Functions -------------

# App -------------------
app = App("TOP SECRET")

# Widgets ---------------
title = Text(app, "Push the red button to find out your spy name")

# Display ---------------
app.display()
```

Schaltfläche hinzufügen

Bringen wir nun eine Schaltfläche (Button) auf unserer grafischen Oberfläche (GUI) an. Fügen Sie dazu ***PushButton*** zu Ihrer Liste der Importe hinzu. Schreiben Sie unterhalb des Text-Widgets, aber vor der Anzeige der App, folgende Codezeile:



▶ **Bild 1** Anzeige des Textes in einem Fenster.



▲ **Bild 2** Sie haben jetzt eine Schaltfläche programmiert.

```
button = PushButton(app, choose_name, text="Tell me!")
```

Ihr Code sollte nun demjenigen im Programm **spy1.py** entsprechen. Beim Ausführen wird jedoch noch keine Schaltfläche erscheinen, sondern nur eine Fehlermeldung im Shell-Fenster:

```
NameError: name 'choose_name' is not defined
```

Das liegt daran, dass `choose_name` eine Art Befehl zum Aufruf einer Funktion mit gleichem Namen ist, die ausgeführt wird, wenn die Schaltfläche gedrückt wird. Diese Funktion existiert jedoch noch nicht. Die meisten GUI-Komponenten können mit einem Befehl verknüpft werden. Für eine Schaltfläche bedeutet dies: "Wenn die Schaltfläche gedrückt wird, führe diesen Befehl aus."

Erstellen Sie eine Funktion

Lassen Sie uns daher nun die Funktion `choose_name` schreiben, damit Ihre Schaltfläche etwas bewirken kann, wenn sie gedrückt wird.

Schauen Sie sich Ihr Programm an und suchen Sie den Abschnitt ***functions***. Hier sollten alle Funktionen stehen, die an GUI-Widgets angehängt werden, um sie vom Code für die Anzeige des Widgets getrennt zu halten. Fügen Sie folgenden Code in den Funktionsabschnitt ein:

```
def choose_name():
    print("Button was pressed")
```

Ihr Code sollte nun wie der Inhalt der Datei **spy2.py** aussehen. Wenn Sie die nun erscheinende Schaltfläche anklicken (**Bild 2**), sieht es so aus, als sei nichts passiert, aber in Ihrem Ausgabefenster (**Bild 3**) ist ein Text erschienen.

Für einen ersten Test genügt ein beliebiger Dummy-Text. Sie können dann die Anweisung zur Ausgabe des Textes später durch den eigentlichen Code für die Aufgabe ersetzen, die Ihre Schaltfläche ausführen soll.

Geben Sie innerhalb Ihrer Funktion `choose_name` ein #-Symbol vor der Codezeile ein, die den Text "Button was pressed" ausgibt. Programmierer nennen dies "Auskommentieren". Sie sagen dem Computer damit, dass er diese Codezeile wie einen Kommentar behandeln soll, oder mit anderen Worten, sie ignorieren soll. Der Vorteil dieser Methode: Wenn Sie den Code jemals wieder verwenden wollen, können Sie ihn einfach wieder in Ihr Programm einbauen, indem Sie das #-Symbol entfernen.

spy1.py

LADEN SIE DEN KOMPLETTEN CODE HERUNTER:
magpi.cc/guizerocode

Sprache: **Python 3**

```
001.  # Imports ---------------
002.  from guizero import App, Text, PushButton
003.
004.  # Functions -------------
005.
006.  # App -------------------
007.  app = App("TOP SECRET")
008.
009.  # Widgets ---------------
010.  title = Text(app, "Push the red button to find out your spy name")
011.  button = PushButton(app, choose_name, text="Tell me!")
012.
013.  # Display ---------------
014.  app.display()
```

spy2.py

Sprache: **Python 3**

```
001.  # Imports ---------------
002.  from guizero import App, Text, PushButton
003.
004.  # Functions -------------
005.
006.  # App -------------------
007.  app = App("TOP SECRET")
008.
009.  # Widgets ---------------
010.  title = Text(app, "Push the red button to find out your spy name")
011.  button = PushButton(app, choose_name, text="Tell me!")
012.
013.  # Display ---------------
014.  app.display()
```

Große rote Schaltfläche

Im Moment ist Ihre Schaltfläche weder groß noch Rot! Sie haben im ersten Teil dieser Tutorial-Serie Eigenschaften verwendet, um das Aussehen Ihres Textes auf dem "Wanted"-Poster zu ändern. Können Sie nun auch die Eigenschaften des *PushButton*-Widgets verwenden, um die Hintergrundfarbe und die Textgröße zu ändern?

Beachten Sie, dass es unter Umständen nicht möglich ist, die Farbe einer Schaltfläche unter macOS zu ändern, da einige Versionen des Betriebssystems dies nicht zulassen, aber Sie sollten auf jeden Fall in der Lage sein, die Textgröße zu ändern.

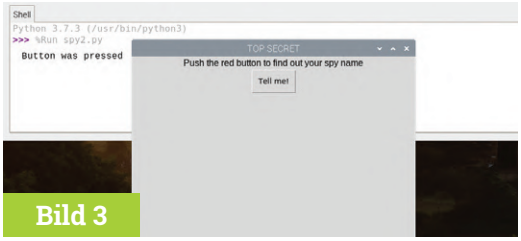


Bild 3 Der Text wird im Shell-Fenster ausgegeben.

Hinzufügen von Namen

Fügen Sie in einer neuen Zeile eine Liste mit Vornamen ein. Sie können die Namen in Ihrer Liste frei wählen und es können so viele Namen sein, wie Sie möchten, aber stellen Sie sicher, dass jeder Name zwischen Anführungszeichen steht und die Namen jeweils durch ein Komma getrennt sind. Eine Sammlung von Buchstaben, Zahlen und/oder Interpunktionszeichen zwischen Anführungszeichen wird als String bezeichnet. Jeder Name muss also ein String sein.

```
first_names = ["Barbara", "Woody",
"Tiberius", "Smokey", "Jennifer", "Ruby"]
```

Fügen Sie nun auch eine beliebig lange Liste mit frei gewählten, phantasievollen Nachnamen hinzu:

```
last_names = ["Spindleshanks", "Mysterioso",
"Dungeon", "Catseye", "Darkmeyer",
"Flamingobreath"]
```

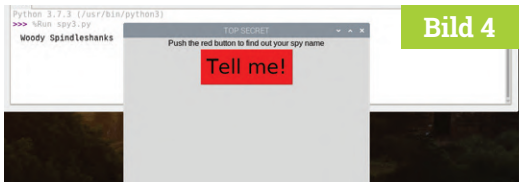
Nun brauchen wir eine Möglichkeit, um einen zufälligen Namen aus jeder Liste auszuwählen, um einen aus Vor- und Nachname bestehenden Nickname zu erzeugen. Ihre erste Aufgabe besteht darin, eine neue Import-Zeile in Ihrem Import-Abschnitt hinzuzufügen:

```
from random import choice
```

Damit teilen Sie dem Programm mit, dass Sie eine Funktion namens ***choice*** verwenden möchten, die ein zufälliges Element aus einer Liste auswählt. Jemand hat bereits den Code geschrieben, der dies für Sie erledigt, und er steht in Python zu Ihrer Verfügung.

Fügen Sie in Ihrem Code für die Funktion `choose_name` direkt unter den Namenslisten eine Codezeile ein, die den Vornamen Ihres Nickname auswählt und ihn dann mit dem Nachnamen verkettet, mit einem Leerzeichen dazwischen. Verketten bedeutet, dass zwei Zeichenketten miteinander verbunden werden. Das Symbol (nicht nur) in Python für Verkettung ist ein Plus (+).

```
spy_name = choice(first_names) + " " +
choice(last_names)
```



▲ **Bild 4** Ausgeben eines Nickname.

```
print(spy_name)
```

Ihr Code sollte nun demjenigen **spy3.py** ähneln. Speichern Sie ihn und führen Sie ihn aus. Wenn Sie die Taste drücken, sollten Sie sehen, dass ein zufällig generierter Name in Ihrer Konsole oder Shell erscheint, und zwar an der gleichen Stelle, an der vorher die ursprüngliche Meldung "Button was pressed" erschien (**Bild 4**).

Name erscheint auf GUI

Das ist gut, aber wäre es nicht schöner, wenn der Nickname auch auf der grafischen Oberfläche erscheinen würde? Lassen Sie uns ein weiteres Text-Widget erstellen und es zur Anzeige des Namens verwenden.

Fügen Sie im Bereich ***Widgets*** ein neues Text-Widget hinzu, das den Namen anzeigen soll:

```
name = Text(app, text="")
```

Natürlich soll der Text nicht schon erscheinen, bevor der Button angeklickt wird. Also können Sie den Text auf `""` setzen, was als 'leerer String' bezeichnet wird. Kommentieren Sie innerhalb Ihrer Funktion `choose_name` die Codezeile aus, in der Sie den Nickname ausgeben.

Fügen Sie nun eine neue Codezeile am Ende der Funktion ein, um den Wert des ***name*** Text-Widgets auf den gerade erstellten `spy_name` zu setzen. Dies bewirkt, dass das Text-Widget sich selbst aktualisiert und den Namen anzeigt.

```
name.value = spy_name
```

Ihr endgültiger Code sollte wie der Code der Datei ***03-spy-name-chooser.py*** aussehen. Führen Sie das Programm aus und drücken Sie die Schaltfläche, um Ihren per Zufall ermittelten Nickname auf der grafischen Benutzeroberfläche angezeigt zu bekommen (**Bild 5**).

Wenn Ihnen der ermittelte Name nicht gefällt, können Sie den Vorgang beliebig oft wiederholen und sich immer wieder neue Namen vorschlagen lassen.

▲ **Bild 5** Das fertige Nickname-Wahlmenü.

spy3.py

Sprache: **Python 3**

```
001.  # Imports ---------------
002.  from guizero import App, Text, PushButton
003.  from random import choice
004.
005.  # Functions -------------
006.  def choose_name():
007.      #print("Button was pressed")
008.      first_names = ["Barbara", "Woody", "Tiberius", "Smokey",
      "Jennifer", "Ruby"]
009.      last_names = ["Spindleshanks", "Mysterioso", "Dungeon",
      "Catseye", "Darkmeyer", "Flamingobreath"]
010.      spy_name = choice(first_names) + " " + choice(last_names)
011.      print(spy_name)
012.
013.  # App -------------------
014.  app = App("TOP SECRET")
015.
016.  # Widgets ---------------
017.  title = Text(app, "Push the red button to find out your spy name")
018.  button = PushButton(app, choose_name, text="Tell me!")
019.  button.bg = "red"
020.  button.text_size = 30
021.
022.  # Display ---------------
023.  app.display()
```

03-spy-name-chooser.py

Sprache: **Python 3**

```
001.  # Imports ---------------
002.
003.  from guizero import App, Text, PushButton
004.  from random import choice
005.
006.  # Functions -------------
007.
008.  def choose_name():
009.      #print("Button was pressed")
010.      first_names = ["Barbara", "Woody", "Tiberius", "Smokey",
      "Jennifer", "Ruby"]
011.      last_names = ["Spindleshanks", "Mysterioso", "Dungeon",
      "Catseye", "Darkmeyer", "Flamingobreath"]
012.      spy_name = choice(first_names) + " " + choice(last_names)
013.      #print(spy_name)
014.      name.value = spy_name
015.
016.  # App -------------------
017.
018.  app = App("TOP SECRET")
019.
020.  # Widgets ---------------
021.
022.  title = Text(app, "Push the red button to find out your spy name")
023.  button = PushButton(app, choose_name, text="Tell me!")
024.  button.bg = "red"
025.  button.text_size = 30
026.  name = Text(app, text="")
027.
028.  # Display ---------------
029.
030.  app.display()
```

Anmerkung: Dieser Artikel erschien zuerst im MagPi Magazin. Elektor gibt MagPi in niederländischer, französischer und deutscher Sprache heraus.

JETZT ABONNIEREN UND SIE ERHALTEN

WILLKOMMENSGESCHENK

ALLE 2 MONATE, EINE NEUE AUSGABE MIT DEN BESTEN ÜBER RASPBERRY PI

NUR 54,95 € PRO JAHR (6 AUSGABEN)

Mit einem MagPi Abo erhalten Sie:

- 6 gedruckte Ausgaben der MagPi
- ein Willkommensgeschenk
- Zugang zu dem Online Archiv mit allen MagPi Ausgaben seit 2016

Ihre Vorteile:

- 6 gedruckte Ausgaben der MagPi
- ein Willkommensgeschenk
- Zugang zu dem Online Archiv mit allen MagPi Ausgaben seit 2016
- Günstiger als Einzelkauf
- Jede Ausgabe direkt zu Ihnen nach Hause
- Alle Ausgaben auch als PDF
- Willkommensgeschenk mit tollen Inhalten

JETZT ABONNIEREN: WWW.MAGPI.DE

productronica
fast forward
the start-up platform
powered by Elektor

productronica Fast Forward 2021: Die Preisträger

Von **Clemens Valens** (Elektor)

Der Start-up-Wettbewerb *productronica Fast Forward Award 2021* war eine rein virtuelle Veranstaltung, doch die Qualität der Kandidaten war wie immer sehr hoch. Hier sind die Gewinner!

Da lange unklar war, ob die productronica 2021 als reale Messe stattfinden würde oder nicht, blieb dem Fast Forward Award nur der Weg über das Internet. Deshalb mussten die Kandidaten aus Ländern wie Ungarn, Deutschland, Israel und Frankreich Präsentationen ihrer Projekte aufzeichnen, anstatt sie vor einer Live-Jury und einem Publikum vorzustellen. Die Videos ergänzende Informationen konnten in Form von Business-Plänen und Pitch-Decks angefügt werden.

Nach der Sichtung der aufgezeichneten Elevator-Pitches und dem Studium der Unterlagen wählte die Jury, bestehend aus Benjamin Klingenberg (NCAB-Group, Platinum-Sponsor der Veranstaltung) sowie Elektor-Ingenieuren und -Redakteuren, drei Gewinner aus:

ERSTER PREIS

50.000 € Elektor-Marketingbudget

ioTech Group mit Sitz in Jerusalem, Israel: ioTech entwickelt neue Techniken für die additive Serienfertigung und den Multimaterialdruck.

www.elektormagazine.com/news/iotech-group-productronica-fast-forward-2021

ZWEITER PREIS

25.000 € Elektor-Marketingbudget

Reedu GmbH & Co. KG aus Münster, Deutschland: Reengineering Education alias Reedu alias re:edu ist bekannt durch das Bildungs-Toolkit senseBox und die Umweltdatenplattform openSenseMap.

www.elektormagazine.com/news/reedu-gmbh-co-kg-productronica-fast-forward-2021

DRITTER PREIS

15.000 € Elektor-Marketingbudget

Pozi Development Ltd aus Budapest, Ungarn: Pozi sorgt für volle Transparenz in Produktion und Logistik. Pozi identifiziert und eliminiert Verschwendung und unnötigen Abfall in der gesamten Lieferkette.

www.elektormagazine.com/news/pozi-development-ltd-productronica-fast-forward-2021

Die drei siegreichen Start-ups wurden von der Jury aufgrund ihres hohen kommerziellen Potenzials und der Qualität der Teams ausgewählt - zwei wichtige Kriterien, damit sich ein Start-up zu einem erfolgreichen Unternehmen entwickeln kann. Herzlichen Glückwunsch den Gewinnern!

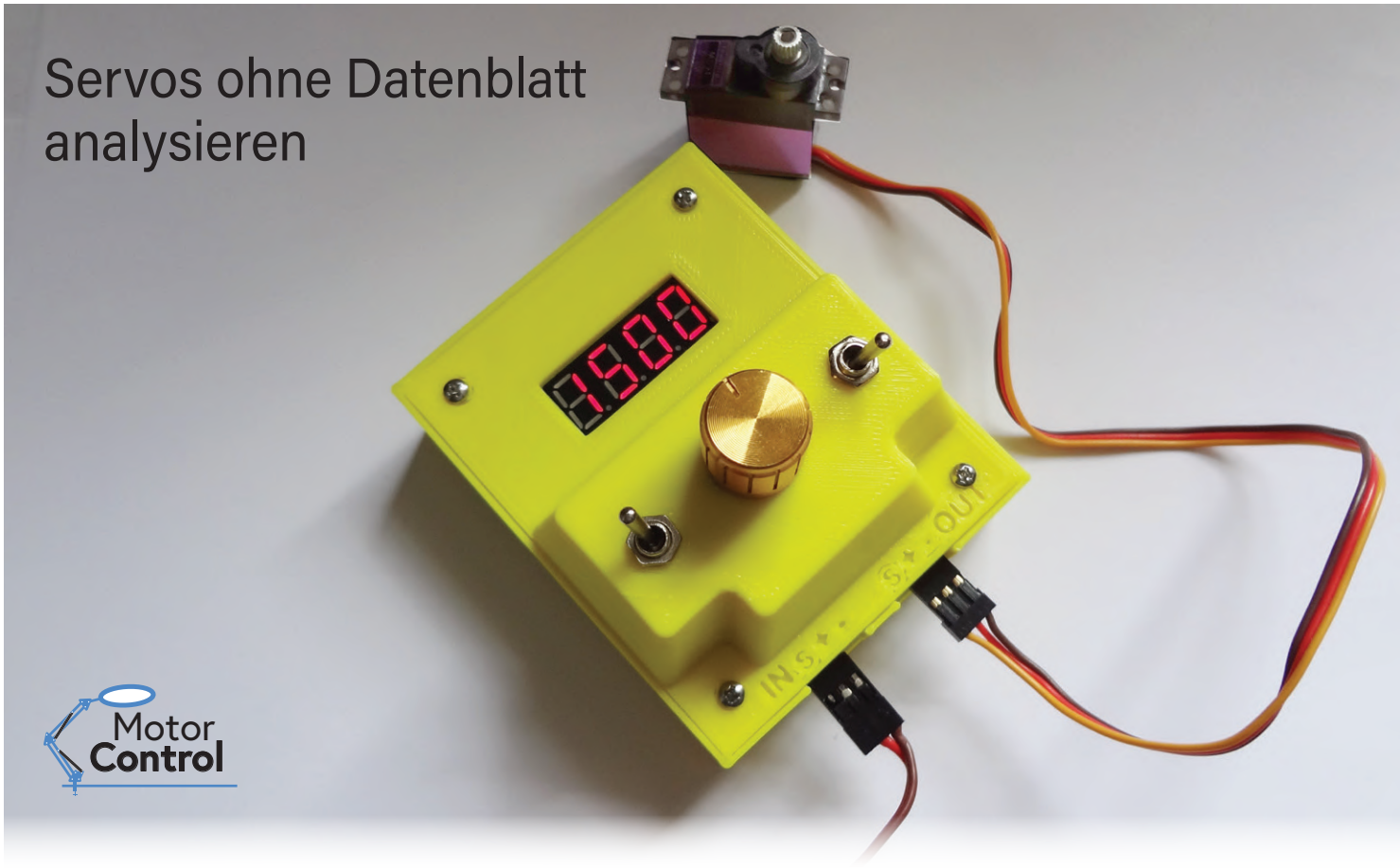
Sehen Sie sich die pFFWD-Zeremonie an!

Auf zur electronica 2022?

Der Fast Forward Award soll auch auf der electronica 2022 in München im November dieses Jahres verliehen werden. Hoffentlich ist bis dahin alles wieder beim Alten und wir sehen uns dort persönlich!

210658-02

Vielseitiger **Servo-Tester**



Von **Marcelo Maggi** (USA)

Es gibt immer mehr verschiedene Servos, die sich in Preis, Qualität und Leistung unterscheiden. Die Kenntnis des genauen Verhaltens eines Servos, insbesondere wenn das Datenblatt nicht verfügbar ist, ist deshalb für die Funktion und Leistungsfähigkeit der Endanwendung von grundlegender Bedeutung.

Die allgemeine Definition des Begriffs „Servo" beschreibt ein Gerät, das zur Steuerung eines gewünschten Vorgangs mittels Rückkopplung verwendet wird. In der Elektronik, insbesondere bei elektromechanischen Aktuatoren, ist ein Servo ein Gerät, das eine mechanische Bewegung erzeugt, die durch einen elektrischen Impuls gesteuert wird.

Beispiele für solche Servos gibt es in sehr verschiedenen Bereichen, deshalb möchte ich mich auf solche konzentrieren, die in funkgesteuerten Modellen und für Hobbys im Allgemeinen sehr beliebt und weit verbreitet sind. Es gibt verschiedene Arten von Servos, analoge und digitale, mit Kunststoff- oder Metallzahnrädern, mit oder ohne Kugellager .... Alle haben jedoch eines gemeinsam: Sie können unabhängig

Eigenschaften und Spezifikationen

Betriebsarten: manuell (niedrige und hohe Auflösung) und automatisch (niedrige und hohe Geschwindigkeit)
Minimale Impulsbreite: per Software einstellbar
Maximale Impulsbreite: per Software einstellbar
Mittenposition: manuell einstellbar
Minimale Schritt/Impulsauflösung: 1 µs
Maximaler Schritt: per Software einstellbar.
Signalperiode: fest auf 20 ms
Impulsstabilität/Genauigkeit: quarzgesteuert
Spannungsversorgung: 4,5...6,0 V

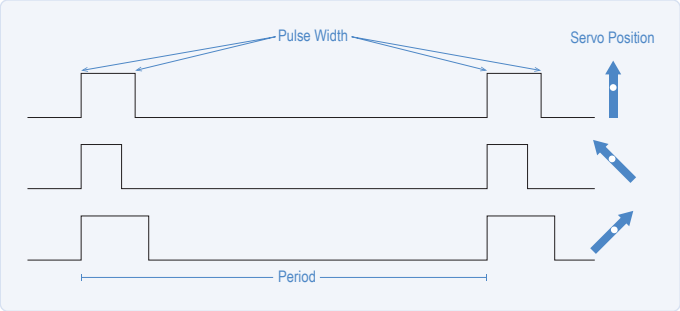


Bild 1. Servo-Steuerimpulse.

von der jeweiligen Anwendung mit demselben elektrischen Signal angesteuert werden.

Dieses Signal besteht aus Impulsen mit variabler Breite, aber fester Periodenlänge (**Bild 1**), die in der Regel 20 Millisekunden (ms) beträgt. Die Impulsbreite ist das, was die Servodrehung effektiv steuert. Die Mittelposition ist bei den meisten Servos auf 1500 Mikrosekunden (µs), die Extrempositionen auf bis zu 500 µs am unteren und 2500 µs am oberen Ende eingestellt. In der Regel entsprechen diese Extrempositionen einer Drehung von -90 Grad und +90 Grad in Bezug auf die Mittelstellung bei 0. Die Drehung kann je nach Hersteller im oder gegen den Uhrzeigersinn erfolgen.

Die Amplitude der Impulse (in Volt) entspricht in der Regel der Stromversorgung des Servos und liegt üblicherweise bei 4,8 V oder 6,0 V, obwohl es heutzutage auch Servos mit Amplituden von 3,7 V bis zu 14 V und mehr gibt.

Warum ein Servo-Tester?

Zu einem Servo gehört auch das Datenblatt des Herstellers mit allen Spezifikationen wie maximale Auslenkung, Geschwindigkeit, Totzone und so weiter. Bei vielen preiswerten Servos oder bei gebrauchten Exemplaren ist manchmal jedoch kein Datenblatt verfügbar, oder eines, in dem wichtige Angaben fehlen oder eines, dessen Leistungsangaben in der Realität keinen Bestand haben. In diesen Fällen bringt nur das Testen der tatsächlichen Leistung eines Servos die Wahrheit ans Licht.

Entwurf eines Servotesters

Ein einfacher Servotester ist, so zeigen es zahlreiche Beispiele im Internet, auch sehr einfach zu bauen. Es genügt, den sehr beliebten Timer 555 und ein paar andere Bauteile zusammenzulöten, um einen analogen Servotester zu realisieren. Wenn Sie jedoch das Verhalten des Servos präzise messen und eine visuelle Anzeige des Steuerimpulses wollen, ist ein etwas komplexerer Ansatz erforderlich. Ein solches Testgerät könnte die Impulsbreite präzise einstellen und hätte ein Display, das diesen Wert anzeigt. Günstige Versionen im Internet

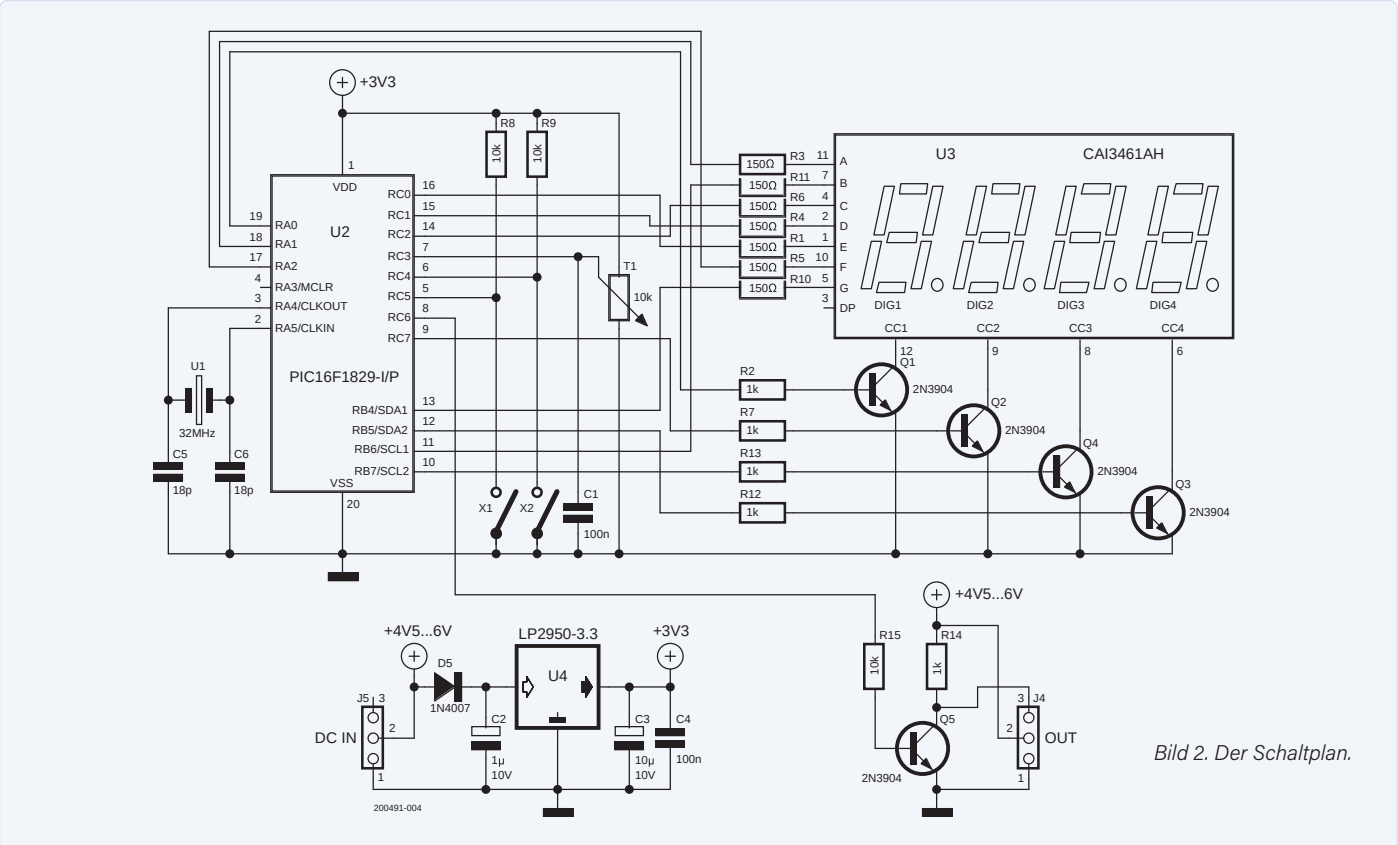


Bild 2. Der Schaltplan.

haben aber den Nachteil, dass die meisten Parameter bereits vorprogrammiert sind und vom Benutzer nicht geändert werden können. Wäre es nicht toll, einen vollständig anpassbaren Servotester zu haben, bei dem der Benutzer die volle Kontrolle über alle Parameter hat, um das gewünschte Signal zu erzeugen, einschließlich manueller und automatischer Testfunktionen?
Und ob! Genau dies bietet dieser Servotester: ein vollständig programmierbares Gerät, das dem Benutzer die Kontrolle über alle Signalparameter ermöglicht, mit manuellen und automatischen Modi, mit einer klaren und intuitiven Benutzeroberfläche.

Hardware

Die vollständige Schaltung ist in **Bild 2** dargestellt. Unten links ist der Anschluss (J5 - DC IN) für die Stromversorgung (durch den Regler des RC-Modells oder eine beliebige andere Stromversorgung) zu sehen. Der Tester benötigt weniger als 15 mA, so dass er keine große Belastung für die Stromversorgung darstellt. Die Spannung kann zwischen 4,5 V und 6,0 V liegen, je nach den Anforderungen des Servos.
Bitte beachten Sie: Der Servo erhält die volle Spannung, die von der Stromquelle geliefert wird. Achten Sie darauf, dass Sie den maximal zulässigen Wert des Servos nicht überschreiten! Bei den meisten Herstellern ist der mittlere Anschluss +V, während GND und SIGNAL auf den beiden äußeren Anschlüssen liegen. Der Stromeingang benötigt natürlich nur die beiden Leitungen +V und GND – das Signal wollen wir ja selber erzeugen - , dennoch wird für J5 eine 3-polige Buchse verwendet, damit der bei den meisten Reglern verwendete Standardstecker passt.
Auf der rechten Seite des Schaltplans befindet sich unten der Servoausgang (J4 - OUT), an dem die drei Leitungen (+V, GND und jetzt auch SIGNAL) für den zu testenden Servo beeitgestellt werden.
Rechts oben befindet sich U3, die 4-stellige 7-Segment-Anzeige mit den dafür erforderlichen Bauteilen. Die Anzeige wird vom Mikrocontroller (U2) angesteuert. Schließlich gibt es noch die „Benutzerschnittstelle", die aus einem einzelnen Potentiometer (T1) und den beiden Schaltern X1 und X2 (1 x an) besteht. Durch Drehen des Potentiometers wird die Impulsdauer verändert, während die Schalter die vier Betriebsmodi wie in der folgenden Tabelle auswählen. Die verschiedenen Modi werden im Abschnitt über die Software erläutert.

SW1	SW2	Betriebsart
0	0	Auto, hohe Geschwindigkeit
0	1	Auto, niedrige Geschwindigkeit
1	0	Manuell, hohe Auflösung
1	1	Manuell, niedrige Auflösung

Platinenlayout

Diese Hardware kann zwar auf einem Breadboard montiert werden, aber der Aufbau auf einer richtig gelayouteten Platine wie in **Bild 3** macht das Gerät robuster und zuverlässiger. Der mit NI Multisim erstellte Entwurf und die Gerberdateien, mit denen Sie die Platine bei Ihrem bevorzugten Lieferanten bestellen können, stehen unter [1] zum Download bereit. Bitte beachten Sie, dass die Kondensatoren der Oszillatorschaltung des Mikrocontrollers in diesem Platinenentwurf fehlen, aber offenbar funktioniert der Servotester des Autors auch ohne C5 und C6 einwandfrei. Die Kondensatoren können aber doch einfach auf der Kupfer-Seite (Unterseite) der Platine angelötet werden, nachdem man den eventuell vorhandenen Lötstopplack von der Massefläche abgekratzt hat. Sie können aber natürlich auch die Designdateien herunterladen und die beiden Kondensatoren dem Platinenlayout hinzufügen.

STÜCKLISTE

Widerstände:
R1,R3,R4,R5,R6,R10,R11 = 150 Ω
R2,R7,R12,R13,R14 = 1 k
R8,R9,R15 = 10 k
T1 = Poti 10 k, linear

Kondensatoren:
C1,C4 = 100 n
C2 = 1 µ/10 V radial
C3 = 10 µ/10 V radial
C5,C6 = 18 p

Halbleiter:
D5 = 1N4007
Q1,Q2,Q3,Q4,Q5 = 2N3904
U2 = PIC16F1829-I/P
U3 = 4-stellige 7-Segment-LED-Anzeige mit gemeinsamer Kathode (3461AH)
U4 = 3,3 V LDO (LP2950-3,3)

Außerdem:
U1 = Quarz 32 MHz
X1,X2 = Schalter 1x an (SPST)

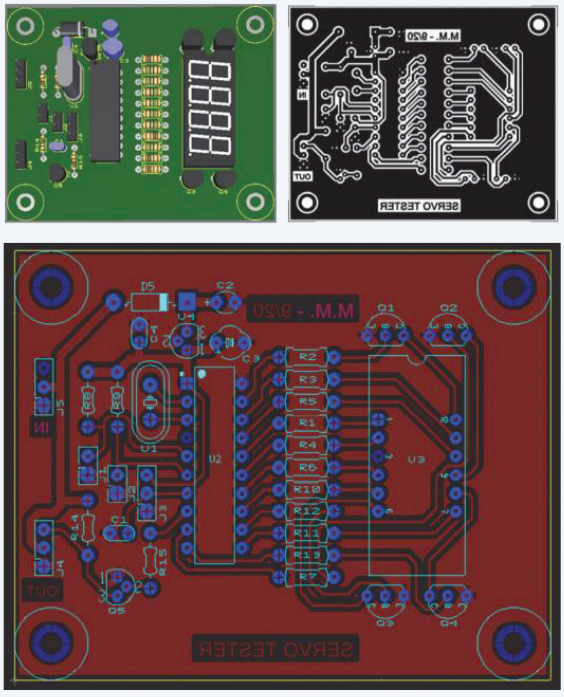


Bild 3. Bestückte Platine mit den Anschlüssen für das Poti (J3) und die Schalter (J1 und J2).

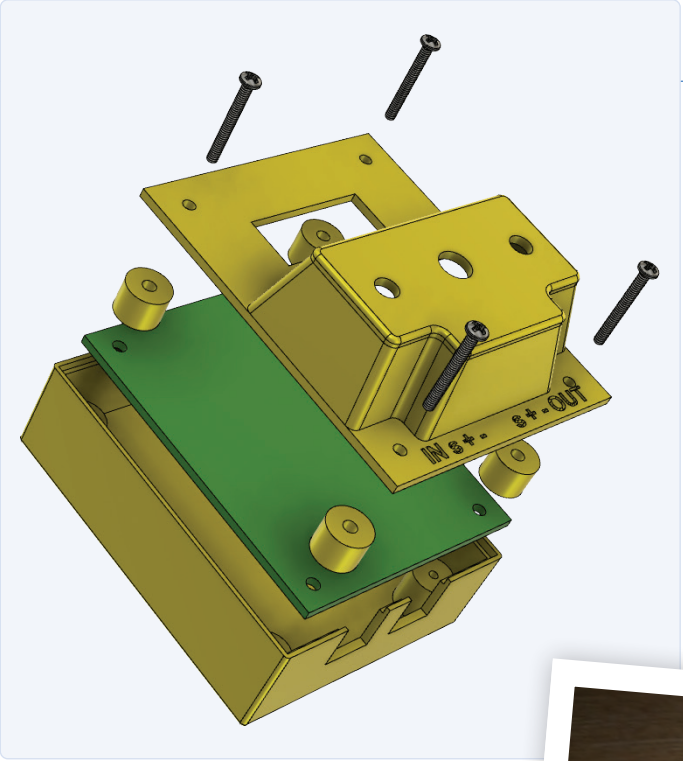


Bild 4. 3D-Entwurf des Gehäuses.

- MAX: maximale Impulsbreite. Dies ist die längste Dauer des Impulses sowohl im manuellen als auch im automatischen Modus. Obwohl auch hier jede Zahl eingestellt werden kann, sollte man nicht höher als 2500 gehen, da das Servo möglicherweise nicht in der Lage ist, ein solches Extrem zu erreichen.

Diese vier Parameter verleihen dem Testgerät eine enorme Vielseitigkeit, wie sie in keinem billigen kommerziellen Gerät zu finden ist. Wenn Sie den Mikrocontroller in eine Fassung einsetzen, können Sie ihn leicht heraushebeln und neu programmieren und haben dann jederzeit einen neuen Tester zur Hand. Wenn Sie mit einem Parametersatz jedoch zufrieden sind, können Sie das gleiche Programm für immer beibehalten.

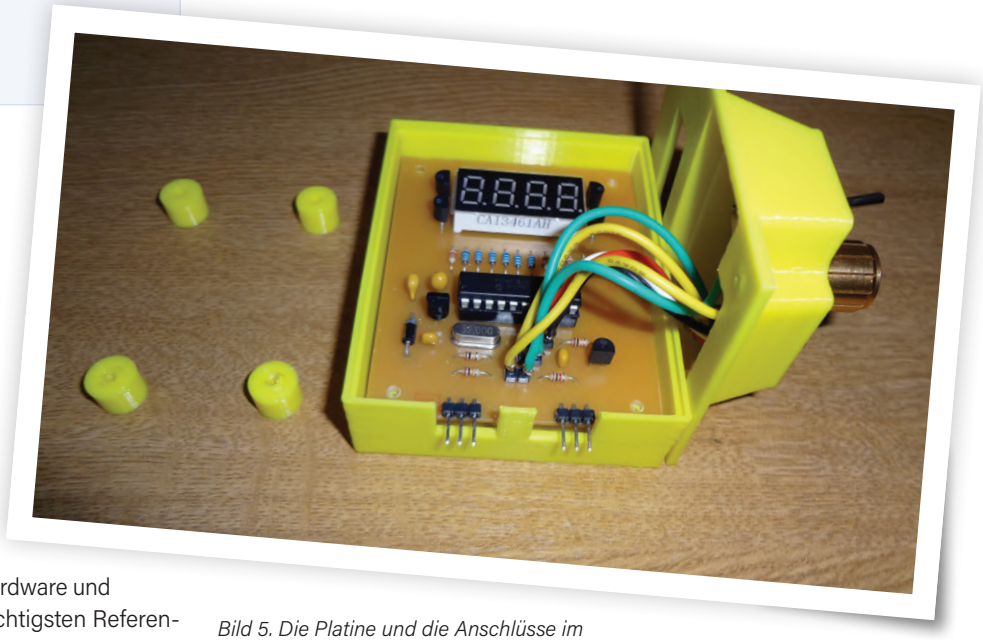


Bild 5. Die Platine und die Anschlüsse im Inneren des 3D-gedruckten Gehäuses.

Software

Der wohl wichtigste Teil des Servotesters ist die Software, in der die Präzision und die Vielseitigkeit stecken. Das Programm läuft in einem PIC16F1829 mit einem 32-MHz-Quarz als Hauptoszillator, ist in C geschrieben und verwendet den CCS-C-Compiler. Der Quellcode und die HEX-Datei zur Programmierung des Mikrocontrollers stehen unter [1] zum Download bereit.

Das Programm beginnt mit den Definitionen der Hardware und ihren Anschlüssen sowie den Angaben zu den wichtigsten Referenzwerten: Die Buchstaben `a` bis `g` stehen für die sieben Segmente der numerischen Anzeige, während `d1` bis `d4` den vier Digits entsprechen. `PWM` ist das Ausgangssignal zur Steuerung des Servos und `SW1` und `SW2` sind die Schalter. Das Potentiometer ist mit dem Analogeingang AN7 (PIN_C3) verbunden und wird in der `main`-Funktion eingestellt. Am Ende stehen vier sehr wichtige Definitionen (alle Werte in Mikrosekunden):

- BRI: Helligkeit der LED-Anzeige. Diese Zahl steuert die Nachleuchtdauer aller Digits, also ihre scheinbare Helligkeit. Sie ist auf den Wert 50 voreingestellt, kann aber von 1...200 geändert werden.
- SPD: Geschwindigkeit der Servobewegung im Automodus. Diese Zahl gibt die Schrittweite beim Erhöhen/Verringern der Impulsbreite an: Je höher die Zahl, desto schneller bewegt sich der Servo. Jede Zahl im Bereich 1...100 ist möglich.
- MIN: minimale Impulsbreite. Dies ist die kürzeste Dauer des Impulses sowohl im manuellen als auch im automatischen Modus. Obwohl jede Zahl eingestellt werden kann, sollte man nicht unter 500 gehen, da das Servo möglicherweise nicht in der Lage ist, solche extremen Werte zu erreichen.

Jede Ziffer ist vollständig in einer Funktion (`num`) definiert und kann mit einer `switch`-Anweisung ausgewählt werden. Jedes Mal, wenn diese Funktion mit einer in `n` gespeicherten Zahl aufgerufen wird, wird die Zahl auf dem Display angezeigt. Um auszuwählen, welche der vier Digits zu einem bestimmten Zeitpunkt aktiv ist (da die Anzeige sequentiell angesteuert wird), enthält die Funktion `digit` die Auswahl, ebenfalls innerhalb einer `switch`-Anweisung. Immer wenn sie aufgerufen wird, wird die in `z` gespeicherte Zahl angezeigt. Wenn `z` den Wert „5“ enthält, sind alle Ziffern ausgeschaltet, damit ein sauberen Übergang von einer Ziffer zur nächsten erreicht ohne sichtbare Artefakte zu sehen ist.

Die Funktion `display` bringt Ziffernauswahl und Ziffernausgabe zusammen. Diese Funktion nimmt eine vierstellige Zahl, zerlegt sie in die einzelnen Ziffern und zeigt sie von rechts nach links (Einheiten zuerst) nacheinander auf dem Display an, indem sie die beiden vorherigen Funktionen aufruft. Hier wird auch die Helligkeit der Anzeige bestimmt, indem man die Verzögerung vor dem Löschen einer Ziffer festlegt.

Das genaue Timing der Periode und der Impulsdauer wird durch den TIMER1-Interrupt erreicht. Da es sich um einen 16-Bit-Zähler handelt, läuft er bei 65536 über und erzeugt damit den Interrupt. Durch Vorladen von TIMER1 mit einem Offset-Wert kann eine genaue Zeit gemessen werden. Beim ersten Aufruf dieses INT wird TIMER1 mit dem für die Impulsdauer erforderlichen Wert geladen, beim nächsten Mal ist es der Rest einer gesamten Periode. Dieser Vorgang wird kontinuierlich wiederholt (gesteuert von der Variablen `pulse`), so dass man eine präzise Impulszeit erreicht, während die konstante Periodenzeit von 20 ms eingehalten wird.

An dieser Stelle wird im Programm auch der Status der Schalter überprüft, um festzustellen, ob der automatische Modus gewählt ist (SW1 = 0). Ist dies der Fall, bestimmt SW2 die Geschwindigkeit des automatischen Sweeps (von MIN zu MAX und wieder zurück zu MIN): SW2 = 1 bedeutet normale Geschwindigkeit (der Impuls ändert sich um 1 µs) und SW2 = 0 steht für hohe Geschwindigkeit (der Impuls ändert sich jeweils um SPD µs). Sie erinnern sich, dass SPD eine beliebige Zahl im Bereich 1...100 sein kann? Wenn SPD auf 1 eingestellt ist, entspricht die hohe Geschwindigkeit der normalen (was nicht sehr sinnvoll ist). `main` fasst schließlich alles zusammen. Neben der Initialisierung des Analog-Digital-Wandlers (ADC), von TIMER1 und der Interrupts befindet sich hier die Hauptprogrammschleife, die den Wert des ADC liest (also die Potistellung) und die Impulsdauer entsprechend einstellt.

Wenn wir uns im manuellen Modus befinden (SW1 = 1), dann wird SW2 ausgewertet. Bei einer „1" ist der Modus *niedrige Auflösung* aktiv und das Potentiometer kann den gesamten Bereich der Impulsbreite (MIN - MAX) durchlaufen. Dieser Modus heißt nicht ohne Grund so: Der ADC kann nur 1024 verschiedene Pegel erfassen, aber der Pulsbreitenbereich kann 1400 µs oder mehr betragen, so dass es nicht möglich ist, eine Auflösung von 1 µs innerhalb des gesamten Bereichs zu erreichen. Wenn SW2 = 0 ist, dann ist der *hochauflösende Modus* aktiv. Hier ist der Pulsbreitenbereich auf 1469...1531 µs begrenzt. Mit dem Poti kann leicht eine Pulsbreitenänderung von 1 µs vorgenommen werden, was äußerst nützlich ist, um den Servo exakt in Mittelstellung (1500 µs) zu bringen und die Totzeit zu ermitteln, also die Zeit (in Mikrosekunden), die verstreicht, bis sich auf einen Impuls hin der Servo bewegt.

Professionelles Äußeres

Mit einem schönen Gehäuse, das die Hardware im Inneren schützt, sieht dieses Gerät ziemlich professionell aus. Das einfache 3D-gedruckte Gehäuse in **Bild 4** sollte diese Aufgabe erfüllen. Die Designdatei steht unter [1] zum Download bereit. Die Platine wird zwischen vier zylindrischen Abstandshaltern und der unteren Gehäuseschale eingeklemmt und die obere Gehäuseschale wird mit vier Kreuzschlitzschräubchen (M2,5 x 20 mm mit Linsenkopf) befestigt. Das zusammengebaute Gerät ist im geöffneten Zustand in **Bild 5** zu sehen.

200491-02

Ein Beitrag von

Text: **Marcelo Maggi**
Illustrationen: **Marcelo Maggi, Patrick Wielders**
Redaktion: **Luc Lemmens**
Übersetzung: **Rolf Gerstendorf**
Layout: **Giel Dols**

Haben Sie Fragen oder Kommentare?

Haben Sie technische Fragen oder Kommentare zu diesem Artikel? Schicken Sie eine E-Mail an die Redaktion unter luc.lemmens@elektor.com oder kontaktieren Sie Elektor unter redaktion@elektor.de.

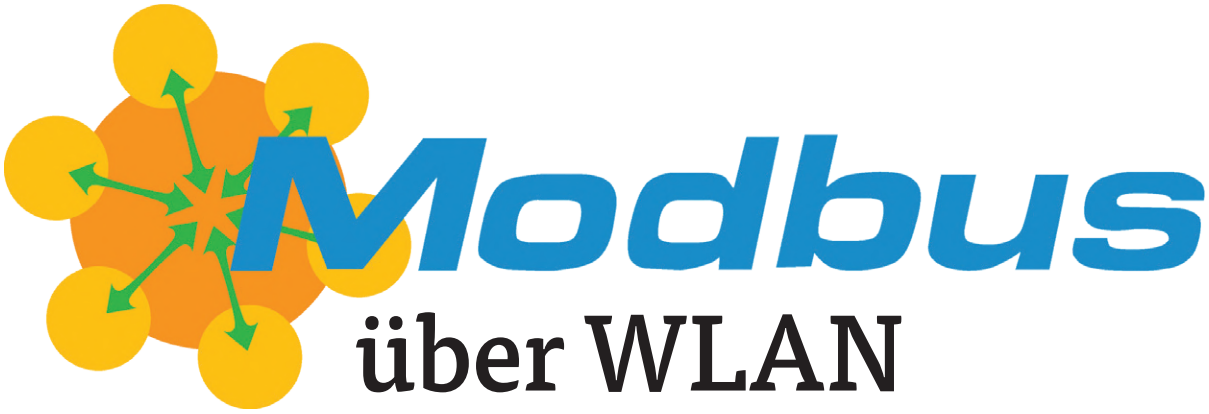
PRODUKTE

› **FeeTech Mini 360 Degree Continuous Rotation Servo FS90R (SKU 19784)**
www.elektor.de/19784

› **FeeTech FS90MG Metal Geared Servo with Accessories (SKU 19788)**
www.elektor.de/19788

WEBLINK

[1] Software, Platinenlayout und 3D-Entwurf zum Herunterladen: https://www.elektormagazine.de/200491-02



über WLAN

Teil 2: Software für das Modbus-TCP-WLAN-Modul

Von **Josef Bernhardt** und **Martin Mohr**

Im ersten Teil [1] haben wir ins mit der Hardware des Projekts beschäftigt. Kern des Moduls ist ein NodeMCU-Board mit dem Mikrocontroller ESP8266; eine zusätzliche Basis-Platine stellt industriekompatible Anschlüsse bereit. Nach einer kurzen Einführung in die Funktionsweise des Modbus-Protokolls gehen wir nun auf die Ansteuerung des Moduls ein und erfahren, welche Software man dazu verwendet. In dem Beispiel kommt wieder die Garagentorsteuerung aus dem ersten Teil zum Einsatz.

Modbus ist ein Protokoll, das in der Automationstechnik weit verbreitet ist. Es arbeitet nach dem Master/Slave-Prinzip. Der Busmaster kann bis zu 246 Slaves ansteuern. Die Adressen 1...247 können für Busteilnehmer verwendet werden. Die Adresse 0 ist die Broadcast-Adresse: Alle Daten, die an diese Adresse gesendet werden, werden von allen Teilnehmern empfangen. Die Konsistenz der einzelnen Datenpakete wird über CRC-Checksummen gewährleistet. Die Slaves besitzen interne Register mit unterschiedlichen Funktionen. **Tabelle 1** gewährt einen Überblick über die Modbus-Funktionen.

Das Modbus-Protokoll unterscheidet drei Varianten:

- Modbus RTU: binäre Übertragung über RS485 (EIA485).
- Modbus ASCII: Klartext-Übertragung über RS485 (EIA485), weniger effizient als binäre Übertragung, dafür von Menschen lesbar. Man kann die Kommandos mit einem einfachen Terminal versenden.
- Modbus TCP: Hier werden die Modbus-Kommandos mit TCP/IP übertragen, meistens über Ethernet oder wie bei unserem Modul über WLAN.

Detaillierte Informationen und Protokollbeschreibungen gibt es auf der Modbus-Webseite [2].

Vorbereitung des Modbus-WLAN-Moduls

Damit wir das WLAN-Modul auch mit dem Modbus-Protokoll verwenden können, muss als Erstes die dazu nötige Firmware geladen werden. Bitte ziehen Sie dazu das NodeMCU-Board von der Modbus-Platine ab und schließen es an den USB-Anschluss Ihres PCs an. Nun starten Sie die Arduino-IDE, die wir im ersten Teil eingerichtet haben. Die Firmware, die das Modul in einen Modbus-Client verwandelt, kann von der Elektor-Projektseite [3] heruntergeladen werden.

In der Arduino-IDE öffnet man die heruntergeladene Datei (*Datei -> Öffnen...*) *OpenPLC_ESP8266_1_0_MUX_V1_1.ino*. Die Arduino-IDE fragt nun, ob das Projekt in den Sketch-Bereich verschoben werden kann, was man bejahen sollte. Falls die IDE dabei ein neues Verzeichnis anlegt, kopieren Sie auch die Headerdatei (*modbus.h*) dorthin, wo auch die *ino*-Datei liegt, damit der Header beim Kompilieren gefunden wird. Damit Sie im WLAN auf das Modul zugreifen können, müssen dessen Zugangsdaten am Anfang des Sourcecodes angepasst werden (**Listing 1**). Tragen Sie hier die Daten Ihres eigenen WLAN ein.

Nachdem diese Anpassungen durchgeführt sind, können Sie das Programm wie im ersten Teil beschrieben auf das Modbus-WLAN-Modul laden. Das Modul arbeitet mit DHCP, erhält also automatisch eine IP-Adresse von Ihrem Router (die in der Router-Firmware nachgesehen werden kann). Alternativ dazu können Sie sich die Ausgabe des Programms im Seriellen Monitor der Arduino-IDE (Lupen-Icon rechts oben) ansehen. Die Geschwindigkeit der Schnittstelle muss unten rechts in der Dropdown-Box auf 115200 Baud eingestellt werden.

Tabelle 1. Übersicht der Modbus-Funktionen.

Beschreibung	Modus	Bits
Einzelner Ein-/Ausgang „Coil“	Lesen und Schreiben	1
Einzelner Eingang „Discrete Input“	nur Lesen	1
(analoge) Eingänge „Input Register“	nur Lesen	16
(analoge) Ein-/Ausgänge „Holding Register“	Lesen und Schreiben	16

Wie die Ausgabe der IP-Adresse im Seriellen Monitor aussehen kann, zeigt **Listing 2**. Damit Sie diese Prozedur nicht bei jedem Einschalten des Modbus-Moduls wiederholen müssen, sollten Sie eine statische IP-Adresse vergeben, was alle modernen Router ermöglichen. Wie auch immer, notieren sie die ausgegebene IP-Adresse des Moduls, damit Sie über WLAN darauf zugreifen können. Falls Sie keine Ausgabe sehen, betätigen Sie den Reset-Taster auf dem NodeMCU-Board, um die Software neu zu starten, denn die IP-Adresse wird nur während des Neustarts ausgegeben.

Listing 1. WLAN-Konfiguration des Modbus-Moduls.

```
/*********NETWORK CONFIGURATION*********/

const char *ssid = "<YOUR_SSID>";
const char *password = "<YOUR_PASSWORD>";

/*****************************************/
```

Listing 2. IP-Ausgabe des Modbus-WLAN-Moduls.

```
Connecting to Vodafone-3980
.......
WiFi connected
Server started
My IP: 192.168.0.85
```

Erster Test

Um zu überprüfen, ob das Modbus-WLAN-Modul einwandfrei arbeitet, verwenden wir das Tool EasyModbus, das zum Download auf SourceForge bereit steht [4]. EasyModbus stellt uns einen Server, einen Client und eine Bibliothek bereit. Wir benötigen für unseren Test nur den Client, der direkt unter [5] heruntergeladen werden. Um das Tool zu verwenden, benötigen Sie eine installierte Java-Version auf Ihrem PC. Falls Java noch nicht installiert ist, können Sie das freie OpenJDK verwenden [6], das sich in der Entwickler- und Maker-Community einer großen Beliebtheit erfreut und in allen Linux-Distributionen zum Standard gehört. Es wird beispielsweise unter Ubuntu mit dem Terminal-Befehl `sudo apt install openjdk-11-jdk` installiert. Um den EasyModbus-Client zu starten, gibt man den Befehl `java -jar EasyModbusJavaClient.jar` ein. **Bild 1** zeigt die Ausgabe des EasyModbus-Clients (links unter Linux, rechts unter Windows). Wählen Sie hier *Modbus TCP* und geben die IP-Adresse Ihres Moduls ein. Belassen Sie die *Starting Address* auf *1* und ändern Sie die *Number of Values* auf *4*. Wenn Sie nun auf *Read Discrete Inputs – FC 2* klicken, werden die vier digitalen Eingänge des Moduls gelesen und angezeigt. Es ist sinnvoll, wenn dabei zumindest ein Eingang mit einem Signal belegt ist, damit Sie sehen können, ob alles erwartungsgemäß funktioniert. Der EasyModbus-Client kann nur Daten vom Modbus lesen, nicht aber schreiben. Das Schreiben testen wir mit dem Beispielprogramm, das wir gleich erstellen. Im unteren Bereich des EasyModbus-Clients sind die real übertragenen Daten zu sehen. Diese Funktion ist für das Debuggen extrem hilfreich. Wenn der erste kleine Test erfolgreich war, können wir dazu übergehen, das Modul mit Python zu programmieren.

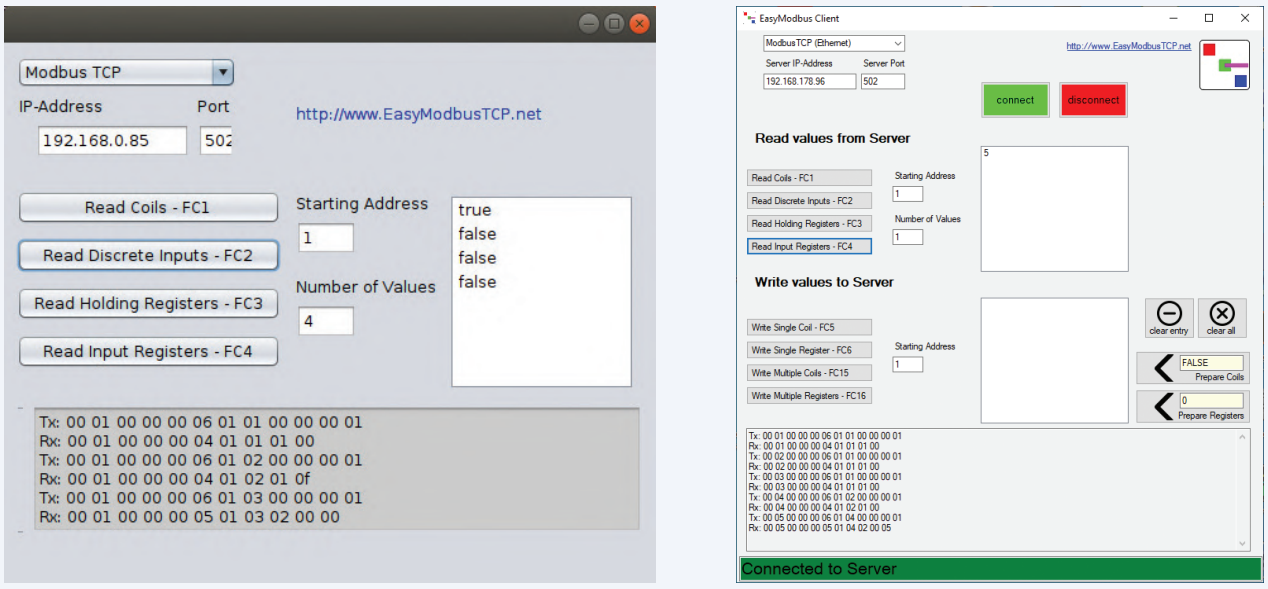


Bild 1. Der EasyModbus-Client liest die Daten aus dem Modul aus. Links ist die Linux-, rechts die Windows-Variante zu sehen.

Listing 3. Beispiel-Programm tor.py zur Torsteuerung.

```
from pyModbusTCP.client import ModbusClient
client = ModbusClient(host="192.168.0.85",
   port=502, auto_open=True, debug=False)
while(True):
    inputs=client.read_discrete_inputs(0,4)
    endschalter_oben  = inputs[0]
    endschalter_unten = inputs[1]
    taster_runter     = inputs[2]
    taster_hoch       = inputs[3]
    motor_hoch        = 0
    motor_runter      = 1
    # Abfrage Endschalter
    if(endschalter_oben):
        client.write_single_coil(motor_hoch,False)
        print("Tor offen")
    if(endschalter_unten):
        client.write_single_coil(motor_runter,False)
        print("Tor geschlossen")
    # Abfrage Taster
    if (taster_hoch and not endschalter_oben):
        client.write_single_coil(motor_hoch,True)
    if (taster_runter and not endschalter_unten):
        client.write_single_coil(motor_runter,True)
```

Modbus-Bibliothek installieren

Um den Test etwas praxisnäher zu gestalten, verwenden wir wieder unser Garagentormodell [7] aus dem ersten Teil. Wir schreiben ein Programm für den PC, das die Stellung der Taster und Endschalter per Modbus empfängt, um entsprechende Motor-Aktionen (wiederum per Modbus) zu übermitteln.

Unser Python-Testprogramm benötigt eine Bibliothek [8], um auf den Modbus zuzugreifen. Das ist nichts Besonderes; es gibt Modbus-Bibliotheken für fast alle Programmiersprachen. Im Zweifel könnte man sogar die einzelnen Bytefolgen der Kommandos selbst erzeugen und über das Netzwerk übertragen. Das würde aber hier zu weit führen.

Zunächst sollte man sich vergewissern, dass der Python-Paketinstaller Pip an Bord ist (was bei neueren Python-Installationen stets der Fall sein sollte). Man überprüft dies mit

```
$ python -m pip --version
C:\> py -m pip --version
```

(oben Linux, unten Windows). Sollte kein Pip-Installer vorhanden sein, installiert man ihn über Python mit `get-pip.py` [9] (und nicht über den Linux-Paketmanager) nach.

```
wget https://bootstrap.pypa.io/get-pip.py
python get-pip.py
get-pip.py
```

Dann wird die Modbus-Bibliothek installiert:

```
$ python -m pip install -U pymodbusTCP
C:\> py -m pip install -U pymodbusTCP
```

Eine ausführliche Dokumentation der Bibliothek mit vielen Codebeispielen finden Sie unter [10].

Beispielprogramm

Sehen wir uns nun unser Beispielprogramm in **Listing 3** im Detail an. Mit der ersten Zeile wird die Modbus-Bibliothek importiert. In der nächsten Codezeile wird ein neues Verbindungsobjekt erstellt, um mit einem konkreten Modbus-Gerät zu kommunizieren. Es können hierbei mehrere Parameter übergeben werden, in unserem Beispiel die IP-Adresse des Zielgerätes als `host`. Der Port 502 ist der Standard-Port für die Modbus-TCP-Kommunikation. Mit dem Parameter `auto_open` bestimmen wir, ob die Verbindung automatisch oder manuell hergestellt werden soll. In unserem Beispiel nutzen wir aus Bequemlichkeit die automatische Variante, aber wenn Sie den Verbindungsaufbau komplett selbst steuern wollen, wählen Sie bitte `False`. Beim manuellen Verbindungsaufbau haben Sie mehr Möglichkeiten der Steuerung und zur Fehlerbehandlung.

Wird der Parameter `debug` auf `True` gesetzt, gibt die Bibliothek alle übertragenen Daten auf der Konsole aus. Das ist extrem hilfreich, wenn man auf der Suche nach einem Fehler ist. Mit Hilfe einer `while`-Endlosschleife laufen wir immer wieder durch das Programm, wie es auch im ersten Teil beschrieben wurde. Das Programm arbeitet also zyklisch wie alle Industriesteuerungen.

Der Block innerhalb der Endlosschleife ist das eigentliche Programm, in dem zuerst die Eingänge gelesen und dann die Eingangswerte in „sprechende" Variablen überführt werden. Dann definieren wir auch für die zwei digitalen Ausgänge Variablen. So sind die folgenden Programmzeilen leichter zu lesen.

Sie können dort erkennen, dass der Motor abgeschaltet wird, wenn die Endstellungen erreicht sind. Um am Rechner zu sehen, in welcher Stellung sich das Tor gerade befindet, wird ein Text auf der Konsole ausgegeben. Zu guter Letzt wird der Motor in der entsprechenden Drehrichtung gestartet, wenn die Up/Down-Taste gedrückt wird. Die UND-Verknüpfungen mit den Endschaltern verhindern, dass die Relais anfangen zu flattern, wenn sich das Tor schon in der Endlage befindet.

Um das Programm zu starten, geben Sie auf der Kommandozeile `python tor.py` ein. Das Programm wird wie gewohnt mit *Strg+C* beendet.

AdvancedHMI

Eine komfortable Fernsteuerung mit visueller Rückmeldung, die man an eigene Zwecke anpassen kann, lässt sich mit verschiedenen PC-Frameworks verwirklichen. Die AdvancedHMI-Software ist eine offene Software zum Erstellen von HMI-Anwendungen (Human Machine Interface), die mit Ihrer SPS oder einem anderen E/A-Gerät kommunizieren. Die Software unterscheidet sich von anderen Standardpaketen dahingehend, dass Sie ausführbare Dateien erstellen und nicht nur Konfigurationen, die von einer Laufzeit-Engine interpretiert werden. Das Ergebnis ist eine extrem schnelle und effiziente Anwendung.

AdvancedHMI [11] basiert auf dem Microsoft .NET Framework. Anwendungen werden mit dem kostenlosen *Microsoft Visual Studio*

Community 2019 erstellt. Dadurch können Sie per Drag&Drop grundlegende HMIs erstellen, ohne eine einzige Codezeile schreiben zu müssen. Durch die Nutzung des .NET-Frameworks sind Support-Netzwerke und Entwickler reichlich vorhanden. Sie finden für AdvancedHMI weit mehr Unterstützung als für alle anderen Standard-HMI-Pakete zusammen.
Der Autor hat sich mit der Plattform eine Oberfläche zur Steuerung der Basisplatine generiert. Hier wird der Status der Eingänge angezeigt, mit Buttons lassen sich die Ausgänge schalten (**Bild 2**).
Mit Mono laufen .NET-Programme auch unter Linux - so auch die Oberfläche zur Steuerung der Basisplatine. Ein Test auf dem Raspberry Pi verlief erfolgreich.

Fazit

Mit dem Modbus-WLAN-Modul kann man einfach von einem PC aus oder auch mit dem Smartphone Geräte ansteuern, die dem Modbus-Industriestandard entsprechen. Auch wenn Sie sich in das Modbus-Protokoll einfuchsen wollen, ist dieses Modul ein gutes Testobjekt. Die Hardware ist jedoch auch so robust und fehlertolerant aufgebaut, dass sie sie auch in eigenen ESP8266-Projekten verwenden können, wenn Sie Industriegeräte wie Magnetventile, Motoren und andere Arten von Sensoren/Aktoren ansteuern und auslesen wollen.

200507-B-02

Bild 2. Die AdvancedHMI-Software erzeugt Mensch-Maschine-Anwendungen.

Sie haben Fragen oder Kommentare?

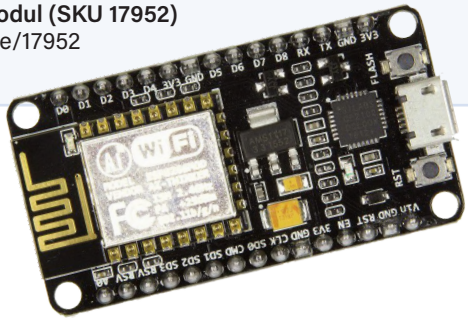
Haben Sie technische Fragen oder Kommentare? Bitte wenden Sie sich per E-Mail an den Autor unter josef@bernhardt.de oder an die Elektor-Redaktion unter editor@elektor.com.

Ein Beitrag von

Idee und Ausführung: **Josef Bernhardt** und **Martin Mohr**
Redaktion: **Rolf Gerstendorf**
Layout: **Giel Dols**

PASSENDE PRODUKTE

› **NodeMCU-Modul (SKU 17952)**
www.elektor.de/17952



WEBLINKS

[1] Modbus über WLAN, Teil 1, ElektorMag 9-10/2021: http://www.elektormagazine.de/200507-02
[2] Modbus-Webseite: https://modbus.org/
[3] Elektor-Projektseite: https://www.elektormagazine.de/200507-B-02
[4] EasyModbus-Webseite: https://bit.ly/2QDIDJz
[5] Download EasyModbus-Client: https://bit.ly/3d2zzFp
[6] OpenJDK-Download: https://bit.ly/2OVK4m6
[7] Garagentormodell auf YouTube: https://youtu.be/VHIBQswdA0E
[8] Modbus-Bibliothek für Python: https://pypi.org/project/pyModbusTCP/#description
[9] get-pip.py:: https://bootstrap.pypa.io/get-pip.py
[10] Modbus-Bibliothek Beispiele: https://pymodbustcp.readthedocs.io/en/latest/
[11] AdvancedHMI: https://www.advancedhmi.com/index.php?main_page=page&id=14&chapter=0
[12] Weitere Infos des Autors: https://bit.ly/3ch1PFl

Die Neuronen in neuronalen Netzen verstehen

Teil 3: Praktische Neuronen

Von **Stuart Cording** (Elektor)

Bisher haben wir ein neuronales Netz erstellt und ein Verständnis für seine Funktionsweise entwickelt. Wir konnten ihm sogar beibringen, wie die XOR-Funktion funktioniert, wofür die Klassifizierung von Eingabemustern erforderlich ist. In diesem Artikel wollen wir versuchen, einen Teil eines autonomen Fahrsystems zu implementieren: die Erkennung des Zustands von Ampeln.

Es gab schon viele Vorhersagen über selbstfahrende Autos [1], von denen sich keine bewahrheitet hat. Die Schwierigkeit besteht vor allem darin, die Absichten anderer Verkehrsteilnehmer und Fußgänger zu verstehen. Ein großer Teil des autonomen Fahrsystems besteht jedoch darin, alles zu klassifizieren, was die vielen Sensoren und Kameras um das Fahrzeug herum „sehen". Das reicht von Ampeln und Verkehrsschildern über Straßen- oder Fahrbahnmarkierungen bis hin zu Fahrzeugtypen und Personen. Da wir nun ein funktionierendes neuronales Netz in Form unseres mehrschichtigen Perzeptrons haben, werden wir es verwenden, um die Farben einer Ampel zu erkennen.

Eine der großartigen Eigenschaften der bisher verwendeten Processing-Umgebung ist der einfache Zugriff auf Webcams. Die hier verwendeten Beispiele laufen auf so gut wie jedem Laptop oder PC unter Windows, Linux oder macOS mit einer integrierten oder externen Kamera.

Finden Sie die Kamera!

Bevor wir loslegen können, muss eine neue Bibliothek zu Processing hinzugefügt werden, die Zugriff auf alle angeschlossenen Webcams bietet. Wählen Sie im Menü *Sketch -> Import Library ... -> Add Library ...* (**Bild 1**). Wählen Sie im sich öffnenden Fenster (**Bild 2**) die Registerkarte *Libraries* und geben Sie *Video* in das Suchfeld ein. In der daraufhin angezeigten Liste wählen Sie *Video | GStreamer-based video library for Processing* aus und klicken dann auf *Install*.

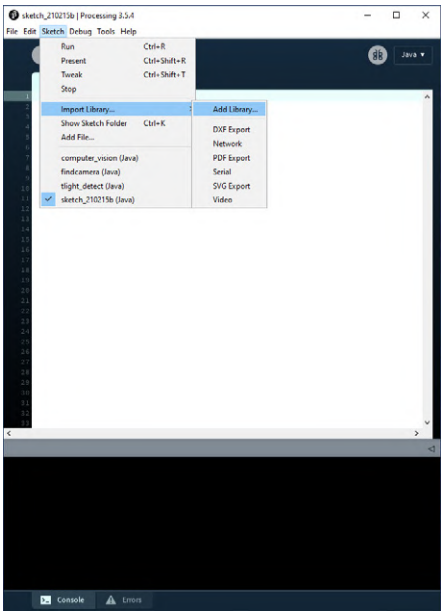


Bild 1. Hinzufügen einer neuen Bibliothek zu Processing.

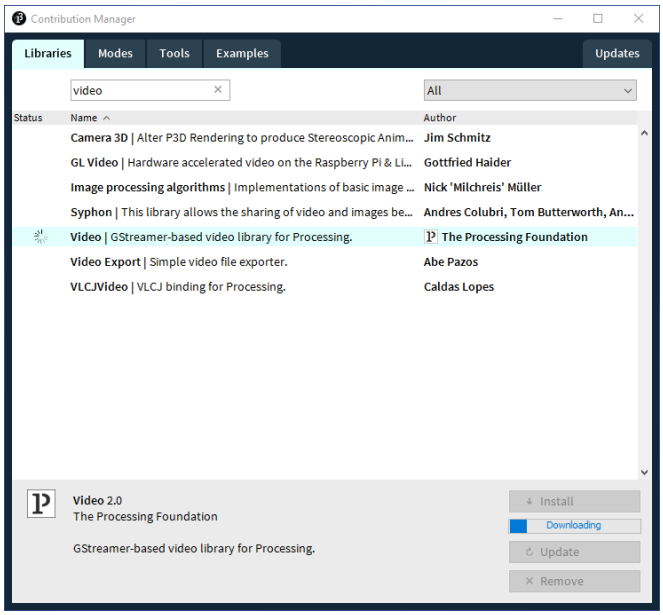


Bild 2. Suchen und Installieren der Videobibliothek.

Wie zuvor werden wir den Code aus dem GitHub-Repository verwenden, der für diese Artikelserie vorbereitet wurde [2].
Sobald die Bibliothek installiert ist, können wir das Projekt */trafficlight/findcamera/findcamera.pde* ausführen. Dieser Code fragt einfach die Liste der verfügbaren Kameras ab, die an den Computer angeschlossen sind, und ermöglicht dem Benutzer die Auswahl einer Kamera durch Eingabe einer Zahl zwischen 0 und 9. Stellen Sie sicher, dass sich der Fokus an der richtigen Stelle im kleinen *findcamera*-Fenster befindet (klicken Sie dazu in das Fenster), bevor Sie eine Zahl drücken. Wenn in der Processing-IDE der Fokus gesetzt ist, geben Sie einfach irgendwo im Quellcode eine Zahl ein.
Sobald eine Kameraquelle ausgewählt wurde, wird ihr Bild in einer niedrigen Auflösung von 320 × 240 Pixeln im Fenster angezeigt (**Bild 3**). In der Textkonsole finden Sie auch den Quellcode, den Sie für die Auswahl Ihrer Kamera für die nächsten beiden Projekte benötigen (**Bild 4**).
Schließlich brauchen wir noch eine Ampel. Da Sie wahrscheinlich keine zur Hand haben, wurde eine in *trafficlight/resources* in verschiedenen Dateiformaten angelegt. Drucken Sie einfach eine aus und halten Sie sie griffbereit.
Wenn Sie die Verarbeitung von Projekten stoppen, die die Kamera verwenden, wird möglicherweise die Meldung „*WARNING: no real random source present!*“ angezeigt. Dies scheint ein Fehler zu sein, der mit der Verwendung der Video-Bibliothek zusammenhängt, aber keinen Einfluss auf die Funktionalität des Codes hat.

Wie sehen Kameras?

Einer der vielleicht wichtigsten Aspekte bei der Verwendung neuronaler Netze ist es, ein Verständnis dafür zu entwickeln, wie ein Computer die eingehenden Daten interpretieren kann. Im Falle der Kameraeingabe wird das Computerbild durch Mischen der drei Grundfarben Rot, Grün und Blau angezeigt. Dies wird gemeinhin als RGB-Format bezeichnet. Die drei Werte variieren zwischen 0 und 255 (oder 0x00 und 0xFF im Hexadezimalformat). Wenn wir die Kamera auf etwas Blaues richten, können wir erwarten, dass der B-Wert relativ hoch und die anderen Werte ziemlich niedrig sind. Richten Sie es auf etwas Gelbes, dann sollte es einen hohen R- und G-Wert haben, da Rot und Grün zusammen Gelb ergeben. Zum besseren Verständnis der additiven Farbmischung dürfte das Projekt */trafficlights/addititive/additive.pde* hilfreich sein (**Bild 5**).
Mit diesem Wissen könnte es sinnvoll sein, zu ermitteln, wie unsere Kamera die Farben unserer Ampel „sieht“ und die von ihr gemeldeten RGB-Werte zu notieren. Dann können wir unserem neuronalen Netz die drei Ampelfarben beibringen, so dass es sie als rot, gelb und grün klassifizieren kann.

Festlegung der RGB-Werte

Das Projekt *trafficlight/computer_vision/computer_vision.pde* ist das nächste Projekt, das wir benötigen. Es zeigt neben dem Bild der ausgewählten Kamera die RGB-Werte an. Mit diesem Projekt können wir also die RGB-Werte aufzeichnen, die die Kamera sieht. Bevor Sie mit dem Code beginnen, vergessen Sie nicht, die Codezeile, die Sie in *findcamera.pde* festgelegt haben, in

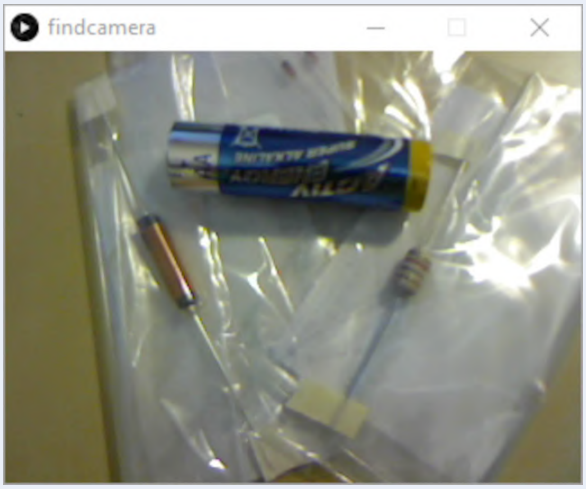


Bild 3. Beispiel für die Ausgabe von findcamera.pde.

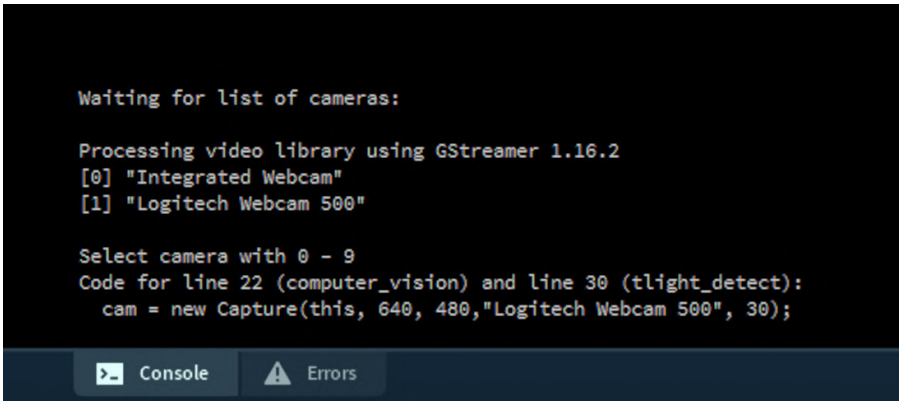


Bild 4. Die Konsolenausgabe von findcamera.pde liefert den Initialisierungscode für die Projekte, um die gewünschte Kamera zu verwenden.

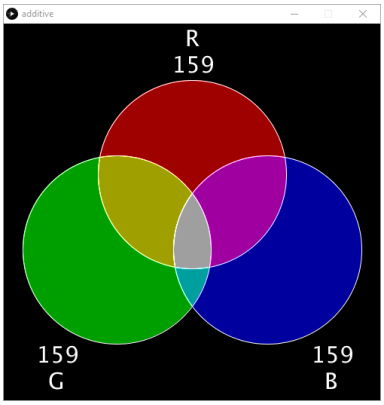


Bild 5. additive.pde demonstriert die additive Farbmischung.

Zeile 22 dieses Projekts einzufügen. Dies stellt sicher, dass die von Ihnen gewählte Kamera verwendet wird.

Während Sie die Kamera auf die rote Ampel richten (halten Sie die „Ampel" ungefähr innerhalb des gestrichelten Kreises), notieren Sie die gesehene Farbe (oben) und die RGB-Werte, die sie definieren (**Bild 6**). Die RGB-Werte beziehen sich auf den Durchschnitt aller Pixel, die die Kamera im roten Begrenzungsquadrat im Zentrum erfasst. Obwohl es sich um einen Durchschnittswert handelt, ändern sich die RGB-Werte schnell nach oben und unten. Zur Erfassung eines einzelnen Wertes drücken Sie „*p*" auf Ihrer Tastatur, und der Code hält bei einem einzelnen Satz von Werten an. Klicken Sie in dieses Fenster, bevor Sie die Taste „*p*" drücken, um dem Code den Fokus mitzuteilen. Durch Drücken von „*s*" werden die Kameraerfassung und die RGB-Generierung erneut gestartet.

Tabelle 1. Erfasste RGB-Werte für die drei Ampelfarben.

Lichtfarbe	R	G	B
Rot	220	56	8
Gelb	216	130	11
Grün	123	150	128

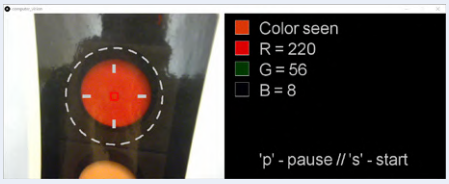


Bild 6. computer_vision.pde liefert die RGB-Werte, die die Kamera für jede Ampelfarbe „sieht".

Bild 7. Die Antwort von tlight_detect.pde vor dem Lernen der Farben.

Der hier verwendete Ausdruck der Ampel wurde laminiert, so dass es an einigen Stellen zu Lichtreflexionen kommt. Wenn Sie Ihr Ampelbild mit einem Laserdrucker ausgedruckt haben, kann es ebenfalls passieren, dass Sie eine leicht reflektierende Oberfläche erhalten. Dann kann, obwohl die Kamera auf die rote Ampel gerichtet ist, die „gesehene" rosa oder sogar fast weiß sein. Das ist für das neuronale Netz natürlich nicht sehr nützlich. Es muss die *rote Ampel* unter optimalen Bedingungen kennen. Bewegen Sie also die Kamera oder ändern Sie die Beleuchtung, bis Sie das Gefühl haben, einen RGB-Wert erreicht zu haben, der die Farbe optimal darstellt.

Dies wirft auch ein weiteres Problem hinsichtlich der Genauigkeit auf. Autofahrer wissen, wie schwierig es ist, die aktive Ampel zu erkennen, wenn die Sonne in die Augen oder direkt in die Lampen scheint. Wenn wir Menschen nicht unterscheiden können, welche Lampe leuchtet, wie kann es dann ein neuronales Netz herausfinden? Die Antwort lautet: Es kann es nicht. Wenn wir einen robusteren Algorithmus brauchen, müssen wir ihn mit Daten über „schlechte Sicht" trainieren. Möglicherweise müssen wir auch einen anderen Input bereitstellen, zum Beispiel, woher die Sonne scheint, damit das neuronale Netz weiß, wann „schlechte Sicht" herrscht und dass es den Datensatz für „schlechte Sicht" verwenden sollte. Eine andere Idee wäre es, die von der Kamera eingehenden Daten zu verbessern, indem wir vielleicht ein Infrarotbild der Ampel (welche Lampe ist heiß?) oder eine andere clevere Filterung hinzufügen. Aber nicht verzweifeln! Wir werden unseren Trainingsdaten eine gewisse Robustheit verleihen, um mit einer gewissen Variation in der Farbe unserer Ampel fertig zu werden.

Die entscheidende Aufgabe aber ist die Erfassung der RGB-Daten für die rote, gelbe und grüne Lampe der Ampel, so wie Sie sie mit Ihrer Kamera und Ihren Lichtverhältnissen ermitteln. Die Daten, die mit dem Aufbau des Autors gesammelt wurden, sind in **Tabelle 1** dargestellt.

Erkennung von Verkehrsampeln

Mit diesen Daten können wir nun das neuronale Netz auf die Farben unserer Ampel trainieren. Für diesen letzten Schritt benötigen wir das Projekt *trafficlight/tlight_detect/tlight_detect.pde*. Stellen Sie zunächst sicher, dass der korrekte Initialisierungscode für die Kamera in Zeile 38 (aus *find_camera.pde*) eingefügt wird.

Dieses Projekt verwendet einen MLP-Knoten mit drei Eingängen, sechs verborgenen Knoten und vier Ausgängen (3/6/4). Die drei Eingänge stehen für die drei Farben R, G und B. Drei der vier Ausgänge dienen der Klassifizierung von „Rot", „Gelb" und „Grün" der Ampel. Der vierte wird später verwendet. Die willkürlich gewählten sechs versteckten Knoten sollten ausreichen, um die Klassifizierungsaufgabe zu bewältigen. Wir werden feststellen, dass diese Konfiguration funktioniert, und wie zuvor werden Sie ermutigt, mit weniger oder mehr versteckten Knoten zu experimentieren.

Wenn Sie den Code so ausführen, wie er ist, arbeitet das neuronale Netz ohne jeglichen Lernprozess. Die Ergebnisse (**Bild 7**) zeigen, dass jede Farbe als alle gültigen Farbe klassifiziert wird, die das Netzwerk erkennen soll.

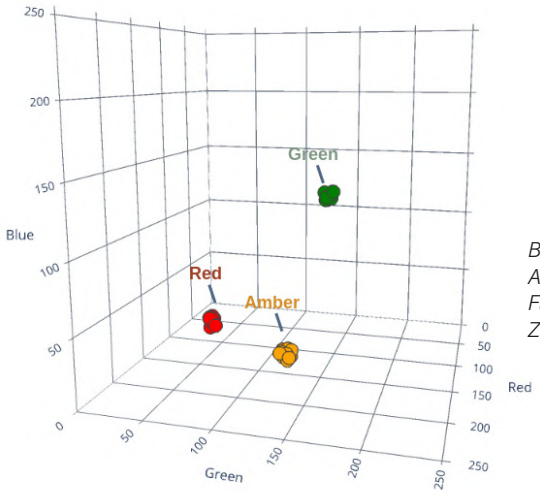


Bild 8. Die drei Ampelfarben im RGB-Farbraum nach der Zufallssteuerung.

Bild 9. Auch tlight_detect.pde hat Rot, Gelb und Grün gelernt.

Das Anlernen des neuronalen Netzes erfolgt in und um Zeile 51. Entfernen Sie die Kommentierung der ersten drei Methoden und fügen Sie die RGB-Werte hinzu, die Sie zuvor ermittelt haben. Die drei Methoden, die zur Definition von Rot, Gelb und Grün verwendet werden, sind folgende:

```
teachRed(159, 65, 37);
teachAmber(213, 141, 40);
teachGreen(128, 152, 130);
```

Jedes Mal, wenn diese Funktionen aufgerufen werden, wird das Netz darauf trainiert, diese RGB-Kombination als die zugehörige Farbe zu klassifizieren (**Bild 8**). Um Beleuchtungsschwankungen und automatische Änderungen der Belichtungseinstellungen durch die Kamera auszugleichen, werden die RGB-Werte mit Hilfe der Funktion `randomise()` (Zeile 336) ein wenig variiert (±4). Auch hier können Sie mit der Wirksamkeit dieses Ansatzes und dem Grad der Zufallssteuerung experimentieren.
Das Training des Netzes geht im Vergleich zu früheren Projekten sehr schnell, da wir die Fehlerwerte beim Lernen nicht mehr in eine Datei schreiben. Führen Sie das Projekt einfach aus, und nach ein paar Sekunden beginnt das neuronale Netz, die Farbe im roten Begrenzungsquadrat des Kamerafensters zu bewerten **(Bild 9**).
Die im Blickfeld der Kamera befindliche Ampel wird ab Zeile 156 ermittelt. Die erfassten RGB-Farben werden auf die Eingänge des MLPs angewendet, und die Ausgabe des Netzes wird berechnet:

```
network.setInputNode(0, (float) r
   / 255.0);
network.setInputNode(1, (float) g
   / 255.0);
network.setInputNode(2, (float) b
   / 255.0);
network.calculateOutput();
```

Die Entscheidung über die gesehene Farbe wird dann ab Zeile 171 getroffen. Die Ausgabe der einzelnen Ausgangsknoten wird ausgewertet. Liegt die Klassifizierungssicherheit über 90 % (0,90), wird die gesehene Farbe in einem Kreis zusammen mit dem Klassifikator „Rot“, „Gelb“ oder „Grün“ angezeigt.

```
// If likelihood of 'Red' > 90%...
if (network.getOutputNode(0) >
   0.90) {
  fill(200, 200, 200);
     // …write "Red"…
  text("Red", 640+(100), 320);
     // … and set color to the
       color seen.
  fill(r, g, b);
} else {
     // Otherwise, set color to
        black
  fill(0, 0, 0);
}
// Now draw the color seen in a
   circle
ellipse(640+(50), 300, 40, 40);
```

Sie können mit der Präzision des MLPs experimentieren, indem Sie die Kamera in verschiedenen Winkeln und unter verschiedenen Lichtverhältnissen auf die verschiedenen Ampeln richten. Versuchen Sie auch, die Kamera auf Objekte in Ihrer Umgebung zu richten, die Ihrer Meinung nach den Farben, die das MLP gelernt hat, nahe kommen.
Sie werden feststellen, dass eine Vielzahl von Farben als eine bekannte Farbe falsch klassifiziert wird. Im Beispiel des Autors wird die Klassifizierung „grün“ auch für die Ampelumrandung, den Rahmen und den gesamten Bildhintergrund angegeben (**Bild 10**).

Schärfung der Klassifizierung

Aus den angezeigten RGB-Werten wird ersichtlich, dass die an das MLP übergebenen

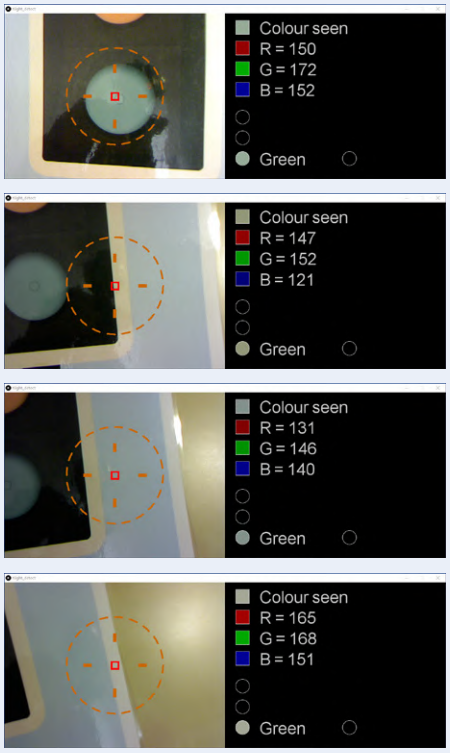


Bild 10. Fehlklassifizierung anderer Farben als „grün“.

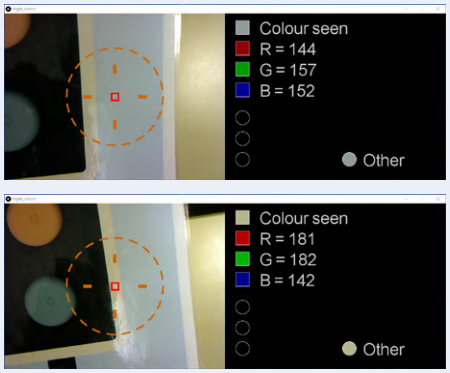


Bild 11. Fehleinstufung anderer Farben als „grün“ mit Hilfe des Ausgangsknotens „andere“ behoben.

Tabelle 2. Erfasste RGB-Werte für die „unerwünschten“ Farben.

Objekt	R	G	B
Dunkler Ampel-hintergrund	76	72	35
Weißer Rahmen	175	167	138
Blauer Hintergrund	152	167	161

Farben der gewünschten Klassifizierung für „Grün“ recht nahe kommen. Es gibt mehrere Möglichkeiten, die Klassifizierung zu verbessern. Die erste wäre, die 90%-Marke für eine genaue Klassifizierung auf einen höheren Wert anzuheben. Eine andere Möglichkeit wäre, die Anzahl der Lernphasen (Epochen) zu erhöhen. Eine weitere besteht darin, die Implementierung der Klassifizierung zu überdenken.

Bislang haben wir uns darauf konzentriert, was wir positiv klassifizieren möchten. Manchmal kann es jedoch hilfreich sein, dem neuronalen Netz beizubringen, was nicht zu den Mustern gehört, nach denen wir suchen. Im Grunde können wir sagen: „Das sind drei Dinge, nach denen wir suchen, aber hier sind einige ähnliche Dinge, nach denen wir definitiv nicht suchen.“ An dieser Stelle kommt unser vierter Ausgabeknoten ins Spiel. Um noch einmal zum Projekt *computer_vision.pde* zurückzukehren, können wir die RGB-Werte für Farben erfassen, die wir als „andere“ klassifizieren möchten. Als Beispiel wurden die RGB-Werte für die dunkle Ampelumrandung, den weißen Rand und den blauen Hintergrund erfasst, wobei die Ergebnisse in **Tabelle 2** dargestellt sind.

Zurück in *tlight_detect.pde*, können diese Werte mit der Methode `teachOther()` als „Andere“ unerwünschte Farben eingelernt werden. Entfernen Sie einfach das Kommentarzeichen in Zeile 60 und fügen Sie Ihre RGB-Werte ein:

```
teachOther(76, 72, 35);
teachOther(175, 167, 138);
teachOther(152, 167, 161);
```

Eine erneute Ausführung des Projekts führt zu einer deutlichen Verbesserung. Der Bereich um die Ampel (Umrandung, Rahmen, Hintergrund) wird als „Other“ anstatt als „Grün“ klassifiziert (**Bild 11**).

Experimente bis zum nächsten Mal

In diesem Artikel haben wir nun gesehen, wie ein neuronales Netz ein reales Klassifizierungsproblem löst. Wir haben auch gelernt, dass es helfen kann, sowohl die gewünschten als auch die unerwünschten Klassifizierungsdaten zu lernen.

Warum versuchen Sie nicht, auf diesem Beispiel aufzubauen und Folgendes zu untersuchen?

- Wie „robust“ kann man das MLP gegen Kamerawinkel und Belichtungsänderungen machen? Ist es besser, die Lerndaten stärker zu randomisieren oder die Messlatte für die Klassifizierung höher zu legen (>90%)?
- Verbessert sich die Genauigkeit, wenn man die Anzahl der Other-Ausgabeknoten erhöht und jedem eine unerwünschte Farbe beibringt?
- Welche Auswirkungen hat die Verringerung oder Erhöhung der Anzahl der versteckten Knoten auf das System?
- Würde ein vierter Eingang für „allgemeine Helligkeit“ helfen, die Erkennungsgenauigkeit bei unterschiedlichen Lichtverhältnissen zu erhöhen?

Während es großartig ist, neuronale Netze auf leistungsstarken Laptops und PCs auszuführen, wünschen sich viele von uns eine solche Fähigkeit auf den kleineren Controllern, die auf Boards wie dem Arduino verwendet werden. In unserem letzten Artikel dieser Reihe werden wir einen RGB-Sensor und einen Arduino verwenden, um ein neuronales Netzwerk zur Farberkennung zu implementieren. Vielleicht bildet es die Grundlage für Ihr nächstes smartes Arduino-Projekt!

210160-C-01

Haben Sie Fragen oder Kommentare?

Haben Sie technische Fragen oder Kommentare zu diesem Artikel? Dann schreiben Sie bitte dem Autor eine E-Mail an stuart.cording@elektor.com.

Ein Beitrag von

Idee, Text and Illustrationen: **Stuart Cording**
Redaktion: **Jens Nickel, C. J. Abate**
Illustrations: **Patrick Wielders**
Übersetzung: **Textmaster**
Layout: **Harmen Heida**

PASSENDE PRODUKTE

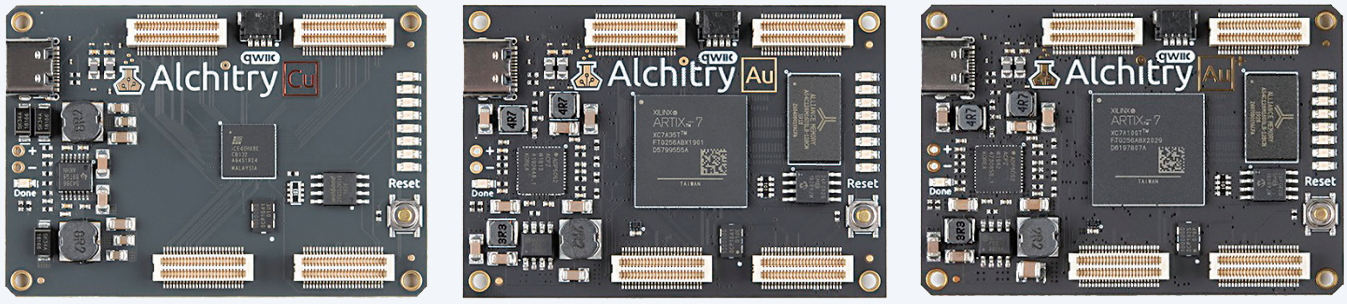
- **B. van Dam, *Artificial Intelligence* (E-book) (SKU 18090)**
  www.elektor.de/18090
- **Google AIY Vision Kit for Raspberry Pi (SKU 19365)**
  www.elektor.de/19365
- **HuskyLens AI Camera with Silicone Case (SKU 19248)**
  www.elektor.de/19248

WEBLINKS

[1] M. Anderson, „Surprise! 2020 Is Not the Year for Self-Driving Cars“, IEEE Spectrum, April 2020: http://bit.ly/2ZSwm5f
[2] GitHub-Repository - simple-neural-network: https://bit.ly/2ZHLv9p
[3] H. Zhang et al., „StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks“, CVF, Dezember 2017: https://bit.ly/3qZccCs

Im Inneren eines **Open-Source-Processors**

Ein Beispiel-Kapitel: Lattice- und Xilinx-FPGAs im Vergleich



Die in diesem Artikel erwähnten Alchitry-FPGA-Entwicklungsboards. Von links nach rechts: Cu - Au - Au+.

Von **Monte Dalrymple** (USA)

In dieser Rubrik stellen wir Ihnen Auszüge aus Monte Dalrymples Buch *Inside an Open-Source Processor* vor, das bei Elektor erschienen ist. FPGA-Technologie in Kombination mit (V)HDL ist unter Elektronikern sehr beliebt, und RISC-V bietet den Open-Source-Weg zu professionellen Anwendungen. In diesem Artikel vergleichen wir einige Lattice- und Xilinx-FPGA-Boards der Alchitry-Reihe (jetzt bei Elektor erhältlich) und sehen uns die Ergebnisse einer Beispielanwendung an. Aber zuerst ein paar Tipps!

Anmerkung der Redaktion: *Dieser Artikel ist ein Auszug aus dem Buch* Inside an Open-Source Processor - an Introduction to RISC-V, *formatiert und leicht bearbeitet, um den redaktionellen Standards und dem Seitenlayout von Elektor zu entsprechen. Da es sich um einen Auszug aus einer größeren Publikation handelt, können sich einige Begriffe in diesem Artikel auf Diskussionen an anderer Stelle im Original beziehen. Der Autor und die Redaktion haben ihr Bestes getan, um solche Fälle auszuschließen, und sind gerne bereit, bei Fragen zu helfen. Kontaktinformationen finden Sie im Kasten* ***Haben Sie Fragen oder Kommentare?***

In der Vergangenheit war das Einrichten eines Projekts in einem FPGA-Toolflow eine beängstigende Aufgabe, aber mit der Weiterentwicklung der Technologie ist diese Aufgabe viel einfacher geworden. Bei unserem Beispiel einer 24-Stunden-Uhr (die an anderer Stelle im Buch beschrieben wird) ist der größte Teil der Arbeit bereits erledigt.

Man sollte sich aber darüber im Klaren sein, dass große Projekte, die die Möglichkeiten des FPGAs maximal ausnutzen, doch unweigerlich viel komplexer sind.

Sowohl bei Lattice als auch bei Xilinx müssen neben der Auswahl des geeigneten FPGAs für das Projekt nur noch zwei oder drei Dateien in den Tools verknüpft werden. Bei diesem vereinfachten Ansatz müssen sich alle Projektdateien im selben Verzeichnis befinden. Es ist dringend zu empfehlen, die Alchrity-Tutorials für die

Installation der FPGA-Software und die Einrichtung eines Projekts zu lesen, da hier nur die wesentlichen Details behandelt werden.

Tipps: Lattice (Alchrity Cu)

Das auf diesem Board verwendete FPGA erfordert die Lattice-Software iCEcube2, ein ausgereiftes Produkt ohne viel Schnickschnack. Es gilt aber ein paar Dinge zu beachten:

- Stellen Sie sicher, dass Sie das Projekt mit den richtigen Device-Optionen einrichten. Der Alchrity Cu verwendet den HX8K-FPGA aus der iCE40-Familie im CB132-Gehäuse. Alle I/O-Bänke verwenden eine 3,3-V-Versorgung.
- Nur die Datei *yrv_alchrity.v* muss als Designdatei eingegeben werden, da alle anderen Designdateien automatisch in der richtigen Reihenfolge eingefügt werden.
- Wählen Sie für das Synthesewerkzeug die *Lattice LSE Synthesis* aus. Während der Logiksynthese sind keine Constraint-Dateien erforderlich.
- Nach Abschluss der Synthese müssen die Pin-Constraint-Datei *yrv_alchrity.pcf* und die Timing-Constraint-Datei *timing.sdc* verknüpft werden, bevor die restlichen Tools ausgeführt werden.

Es dürfte eine Reihe von Warnungen in den verschiedenen Protokolldateien geben, aber keine von ihnen ist wichtig genug, um irgendwelche Maßnahmen einzuleiten.

Tipps: Xilinx (Alchrity Au und Au+)

Die FPGAs auf diesen Boards erfordern die Vivado-Software von Xilinx. Dies ist ein sehr komplexes Produkt, und wir werden nur einen kleinen Teil der Möglichkeiten dieser Software nutzen. Auch hier sind ein paar Dinge zu beachten:

- Nur die Datei *yrv_alchrity.v* muss als Design-Datei angegeben werden.
- Es ist nur die Constraint-Datei *yrv_alchrity.xdc* erforderlich, die die Pin-Constraints, die Timing-Constraints sowie die Spezifikationen für die Programmierspannung enthält.
- Die korrekten Device-Optionen sind *xc7a35tftg256-1* für das Au-Board und *xc7a100tftg256-1* für das Au+-Board.

Es gibt für diese FPGA noch mehr Warnungen in den Logdateien, aber auch hier sollte keine von ihnen wichtig genug sein, um Maßnahmen zu ergreifen.

FPGA-Ergebnisse

Dieses Beispielprojekt wurde auf allen drei Ziel-FPGA-Entwicklungsboards implementiert und die Ergebnisse in den folgenden Abschnitten behandelt. An dieser Stelle ist es jedoch interessant, einige der Ergebnisse „side-by-side“ miteinander zu vergleichen. **Tabelle 1** zeigt die gemeldete Taktgeschwindigkeit für den 100-MHz-Teiler.

Der 100-MHz-Teiler wurde, um eine maximale Geschwindigkeit zu erreichen, mit nur einer Logikebene zwischen den Flipflops entworfen. Dieses Ergebnis spiegelt also die Möglichkeiten der Technologie wider.

Tabelle 1: FPGA-Ergebnisse.

100-MHz-Teiler	Alchrity Cu	Alchrity Au	Alchrity Au+
Geschwindigkeit (bei Synthese)	313,9 MHz	114,1 MHz	114,2 MHz
Geschwindigkeit (bei der Platzierung)	542,3 MHz	-	-
Geschwindigkeit (endgültig)	625,4 MHz	111,9 MHz	111,9 MHz

Tabelle 2: FPGA-Ergebnisse.

FPGA Attribut	Alchrity Cu	Alchrity Au	Alchrity Au+
Logische Elemente	7.680	20.800	63.400
Verwendete LEs	4.646	2.189	2.185
Verwendete LEs (%)	60,6%	10,5%	3,5 %
Flip-Flops	1331	1377	1377
Geschwindigkeit (bei Synthese)	11,8 MHz	28,1 MHz	27,7 MHz
Geschwindigkeit (bei der Platzierung)	25,8 MHz	-	-
Geschwindigkeit (endgültig)	30,8 MHz	34,4 MHz	34,2 MHz

Die für das Lattice-FPGA angegebene Geschwindigkeit ist, wenn sie stimmt, beeindruckend. Besonders interessant ist, dass von der Implementierung der Logiksynthese über die Logikplatzierung bis hin zum endgültigen gerouteten Ergebnis die gemeldete Geschwindigkeit besser und die Angabe vermutlich auch genauer wird. Obwohl es auf den ersten Blick günstig erscheint, wenn die endgültige Leistung besser ist als die anfängliche Schätzung, ist eine Abweichung um den Faktor zwei nicht ideal, da dies dazu führen könnte, dass ein Projekt anfangs fälschlicherweise als nicht durchführbar eingestuft wird.

Die Xilinx-Tools geben bei der Logikplatzierung keine separate Geschwindigkeitsschätzung ab, aber die gemeldeten Geschwindigkeiten bei der Logiksynthese und beim Endergebnis sind nahezu identisch. Aus der Sicht eines Entwicklers ist dies vorzuziehen.

Tabelle 2 zeigt die für das Projekt erforderlichen Ressourcen sowie die maximale Taktfrequenz für die CPU selbst. Diese Ergebnisse enthalten keinen der 64-Bit-Zähler, die Teil der *RISC-V Privileged Architecture* sind.

Die zweite Zeile dieser Tabelle zeigt, warum der Vergleich von FPGA-Ergebnissen so kompliziert sein kann. Aus der Tabelle geht

Listing 1: Auszug aus dem Lattice-Report.

```
Total Logic Cells: 4646/7680
       Combinational Logic Cells: 3315    out of   7680    43.1641%
       Sequential Logic Cells:    1331    out of   7680    17.3307%
       Logic Tiles:               847     out of   960     88.2292%
   Registers:
       Logic Registers:           1331    out of   7680    17.3307%
       IO Registers:              0       out of   1280    0
   Block RAMs:                    12      out of   32      37.5%
   Warm Boots:                    0       out of   1       0%
   Pins:
       Input Pins:                36      out of   95      37.8947%
       Output Pins:               50      out of   95      52.6316%
       InOut Pins:                0       out of   95      0%
   Global Buffers:                2       out of   8       25%
   PLLs:                          0       out of   2       0%
```

hervor, dass die Xilinx-Implementierung weniger als die Hälfte der von der Lattice-Implementierung benötigten Ressourcen benötigt. Allerdings enthält ein Xilinx-Logikelement wesentlich mehr Logik als eines von Lattice. Darüber hinaus verwendet das Xilinx-Design dedizierte DSP-Blöcke, über die der Lattice-FPGA nicht verfügt.
Ein Blick auf den prozentualen Anteil der verwendeten Logikelemente zeigt die Größenunterschiede zwischen den drei FPGAs auf. Etwas mehr als die Hälfte des Lattice-FPGAs wird für das *yrv_mcu*-Design benötigt, aber nur etwa ein Zehntel beim mittleren Xilinx- und sogar weniger als ein Zwanzigstel beim großen Xilinx-Baustein. Dies steht im Einklang mit den Kosten der verschiedenen Entwicklungsboards.
Wie zu erwarten, ist die Anzahl der für das Design erforderlichen Flip-Flops bei allen drei FPGAs ähnlich. Die genaue Anzahl liegt eher in der Arbeitsweise des Logiksynthesetools begründet, die Flipflops replizieren, um die Leistung zu erhöhen oder das Routing zu vereinfachen.
Die Geschwindigkeitsangaben zeigen, dass alle drei FPGAs ungefähr die gleiche Leistung bieten, was etwas überraschend ist, wenn man bedenkt, dass das Lattice-FPGA nicht die dedizierten DSP-Blöcke enthält, die in der Xilinx-Implementierung verwendet werden.

Details: Alchrity Cu

Listing 1 ist direkt dem Lattice-Implementierungsbericht entnommen und zeigt die Details der Ressourcennutzung. **Bild 1** ist ein Screenshot des Floorplanner-Tools von Lattice und zeigt die Auslastung der Logikelemente verteilt über das FPGA. Diese Verteilung ist ziemlich gleichmäßig über das gesamte Device, obwohl insgesamt nur etwa 60 % genutzt werden.

Details: Alchrity Au

Listing 2 wurde direkt aus dem Xilinx-Implementierungsbericht extrahiert und zeigt die Details der Ressourcennutzung. Der

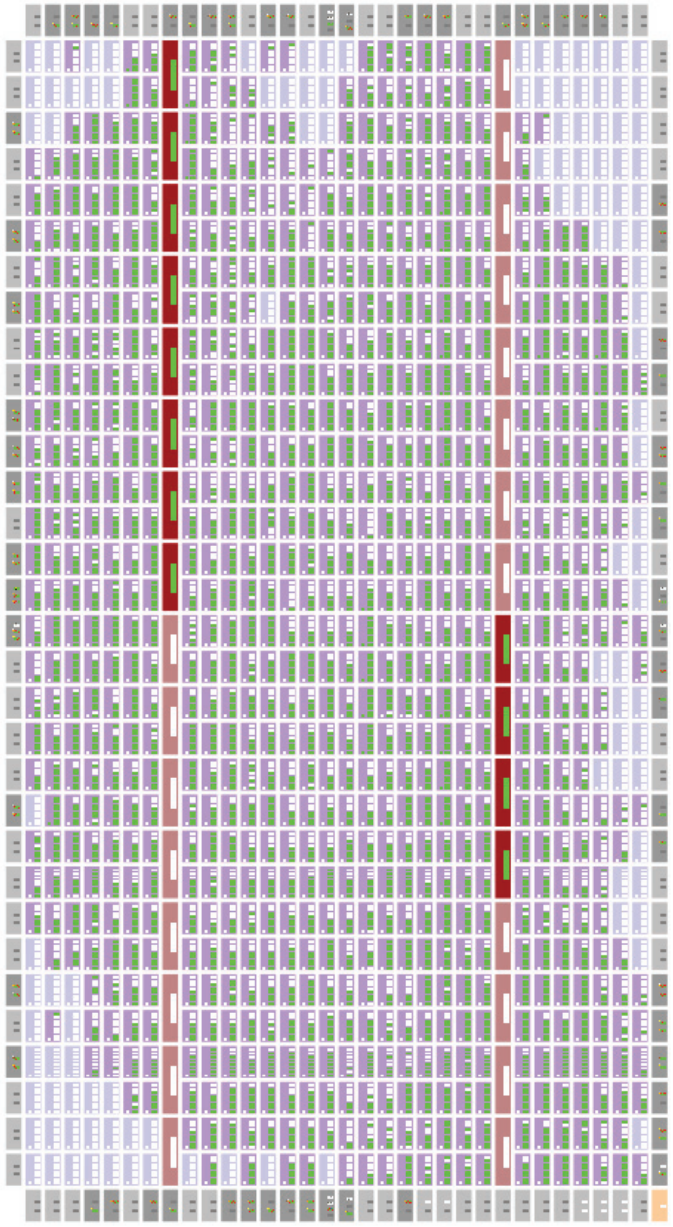


Bild 1. Floorplanner-Ansicht bei Lattice (Alchitry Cu).

vollständige Xilinx-Bericht ist wesentlich umfangreicher und deckt die gesamte im FPGA verfügbare dedizierte Hardware ab.
Bild 2 stammt aus einem Screenshot des Floorplanner-Tools von Xilinx und zeigt die Auslastung der Logikelemente, wie sie im FPGA verteilt sind, sowie die Verteilung der dedizierten Logikblöcke. Es scheint, dass die Platzierungswerkzeuge immer in der linken unteren Ecke eines Bausteins beginnen. Warum die Logik in zwei Teile aufgeteilt zu sein scheint, ist mir nicht ersichtlich.

Listing 2: Auszug aus dem Xilinx-XCA35T-Report.

```
1. Slice Logic
--------------

+-------------------------+------+-------+-----------+-------+
|        Site Type        | Used | Fixed | Available | Util% |
+-------------------------+------+-------+-----------+-------+
| Slice LUTs              | 2189 |     0 |     20800 | 10.52 |
|   LUT as Logic          | 2189 |     0 |     20800 | 10.52 |
|   LUT as Memory         |    0 |     0 |      9600 |  0.00 |
| Slice Registers         | 1377 |     0 |     41600 |  3.31 |
|   Register as Flip Flop | 1377 |     0 |     41600 |  3.31 |
|   Register as Latch     |    0 |     0 |     41600 |  0.00 |
| F7 Muxes                |    3 |     0 |     16300 |  0.02 |
| F8 Muxes                |    0 |     0 |      8150 |  0.00 |
+-------------------------+------+-------+-----------+-------+
```

Details: Alchrity Au+

Listing 3 ist ebenfalls direkt aus dem Xilinx-Implementierungsbericht entnommen und zeigt Details zur Ressourcenauslastung. Beachten Sie, dass die Zahlen für die genutzten Ressourcen bei den beiden Xilinx-FPGAs nahezu identisch sind.

Bild 3 ist ebenfalls einem Screenshot des Xilinx-Floorplanner-Tools entnommen. Interessant ist hier, dass, obwohl dieses FPGA die dreifache Anzahl von Logikelementen enthält, der Gesamtflächenbedarf des Beispiels hier fast auf die gleiche Fläche wie bei der ersten Xilinx-Implementierung verteilt zu sein scheint.

Hardware-Programmierung

Das Herunterladen des FPGA-Bitstreams auf eine Alchrity-Entwicklungsplatine ist mit dem Alchrity-Loader-Programm einfach. Dieses eigenständige Programm wird automatisch als Teil der bei Alchrity erhältlichen Labs-Software installiert. **Bild 4** zeigt die Benutzeroberfläche dieses Programms.

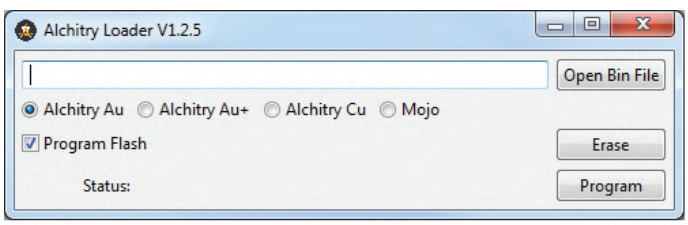


Bild 4. GUI des Alchitry-Loaders.

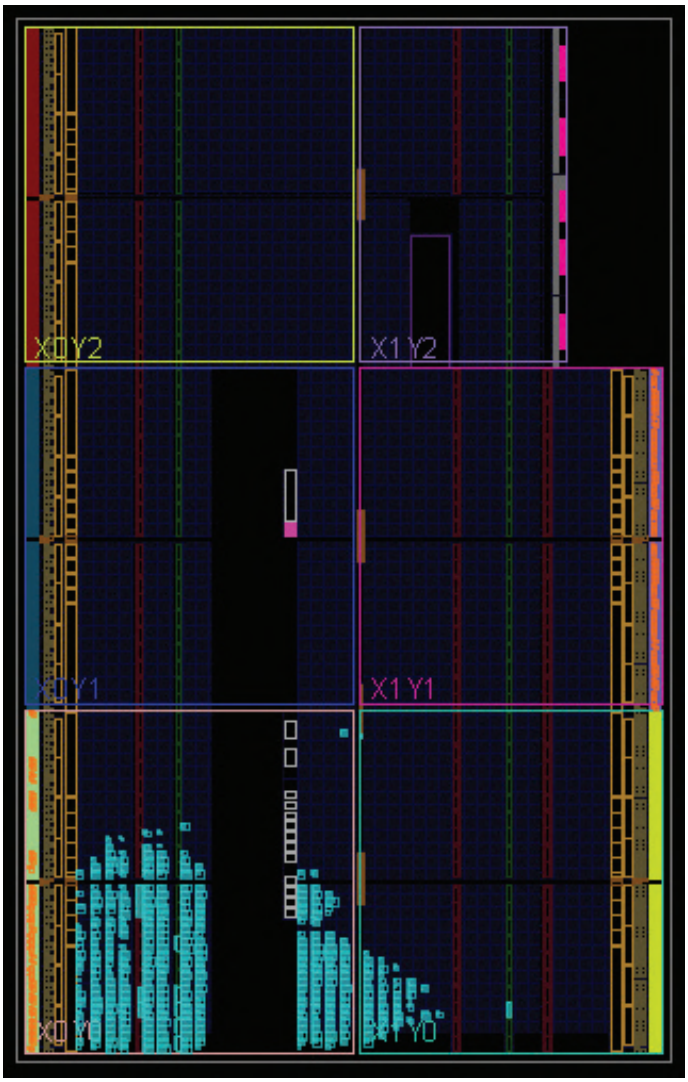


Bild 2. Floorplanner-Ansicht bei Xilinx XCA35T (Alchitry Au).

Zur Programmierung des Flash-Speichers auf der Entwicklungsplatine, die zum Laden des FPGAs verwendet wird, ist lediglich die Angabe der Bitstream-Datei, die Auswahl der Zielplatine und ein Klick auf den *Program*-Knopf erforderlich.

Standardmäßig sucht das Programm nach einer Bitstream-Datei vom Typ *.bin*, die von der iCEcube2-Software standardmäßig verwendet wird. Die Vivado-Software generiert eine Bitstream-Datei des Typs *.bit*, was Sie beim Laden der Xilinx-Bitstream-Datei berücksichtigen sollten.

210347-02

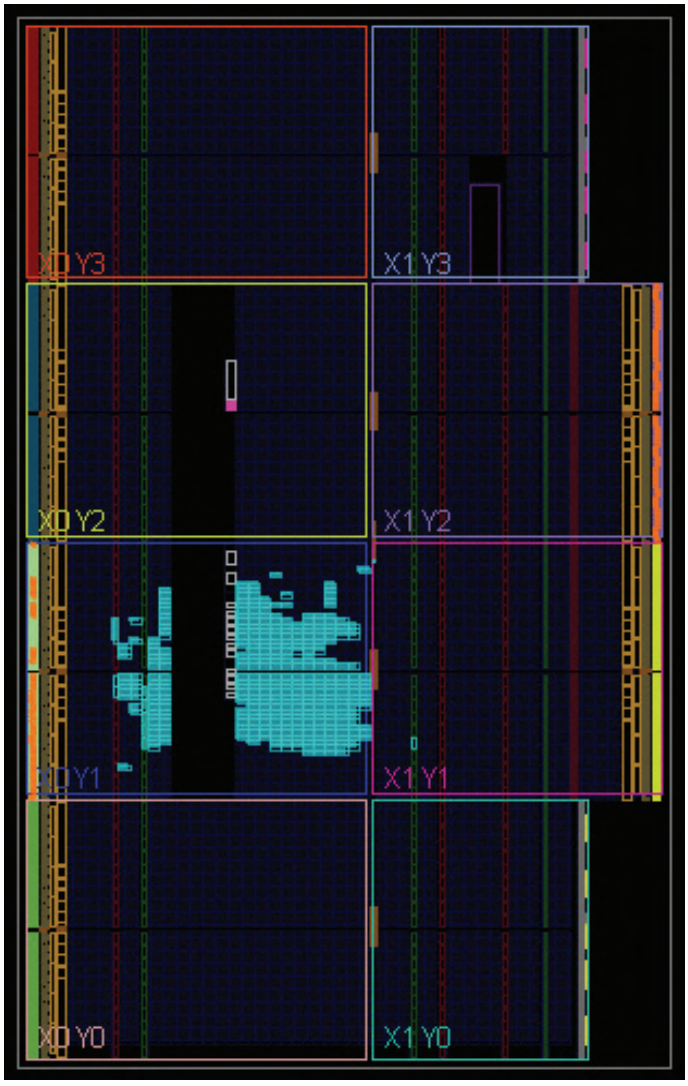


Bild 3. Floorplanner-Ansicht bei Xilinx XCA100T (Alchitry Au+).

Listing 3: Auszug aus dem Xilinx-XCA100T-Report.

```
1. Slice Logic
--------------

+-------------------------+------+-------+-----------+-------+
|        Site Type        | Used | Fixed | Available | Util% |
+-------------------------+------+-------+-----------+-------+
| Slice LUTs              | 2185 |     0 |     63400 |  3.45 |
|   LUT as Logic          | 2185 |     0 |     63400 |  3.45 |
|   LUT as Memory         |    0 |     0 |     19000 |  0.00 |
| Slice Registers         | 1377 |     0 |    126800 |  1.09 |
|   Register as Flip Flop | 1377 |     0 |    126800 |  1.09 |
|   Register as Latch     |    0 |     0 |    126800 |  0.00 |
| F7 Muxes                |    3 |     0 |     31700 | <0.01 |
| F8 Muxes                |    0 |     0 |     15850 |  0.00 |
+-------------------------+------+-------+-----------+-------+
```

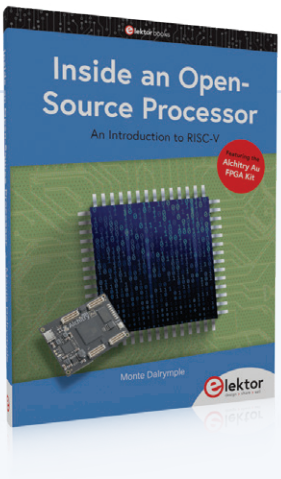
Haben Sie Fragen oder Kommentare?

Haben Sie technische Fragen oder Kommentare zu diesem Artikel? Dann schicken Sie eine E-Mail an den Autor unter monted@systemyde.com oder an Elektor unter redaktion@elektor.de.

Ein Beitrag von

Text: **Monte Dalrymple**
Redaktion: **Jan Buiting**
Übersetzung: **Rolf Gerstendorf**
Layout: **Giel Dols**

PASSENDE PRODUKTE



- **Book: M. Dalrymple, *Inside an Open-Source Processor* (Elektor 2021) (SKU 19826)**
  www.elektor.de/19826
- **E-Book: M. Dalrymple, *Inside an Open-Source Processor* (Elektor 2021) (SKU 19827)**
  www.elektor.de/19827
- **Alchitry Au FPGA Kit (SKU 19638)**
  www.elektor.de/19638

Aller Anfang …

muss nicht schwer sein: Die Spule lässt uns nicht los!

Von **Eric Bogers** (Elektor)

In dieser Folge fahren wir mit der Spule fort. Von Elektronikbastlern ungeliebt, aber so enorm wichtig.

Reihen- und Parallelschaltungen

Schwingkreise (**Bild 1**) sind die frequenzbestimmenden Kreise in Oszillatoren. Bei der sogenannten Resonanzfrequenz ist ihre Impedanz minimal oder maximal und die Impedanz des Kondensators gleich der Impedanz der Spule. Es gilt:

$$X_C = \frac{1}{2 \cdot \pi \cdot f_R \cdot C} = X_L = 2 \cdot \pi \cdot f_R \cdot L$$

$$\Rightarrow f_R = \frac{1}{2 \cdot \pi \cdot \sqrt{L} \cdot C}$$

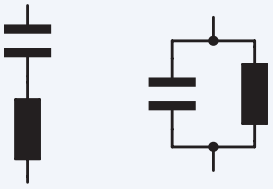


Bild 1. Serien- und Parallelschwingkreise.

Diese Formel gilt sowohl für Serien- als auch für Parallelschwingkreise. In **Bild 2** sehen Sie die Impedanzcharakteristik in Abhängigkeit von der Frequenz. Ein Serienschwingkreis kann (unter anderem!) in einer Lautsprecherbox verwendet werden, um einen kleinen Frequenzbereich abzuschwächen, um die Gesamtcharakteristik der Box zu linearisieren. Zu diesem Zweck muss in den meisten Fällen ein Widerstand in Reihe zu diesem Serienschwingkreis geschaltet werden.

Frequenzweichen

Bild 3 zeigt die Impedanz- und Phasenkennlinie eines Basslautsprechers (der übrigens in einem Frontloaded-Horngehäuse untergebracht ist, aber dieses Detail braucht man sich nicht zu merken). **Bild 4** zeigt die Frequenzcharakteristik dieses Systems

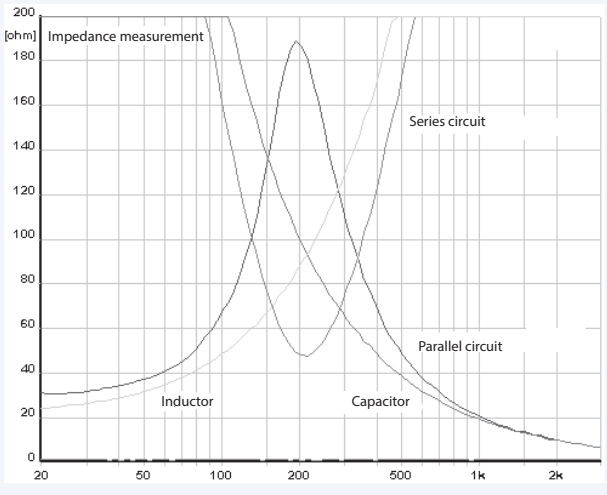


Bild 2. Impedanzkennlinien von Serien- und Parallelschwingkreisen.

und **Bild5** vermittelt einen Eindruck vom Ausmaß der vorhandenen Verzerrungen.

Wir sehen, dass wir den gesamten Frequenzbereich in vier Bereiche unterteilen können. Der *erste Bereich* liegt unterhalb der Resonanzfrequenz – bei diesem Lautsprechers ist dies der Bereich von 0 Hz bis etwa 30 Hz. Der Wirkungsgrad des Lautsprechers ist hier eher als mittelmäßig zu bezeichnen. Die Verzerrungen sind sehr stark, Spannung und Strom sind um etwa 60° bis 90° zueinander phasenverschoben. Das bedeutet auch, dass die Verlustleistung im Verstärker am höchsten ist. Aus diesen Gründen ist es sinnlos, den Lautsprecher in diesem Frequenzbereich zu verwenden: Erstens klingt er nicht gut, zweitens benötigt er wegen seines schlechten Wirkungsgrades viel zu viel elektrische Leistung, und drittens dürfte der Verstärker schnell überhitzen (mit allen damit verbundenen negativen Folgen).

Für diesen Lautsprecher sollte deshalb ein Hochpassfilter mit einer Eckfrequenz von mindestens 50 Hz verwendet werden. Für die Wiedergabe tieferer Frequenzen müssten wir einen anderen Lautsprechertyp und/oder ein anderes Gehäuse verwenden.
Der *zweite Bereich* liegt um die Resonanzfrequenz. Hier steigt die Impedanz (und damit auch der Wirkungsgrad) dramatisch an. In den meisten Fällen arbeiten Lautsprecherboxen in diesem Frequenzbereich recht gut (in diesem Beispiel allerdings nicht...). Im *dritten Bereich* ist die Impedanz eine einigermaßen horizontale (konstante) Linie und der Wirkungsgrad ist akzeptabel bei einem kleinen Klirrfaktor. Auch die Phasenkennlinie gibt keinen Anlass zur Sorge. Im Beispiel liegt dieser Bereich bei etwa 100...1000 Hz. Und das ist der Frequenzbereich, in dem dieser Lautsprecher eingesetzt werden sollte. Das Verzerrungsdiagramm zeigt die relativ großen Verzerrungen zweiter Ordnung (auch Oberwellen genannt), die für Hornsysteme charakteristisch sind.

Im *vierten Bereich* schließlich nimmt die Impedanz infolge der Selbstinduktion der Schwingspule allmählich zu, der Ausgangspegel sinkt deutlich und auch die Verzerrungen nehmen wieder zu. Dieser Punkt ist die obere Frequenzgrenze des Signals, das an diesen Lautsprecher angelegt werden sollte. Fällt bei passiven Frequenzweichen (siehe weiter unten) die Eckfrequenz in diesen vierten Bereich, so muss eine Impedanzkompensation eingesetzt werden (dies ist ein Thema der nächsten Folge).

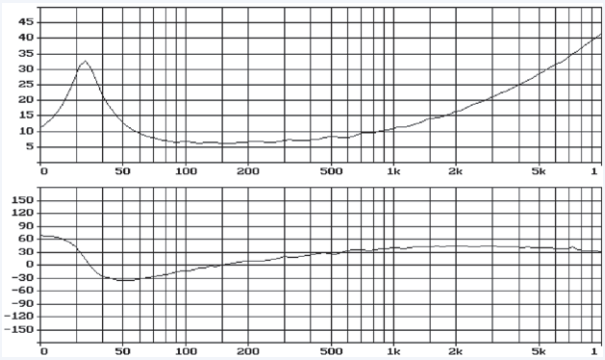


Bild 3. Impedanz- und Phasenkennlinien.

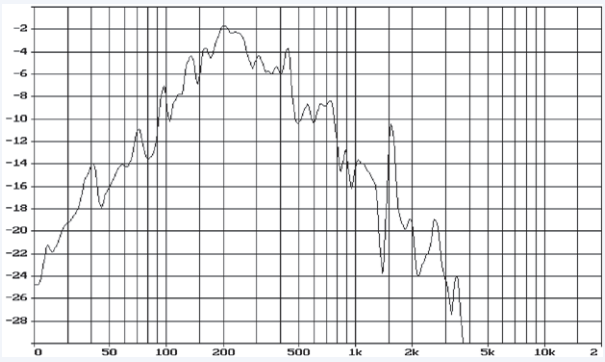


Bild 4. Frequenzkennlinie.

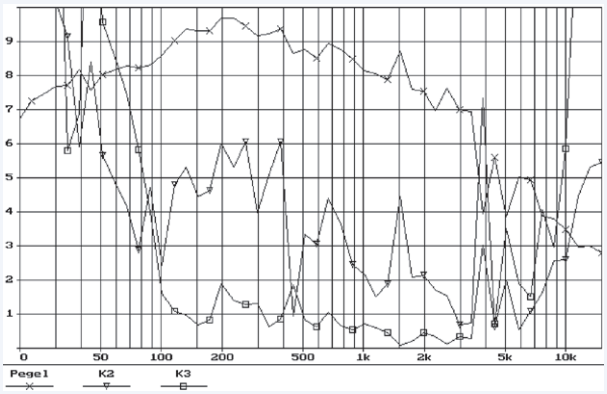


Bild 5. Verzerrung.

Warum Frequenzweichen?

Wie wir im obigen Beispiel gesehen haben, kann ein Lautsprecher (normalerweise) nur einen begrenzten Frequenzbereich gut wiedergeben. Deshalb muss das Audiospektrum von einer Frequenzweiche in eine Reihe von Bereichen aufgeteilt werden, die dann jeweils einem Lautsprecher zugeführt werden, der speziell für den entsprechenden Frequenzbereich ausgelegt ist.

Dies kann prinzipiell auf zwei verschiedene Arten geschehen: Das Signal kann *nach* dem Endverstärker mit Hilfe eines passiven Filternetzwerks (das aus Spulen und Kondensatoren besteht) aufgeteilt werden oder es wird *vor* dem Endverstärker in die verschiedenen Frequenzbereiche aufgeteilt, woraufhin jeder Signalbereich einen eigenen Endverstärker und Lautsprecher erhält. In diesem Fall spricht man von einer aktiven Frequenzweiche.

Nebenbei bemerkt, es ist auch eine Mischform möglich. Dabei werden die tiefen Frequenzen mit einem aktiven Filter vom Rest

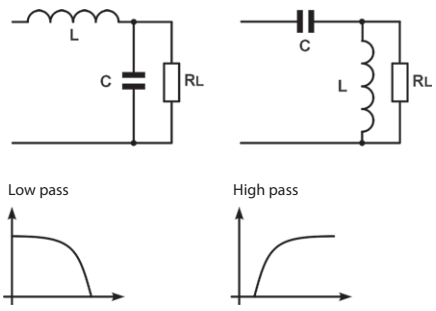


Bild 6. Passive Tiefpass- und Hochpassfilter mit einer Flankensteilheit von 12 dB/Oktave.

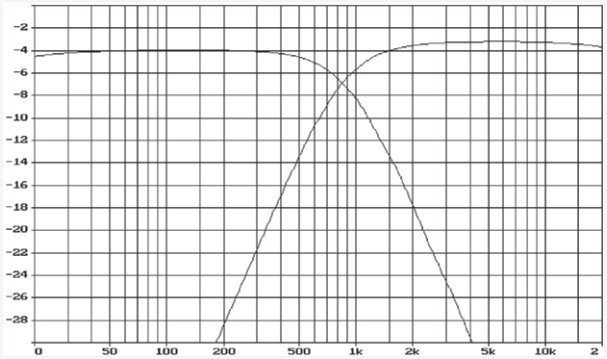


Bild 7. Frequenzgang eines Hoch- und eines Tiefpasses.

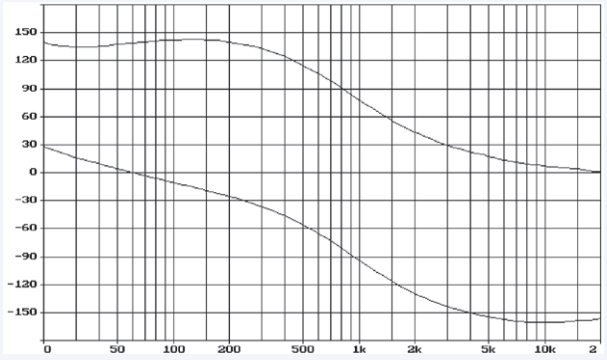


Bild 8. Phaseneigenschaften eines Hoch- und eines Tiefpassfilters.

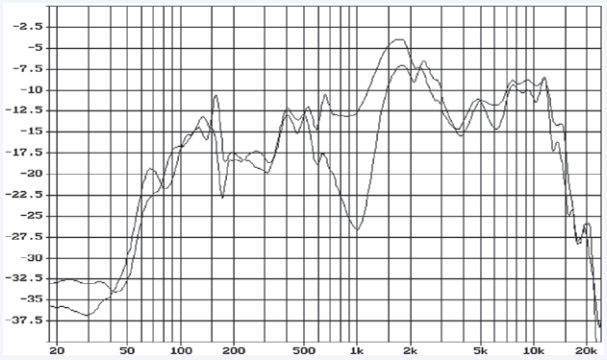


Bild 9. Richtig und falsch angeschlossene Frequenzweiche.

getrennt, und für diesen Rest wird ein passives Filter verwendet. Dies wird manchmal als Bi-Amping bezeichnet.
Nachfolgend werden wir mit passiven Frequenzweichen fortfahren; aktive Filter werden in einer späteren Folge behandelt.

Passive Frequenzweichen

Bild 6 zeigt die Schaltung eines Tiefpasses und eines Hochpasses mit einer Flankensteilheit von 12 dB/Oktave. Flankensteilheiten von 6 dB/Oktave, die mit einer in Reihe mit dem Signal geschalteten Spule oder einem Kondensator realisiert werden könnten, müssen bei Audioanwendungen vermieden werden. **Bild 7** veranschaulicht die Frequenzcharakteristik dieser Filter.
Die Eck- oder Übergangsfrequenz eines Filters ist die Frequenz, bei der der Pegel des Ausgangssignals um 3 dB gegenüber dem Eingangssignal verringert ist. **Bild 8** zeigt, dass bei der Eckfrequenz die Phase des Ausgangssignals im Vergleich zur Phase des Signals im Durchlassbereich um +90° (Hochpassfilter) beziehungsweise -90° (Tiefpassfilter) verschoben ist.
Wenn die Lautsprecher phasengleich angeschlossen sind, sind ihre Signale im Bereich der Trennfrequenz um 180° zueinander verschoben, so dass sich diese Signale gegenseitig aufheben. Das Ergebnis wäre eine „Senke“ in der Charakteristik des Systems, wie in der unteren Kurve von **Bild 9** zu sehen ist.

Im Prinzip spielt es keine Rolle, welcher der beiden Lautsprecher mit umgekehrten Anschlüssen versehen werden muss, aber um zu gewährleisten, dass zwei nahe beieinander stehende Lautsprecherboxen die gleiche Phase haben, wird empfohlen, sich an den internationalen „Standard“ zu halten, nämlich den Tieftonlautsprecher normal anzuschließen und den Hochtonlautsprecher mit umgekehrter Polarität.

In der Praxis kann man sich jedoch nicht blind auf die Beschriftung verlassen, die der Hersteller neben den Anschlüssen seiner Lautsprecher anbringt - erstens, weil jeder Hersteller machen kann, was er will, und zweitens, weil aufgrund unterschiedlicher Verzögerungen die Phase bei der Trennfrequenz tatsächlich umgekehrt sein kann. Sowohl bei aktiven als auch bei passiven Filtern muss die Polarität der Lautsprecher in der Praxis immer überprüft werden!
Für die Werte der Komponenten gilt (bei einer Butterworth-Charakteristik) folgendes:

$$C = \frac{1}{2 \cdot \sqrt{2} \cdot \pi \cdot f \cdot Z} = \frac{0{,}1125}{f \cdot Z}$$

$$L = \frac{Z}{\sqrt{2} \cdot \pi \cdot f} = \frac{0{,}2251 \cdot Z}{f}$$

Für den Entwurf von Frequenzweichenfiltern sind verschiedene Charakteristiken gebräuchlich, die jeweils Vor- und Nachteile in Bezug auf die Flankensteilheit bei der Trennfrequenz und das Impulsverhalten haben. Weitere Einzelheiten erläutert die einschlägige Fachliteratur.

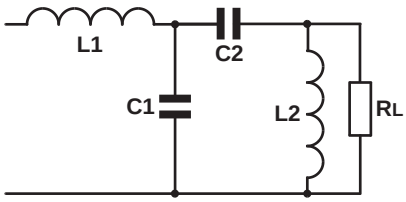


Bild 10. Bandpassfilter.

Bild 10 zeigt schließlich den prinzipiellen Aufbau eines Bandpassfilters, das eigentlich nur eine Reihenschaltung von Tiefpass- und Hochpassfilter ist. Die oberen und unteren Eckfrequenzen werden dann ebenfalls nach der bekannten Methode berechnet.

210564-03

Die Artikelreihe „Aller Anfang …" gründet auf dem Buch „Basiskurs Elektronik" von Michael Ebner, erschienen im Elektor Verlag.

Haben Sie Fragen oder Kommentare?

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

Ein Beitrag von

Idee und Illustrationen: **Michel Ebner**
Text und Redaktion: **Eric Bogers**
Übersetzung: **Rolf Gerstendorf**
Layout: **Giel Dols**

PASSENDE PRODUKTE

- **Ebner, Michael, „Basiskurs Elektronik"**
  www.elektor.de/basiskurs-elektronik-pdf
- **B. Kainka, Elektronik-Grundlagen und Einsteiger-Projekte (E-Book), Elektor 2020**
  www.elektor.de/elektronik-grundlagen-und-einsteiger-projekte-pdf

Anzeige

LPN liefert Leiterplatten aus Deutschland, vom Weltmarkt, aus NATO-Partnerländern oder mit anderen Restriktionen.

LPN ist nach ISO 9001:2015 Zertifiziert und das Personal beim FraunhoferInstitut geschult.

LPN liefert jedes Basismaterial und jede in Deutschland oder am Weltmarkt verfügbare Technik.
- Multilayer bis 56 Lagen.
- Starrflex, Flex, Semiflex.
- Aluminium, auch Bergquist, Kupferkern, Messingkern, Stahlkern.
- Teflon, auch Rogers.
- Montagehilfen Kaptonband, Abziehlack und Weiteres.

LPN Qualitätsprüfungen
- 100% Kontrolle
- Kupferstärkenmessung mit Magnetfeld Messgeräten.
- Nachmessen gedruckter Induktivitäten.
- Schliffbildauswertung.
- Lot-Benetzungs-Test.
- Delaminations-Test.
- alle Fertigungsstätten halten ISO 14001 ein.

LPN Dienstleistungen
- Datenaufbereitung incl. Nutzenaufbau,
- Machbarkeitsprüfung,
- EMPB.
- geklebte Vorlagen digitalisieren.
- Filme digitalisieren.
- Leiterplatten clonen.
- Leiterplatten nachlayouten.
- Terminaufträge.
- Abruflager für Jahreslose.

Profitieren Sie von den LPN Qualitätsstandards und den weltweiten Kontakten.

LPN Leiterplatten Nord GmbH
Hermann-Bössow-Straße 13-15
23843 Bad Oldesloe
leiterplatten-nord.de

Anfragen/Bestellungen:
lpn@lp-nord.de
Telefon 04531 1708 0

Projekt 2.0

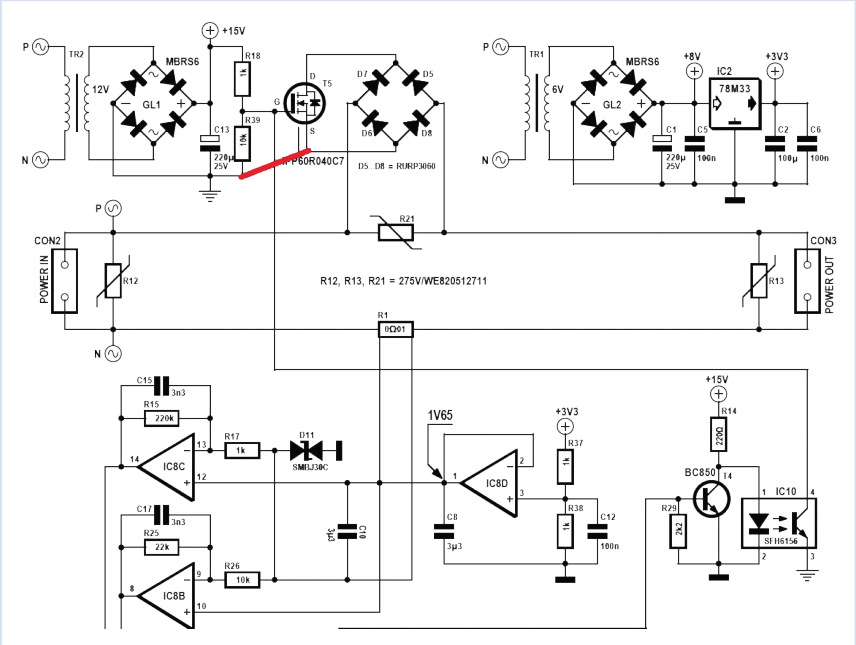
Korrekturen, Updates und Leserbriefe

Redaktion: **Ralf Schmiedel, Jens Nickel**

Netztransformatoren aus der Nähe betrachtet

Heft 09-10/2021, S. 34 (200587)

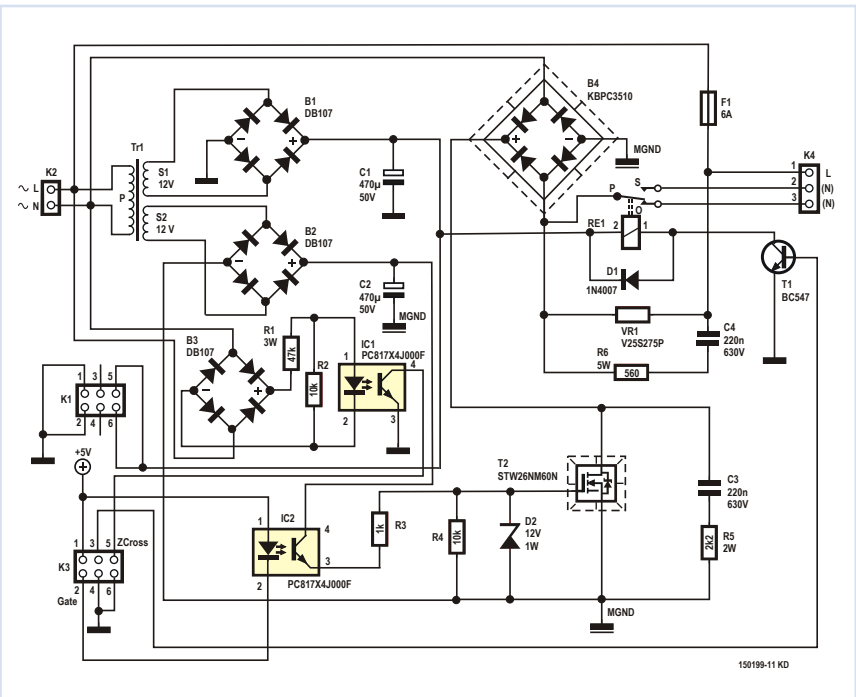
Im Schaltplan der Trafo-Testanordnung auf S. 35 fehlt eine Masseverbindung (siehe Abbildung).



Einstellbarer 1-kW-AC-Motortreiber

Heft 1-2/2020, S. 62 (191142)

Im Schaltplan der Netzteilplatine des 1-Kilowatt-AC-Motorantriebs wurde ein Fehler gefunden. Die Dioden von B4 sind falsch herum gezeichnet, sie sollten um 180° gedreht sein. Die „+"- und „-"-Symbole sind korrekt, die Leiterplatte ist ebenfalls korrekt. Der korrigierte Schaltplan ist abgebildet.

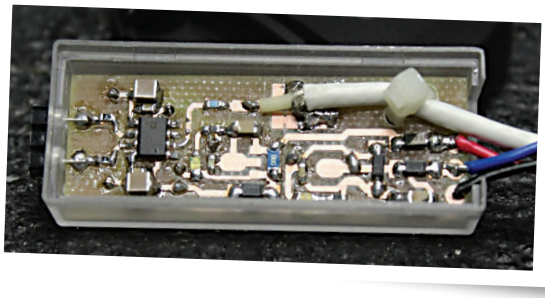


Differentieller Stromtastkopf für Oszilloskope 2.0

Heft 11-12/2020, S. 68 (180665)

Für den bei diesem Tastkopf verwendeten AD8421 gibt es mit dem INA849 von TI einen besseren pinkompatiblen Ersatz. Er benötigt keine Frequenzgangkompensation und hat eine deutlich höhere Bandbreite. Lediglich der Widerstand, der die Verstärkung festlegt, muss angepasst werden.

Alfred Rosenkränzer



Multifunktions-Komponententester LCR-T7 von Joy-IT

Heft 11-12/2021, S. 22 (210365)

Im Artikel wird der Eindruck erweckt, die Firma Joy-IT hätte das beschriebene Gerät selbst entwickelt. Dem ist aber definitiv nicht so. Das LCR-T7 ist eine Kopie eines Gemeinschaftsprojektes. Das LCR-T7 basiert auf einem Open Source Projekt (www.mikrocontroller.net/articles/AVR_Transistortester). Die ursprüngliche Idee eines automatischen Transistortesters stammt von von Markus Frejek, sie wurde von mehreren begeisterten Elektronikbastlern wie Markus Reschke, Karl-Heinz Kübbeler, Pieter-Tjerk de Boer und vielen anderen aufgegriffen und mit viel Mühe und Sachverstand immer weiter verbessert. Es gibt zu dem Tester auch ein spezielles Forum (www.mikrocontroller.net/topic/transistortester-avr), in dem sich die Entwicklung von 2012 bis heute nachvollziehen lässt.

Aber im Artikel von Elektor liest man dazu kein Wort. Ich gehe mal davon aus, dass das keine Absicht war, und nur der Unkenntnis der Redaktion zuzuschreiben ist.

Es wird heute viel diskutiert über Plagiate und Urheberrechte. Es soll aber keineswegs der Eindruck entstehen, es handle sich hier um eine Rechtsverletzung. Die Autoren des abgekupferten Projektes haben meines Wissens keinerlei Ansprüche auf Urheberrechte oder finanzielle Vergütung für die industrielle Verwertung ihrer Ideen erhoben. Aber die Fairness gebietet, dass Ehre erwiesen wird, wem Ehre gebührt.

Marcel „Derri“ Derrmann

Sehr geehrter Herr Derrmann,
vielen Dank für Ihre detaillierten Informationen. Sie haben natürlich Recht, das wir hier das Rad nicht neu erfunden haben. Es gebührt den Entwicklern dieses Projekts großen Respekt für die tolle Idee. Unser Unternehmen entwickelt nicht nur eigene Ideen, sondern setzt auch tolle Ideen in seiner bekannten Qualität zu einem möglichst attraktiven Preis um, damit möglichst viele Menschen daran teilhaben können. Hier gibt unsere Entwicklungsabteilung dann oft den „letzten Schliff“ hinzu.

Volker Bode, SiMAC/Joy-IT

Balkonkraftwerk

Heft 09-10/2021, S. 44 (210326)

SmartHome-Steckdosen zur Messung des eingespeisten Solarstroms? Diese Idee hatte unser Leser Jörg Trautmann und schrieb uns über seine Erfahrungen: Anfangs war ich etwas enttäuscht, dass die ersten von mir getesteten SmartHome-Steckdosen nicht in der Lage waren, den eingespeisten Solarstrom zu messen. Nach längerem Testen bin ich dann auf die SmartHome-Steckdose LogiSmart PA0066 der Firma LogiLink gestoßen, welche tatsächlich den eingespeisten Solarstrom erfassen kann. Die Modelle von drei anderen Herstellern waren lediglich in der Lage, die Stromaufnahme echter Verbraucher zu erfassen. Nach meinem Kenntnisstand ist die einfachste Variante der Strommessung das Stromzangenprinzip, bei dem die Stromflussrichtung irrelevant für die Messung ist. Hier müssten sich die Modelle von der Art der Messung unterscheiden, was ein sehr interessanter Aspekt ist.

Jörg Trautmann

Möchten Sie selbst etwas beitragen?
Gerne können Sie sich an die Elektor-Redaktion wenden unter der E-Mail-Adresse: redaktion@elektor.de

210560-02

Color to Sound

Wie man einen Farbsensor über I²C ausliest

Von **Annika Schlechter** (Deutschland)
und **Volker Ziemann** (Schweden)

Dieses Lernprojekt zeigt, dass Elektronik für behinderte Menschen hilfreich sein kann. Für diejenigen, die keine Farben sehen oder unterscheiden können, übernimmt ein Farbsensor diese Aufgabe. Zur akustischen Rückmeldung wird ein Summer verwendet, der über Farben und Helligkeit informiert. Wir zeigen zwei Versionen des Projekts, die beide auf dem preiswerten Arduino Nano basieren.

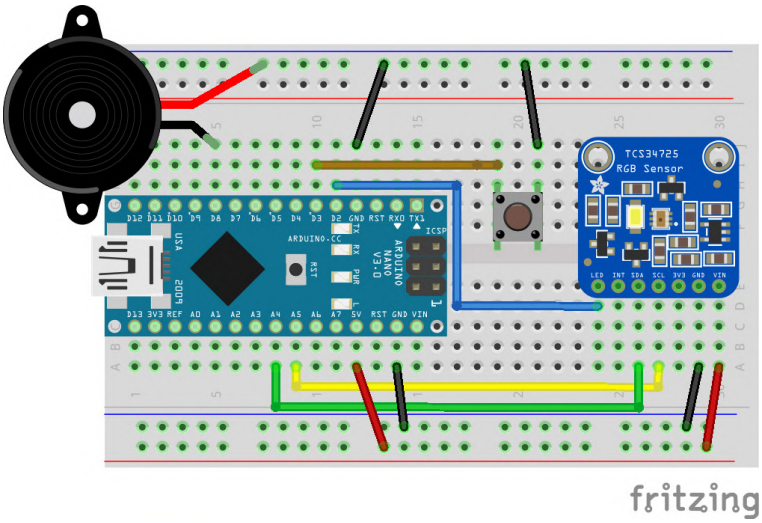


Bild 1. Schaltung des Prototyps. Ein Piezo-Summer und ein TCS34725-Farbsensor, montiert auf einem kleinen Breakout-Board, sind mit einem Arduino Nano verbunden.

Wie kann ein sehbeeinträchtigter Mensch morgens die farblich passende Kleidung auswählen? Die Idee hinter diesem Projekt ist, dass die Farbinformation in ein Tonsignal umwandelt wird, dessen Frequenz die Farbe und dessen Dauer die Helligkeit widerspiegelt.

Bild 1 zeigt den Schaltplan eines Prototyps. Auf einem Steckboard sind ein Piezo-Summer und ein ams-Farbsensor TCS34725 auf einem kleinen Breakout-Board [1] zusammen mit einem Arduino Nano montiert. Ein solcher Prototyp wurde im Rahmen des Kurses „From Sensor to Report" an der Universität Uppsala [2] entwickelt, in dem grundlegende Fähigkeiten zur Datenerfassung einschließlich der Anbindung von Sensoren und Mikrocontrollern an die Außenwelt vermittelt werden. Und weil man nichts lernt, wenn man vorgefertigten Bibliotheken zur Sensoranbindung verwendet, programmieren wir die Grundfunktionen zum Auslesen der Sensoren selbst. Außerdem befindet sich die Logik zur Zuordnung der Töne zu den Farben zunächst in einem Python-Skript auf einem Laptop und nicht auf dem Mikrocontroller. Die Kommunikation zwischen PC und Mikrocontroller erfolgt über eine serielle Schnittstelle. Wir haben ein Kommunikationsprotokoll entwickelt, um die Sensorwerte vom Nano zum PC zu senden; in die andere Richtung gehen Befehle, um den Summer mit der richtigen Frequenz zu aktivieren. Später haben wir die Funktionalität jedoch so erweitert, dass ein Stand-alone-Betrieb auf dem Nano möglich ist. Dennoch handelt es sich hier nicht um ein „serienreifes" System, sondern um einen Prototyp, der die Grundfunktionalität zeigt und zum weiteren Experimentieren einlädt.

Bevor wir mit dem Zusammenbau der Komponenten beginnen, müssen wir das Arduino-Entwicklungssystem (IDE) aus [3] installieren, um Programme für den Nano entwickeln zu können. Die IDE ist für alle gängigen Betriebssysteme verfügbar. Bitte beachten Sie die für Ihr System relevanten Hinweise. Wenn Sie bisher noch nicht mit dem Arduino gearbeitet haben, folgen Sie bitte [4], um sich mit der Arduino-IDE vertraut zu machen. Als zweite Voraussetzung müssen wir Python [5] installieren, das ebenfalls für alle Systeme verfügbar ist. Die Binärdateien für Windows und MAC OS finden Sie unter [5] und nahezu alle Linux-Distributionen haben entsprechende Pakete in ihren Repositories. Auch hier gilt: Folgen Sie den Anweisungen für Ihr System. Aber genug der Vorrede, fangen wir an!

Der Farbsensor

Das Herzstück dieses Projekts ist der Farbsensor TCS34725 [6], der aus einem 3x4-Fotodioden-Array besteht (**Bild 2**). Farbsensitive Filter, die vor den Dioden angebracht sind, machen sie empfindlich für rotes,

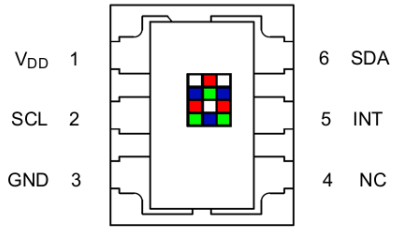


Bild 2. Der Farbsensor TCS34725 enthält ein 3x4-Fotodioden-Array. Farbsensitive Filter, die vor den Dioden montiert sind, machen sie empfindlich für rotes, grünes und blaues Licht. Quelle: ams-Datenblatt [6].

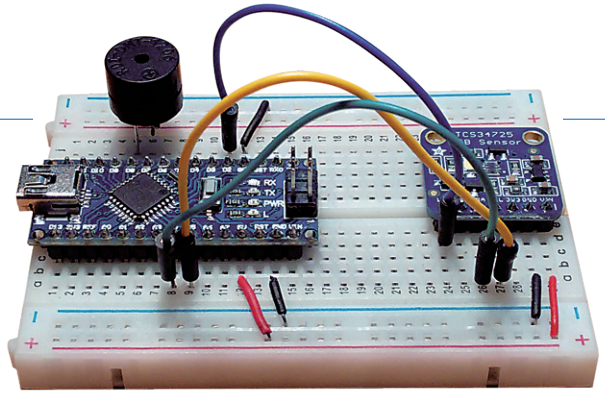


Bild 3. Die Hardware aus Bild 1 ist auf einem lötfreien Breadboard aufgebaut (hier ohne Taster).

PASSENDE PRODUKTE

- **Arduino Nano (SKU 17002)**
  www.elektor.de/17002
- **Breadboard (830 Tie Points) (SKU 17092)**
  www.elektor.de/17092

grünes und blaues Licht. Darüber hinaus liefert eine klare Diode ohne Filter die Information über die Helligkeit. Das analoge Signal der Dioden wird durch Analog-Digital-Wandler auf dem Chip in eine digitale Form umgewandelt und in Registern gespeichert. Diese Farb-/Helligkeitsinformationen werden schließlich über eine I²C-Schnittstelle zur Verfügung gestellt, die mit einem Mikrocontroller verbunden wird. Für die Kommunikation werden nur zwei Drähte benötigt, einer für das Taktsignal (SCL) und einer für die Daten (SDA). Der Farbsensor wird dann durch das Schreiben und Lesen der Register auf dem Sensor-IC gesteuert, aber dazu später mehr, wenn wir die Software beschreiben.
Auf dem Breakout-Board befinden sich außer dem winzigen Farbsensor ein 3V3-Spannungsregler samt Pegelwandler für den I²C-Bus sowie eine vom Controller schaltbare weiße LED.
Die Hardware ist auf einem lötfreien Breadboard aufgebaut (**Bild 3**), ziemlich genau nach Bild 1 mit dem Nano auf der linken und dem Farbsensor-Breakout TCS34725 auf der rechten Seite.
Nano und Breakout-Board sind über schwarze und rote Drähte mit Masse und der 5-V-Schiene verbunden, während der I²C-Bus über den grünen Draht (Daten SDA an A4) und den gelben Draht (Takt SCL an A5) verläuft. Der Nano schaltet über den blauen Draht vom digitalen Ausgangspin D2 zum mit *LED* beschrifteten Pin die besagte weiße LED auf dem Breakout-Board. Der Summer ist mit Pin D8 des Nano verbunden. Der Taster, der über den braunen Draht Pin D3 mit Masse verbindet, löst eine A/D-Wandlung aus, wenn der Nano im Standalone-Modus betrieben wird, wie dies weiter unten besprochen wird.
Die Schaltung wird durch einen Arduino-Sketch zum Leben erweckt, der die *wire.h*-Bibliothek nutzt, die Basisfunktionen zur Kommunikation über die I²C-Leitungen bereitstellt. In unserem Sketch werden die beiden Funktionen `I2Cread()` und `I2Cwrite()` verwendet, die die gesamte Kommunikation mit dem Sensor kapseln.
`I2Cread()` dient zum Auslesen eines Sensorwertes oder Parameters, der in den Registern gespeichert ist. Als Parameter erhält die Funktion die Adresse des Sensors auf dem I²C-Bus (hier 0x29), und die Registeradresse (es hat eine Weile gedauert, bis wir herausgefunden haben, dass zu letzterer immer 0x80 addiert werden muss). Die Funktion gibt den Inhalt des Registers zurück.
Die Funktion `I2Cwrite()` wird verwendet, um Befehle vom Controller an den Sensor zu senden, zum Beispiel, um Konfigurationswerte zu setzen. Diese Funktion erhält die Sensor- und Registeradressen und zusätzlich den neuen Wert, der in dieses Register geschrieben werden soll. Von dieser Funktion wird kein Wert (`void`) zurückgegeben.

Die Anwendung

Kehren wir nun zu unserem Sketch zurück. In der Funktion `setup()` initialisieren wir die serielle Kommunikation, die grundlegende I²C-Funktionalität und den Sensor (über die Funktion `color_begin()`). Innerhalb der Funktion `loop()`, die unendlich oft ausgeführt wird, prüfen wir zunächst, ob ein Befehl vom PC auf der seriellen Leitung angekommen ist. Wenn das der Fall ist, lesen wir die Registerinhalte des Sensors aus, und zwar für alle drei Farben. Für jede Farbe werden zwei 8-Bit-Register ausgelesen und die empfangenen Bytes zu einem 16-Bit-Wort zusammengesetzt - entsprechend der Anleitung aus dem Datenblatt. Dies ist im folgenden Codeschnipsel für die rote Farbe dargestellt:

```
b3=I2Cread(TCS34725,0x16);   // raw data, red
b4=I2Cread(TCS34725,0x17);
red=b4*256+b3;
```

Hier enthält `b3` das niederwertige und `b4` das höherwertige Byte, das wir mit 256 multiplizieren und zu `b3` addieren, um die 16-Bit-Variable `red` zu erhalten. Die anderen Farben werden ähnlich behandelt, nur die Registeradressen sind andere.
Zusätzlich verwenden wir die von der ungefilterten Diode gemeldete Helligkeit und speichern sie in der Variablen `clea`, die die Dauer `d` des Tons bestimmt. Ist der Wert von `clea` kleiner als 1000, wird ein 0,5 s langer Ton ausgegeben, ist er größer, verwenden wir 1,5 s.
Nun lesen wir den empfangenen Befehl und kopieren das Ergebnis in ein Zeichenarray namens `line`, so dass wir mit der Standard-C-Funktion `strstr()` herausfinden können, welcher Befehl gesendet wurde. Der folgende Codeschnipsel zeigt dies:

```
Serial.readStringUntil('\n').toCharArray(line,30);
if (strstr(line,"color?")==line) {
  Serial.print(clea); Serial.print(" ");
  Serial.print(red); Serial.print(" ");
  Serial.print(green); Serial.print(" ");
  Serial.println(blue);
} else if (strstr(line,"tone_red")==line) {
  tone(buzz, 262, d);
...
```

Wir sehen, dass der Befehl *color?* den Nano veranlasst, die Werte der vier Signale `clea`, `red`, `green` und `blue` über die serielle Leitung zurückzusenden. Wird *tone_red* empfangen, wird der eingebaute `tone()`-Befehl verwendet, um den Piezo-Summer zu aktivieren - in diesem Fall mit der Frequenz, die der roten Farbe zugeordnet ist. Es folgen mehrere gleich aufgebaute `strstr()`-Blöcke, die die entsprechenden Aktionen auslösen.
Beachten Sie, dass wir ein einfaches Protokoll verwenden, das auf dem Hin- und Herschicken von Zeichenketten basiert. Eine Anfrage aus Richtung des Laptops wird mit einem Fragezeichen abgeschlossen, zum Beispiel *color?*, und der Nano antwortet, indem er die Werte zurückschickt. Eine Anweisung basiert auf dem einfachen Senden des Befehls, zum Beispiel *tone_red*, wodurch der Summer für die Dauer `d` mit 262 Hz ertönt. Die Kommunikation ahmt ein wenig die SCPI-Befehlssprache nach, die von modernen Oszilloskopen und anderen

Über die Autoren

Volker Ziemanns Interesse an der Elektronik wurde von Elektor geweckt, etwa Mitte der 1970er Jahre, zur Zeit des 40-W-Edwin-Verstärkers. Er wurde aber durch sein Physikstudium abgelenkt und arbeitete seitdem mit Teilchenbeschleunigern - SLAC in den USA, CERN in Genf und jetzt in Uppsala, Schweden. Da Elektronik natürlich eine wichtige Rolle für die Steuerung und Datenerfassung in Beschleunigern spielt, war sein frühes Interesse während seiner gesamten Karriere nützlich. Er lehrt jetzt an der Universität Uppsala und hat drei Bücher verfasst, eines davon über Datenerfassung mit Arduino-Boards und dem Raspberry Pi.

Annika Schlechter studiert mit dem Ziel eines MSc in Physik an der Universität Heidelberg. Im Moment setzt sie ihr Studium im Rahmen ihrer Bachelorarbeit am dortigen Deutschen Krebsforschungsinstitut fort. Dieses Projekt begann während eines Erasmus-Semesters an der Universität von Uppsala, wo sie Volkers Vorlesung „Sensor to Report“ besuchte.

Listing 1. Python-Skript für die Kommunikation zwischen Laptop und Arduino Nano

```
# colortosound_arduino.py
import serial, time

ser=serial.Serial("/dev/ttyUSB0",9600,timeout=1)
time.sleep(3);

ser.write(b"color?\n")  # write command to get colors
time.sleep(2);

reply=ser.readlines()   # read answer

# process answer to get color values
colors = reply[0].split()
for i, entry in enumerate(colors):
    colors[i]= int(entry.decode('utf-8').replace('\r\n', ''))

intensity = colors [0]; red = colors [1]
green = colors [2]; blue = colors [3]

# write command to control buzzer according to color

if red/intensity > 0.6:
    ser.write(b"tone_red\n")
elif green/intensity > 0.3:
    ser.write(b"tone_green\n")
elif blue/intensity > 0.24:
    ser.write(b"tone_blue\n")
else:
    ser.write(b"no_color\n")

ser.close()
```

Test- und Messgeräten unterstützt wird. Dies macht die Anbindung an externe Programme, die den Zugriff auf die serielle Leitung unterstützen, sehr einfach: Python, Octave, Matlab und Labview unterstützen dies.

Das PC-Programm

Wir haben Python 3 auf dem Laptop verwendet, um mit dem Nano zu kommunizieren. Der sehr einfache Code ist in **Listing 1** dargestellt.

Nach dem Importieren der Unterstützung für die serielle Kommunikation und der grundlegenden Timing-Funktionalität öffnen wir den seriellen Port zum Nano mit der passenden Baudrate. Wir geben dem Betriebssystem 3 s Zeit, um die Verbindung herzustellen und senden dann den Befehl *color?*.
Beachten Sie, dass Python 3 Zeichenketten als Unicode kodiert, während der Nano reine ASCII-Zeichenketten erwartet. Wir müssen also dem Befehl *color?* den Buchstaben „b“ voranstellen, um die Zeichenkette als reine ASCII-Daten zu senden. Nach einer weiteren kurzen Wartezeit erhalten wir die Antwort des Nano in Form der Variablen `reply`, die die Werte der vier Farben enthält. Mit der Methode `.split()` rufen wir die Werte einzeln ab und ordnen sie mnemotechnischen Namen wie `red` oder `intensity` zu.
Nachdem wir die Werte zur Verfügung haben, können wir die Logik implementieren, die die Töne den Farben zuzuordnet. Dazu haben wir in einigen Experimenten mit farbigen Papierbögen einem hellen, mittleren und dunklen Ton für jede der drei Grundfarben eine gute Zuordnung gefunden und den Sensor kalibriert. Wir haben die jeweiligen Farbwerte in einem festen Abstand wiederholt gemessen und diese Informationen verwendet, um die Konstanten in dem oben gezeigten Python-Code festzulegen. Zum Beispiel wird der Befehl *tone_red* an den Nano gesendet, wenn der Rotanteil, normiert auf die Intensität, größer als 0,6 ist. Als nächstes wird die Grünintensität getestet und *tone_green* gesendet, wenn der Schwellwert überschritten wird. Schließlich wird der Wert der Blaukomponente getestet. Wenn auch das nicht ausreicht, wird *no_color* an den Nano gesendet, was ihn veranlasst, den entsprechenden Ton auszugeben. Sicherlich ist etwas Experimentieren mit den Konstanten erforderlich, um das System an die Garderobe anzupassen.

Standalone-Version

Die Standalone-Version des Codes ist auf der Website dieses Artikels verfügbar [7]. Sie lehnt sich eng an das obige Beispiel an, mit dem Unterschied, dass der Nano nun nicht mehr auf einen Befehl auf der seriellen Leitung wartet, sondern auf das Drücken der Taste, bevor er den Sensor ausliest und den entsprechenden Ton ausgibt. Außerdem wird die Farbinformation direkt auf dem Nano verarbeitet, wie im folgenden Schnipsel gezeigt:

```
if (red/clea > 0.6) {
  tone(buzz, 262, d);
} else if (green/clea > 0.3) {
...
```

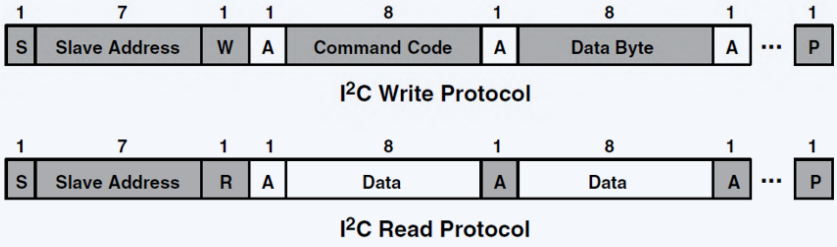
Wenn es wünschenswert ist, die Kleidung mit der eingebauten LED auf dem Breakout-Board zu beleuchten, können wir sie mit `digitalWrite(led,HIGH)` einschalten, kurz bevor wir die Register aus dem TCS34725 lesen, und sie danach wieder ausschalten.
In einem dritten Schritt können wir das System portabel machen, indem wir einen nackten ATmega328, geflasht mit einem Arduino-Bootloader, an den Sensor anschließen. Das System wird von einer Batterie mit Strom versorgt,

I²C-Kommunikation

Die I²C-Kommunikation zwischen zwei integrierten Schaltungen basiert auf einem synchronen seriellen Protokoll, das über zwei Drähte übertragen wird. Es folgt der einfachen Regel, den Spannungspegel an einem der Drähte, genannt SDA, zu erfassen, nachdem sich die Spannung am anderen Draht, genannt SCL, geändert hat. Ein solcher Lesevorgang liefert ein einzelnes Bit, acht aufeinanderfolgende Bits bilden ein Byte. Einer der Teilnehmer am I²C-Bus (Master), in der Regel ein Mikrocontroller, leitet die Kommunikation und liefert das Taktsignal. Die anderen Teilnehmer (Slaves) können nur in die Kommunikation einsteigen, wenn sie angesprochen werden und das Recht erhalten, ihre Informationen auf SDA zu legen. Außerdem erhält der Slave nach jedem Byte einen Taktzyklus Zeit, um den Empfang des vorherigen Bytes zu quittieren.

Um die Kommunikation mit einem Sensor einzuleiten, sendet der Mikrocontroller ein Byte. Die ersten sieben Bits dieses Bytes enthalten die Adresse des Sensors und das letzte Bit zeigt an, ob anschließend der Controller oder der Sensor die Kontrolle über SDA erhält, um das folgende Byte zu senden. Will der Controller weiter senden, um zum Beispiel ein Register des Sensors zu ändern, sendet er zwei weitere Bytes, die Registeradresse und den neuen Wert. Möchte der Controller dagegen ein Register vom Sensor lesen, sendet er das Bit, das anzeigt, dass nun der Sensor für SDA zuständig ist, und liest SDA synchron mit dem Taktsignal, das der Mikrocontroller weiterhin erzeugt.

Die obere Abbildung veranschaulicht das Schreiben auf den Sensor. Die dunkel schattierten Bits werden vom Mikrocontroller gesendet und die nicht schattierten Bits vom Sensor. Um eine Transaktion einzuleiten, erzeugt der Mikrocontroller eine Startbedingung, bezeichnet mit „S“, gefolgt von den sieben Bits der Adresse des Sensors und dem achten „W“-Bit, das gesetzt wird, um anzuzeigen, dass der Controller weiter senden möchte. Ein Bit „A“ wird vom Sensor bereitgestellt, um anzuzeigen, dass er verstanden hat. Das zweite Byte enthält die Adresse des Registers des Sensors und ein Byte mit dem Wert, der in das Register geschrieben werden soll. Beachten Sie, dass der Sensor jedes empfangene Byte mit „A“ quittieren muss. Das Lesen vom Sensor, dargestellt durch das untere Diagramm, funktioniert fast genauso, nur dass das achte „R“-Bit des ersten Bytes nun anzeigt, dass der Sensor das Recht zum Senden erhält und SDA anschließend vom Sensor gesteuert wird, während danach der Mikrocontroller jedes empfangene Byte quittieren muss.



Quelle: ams AG, Datenblatt TCS34725 [6]

solange ein Knopf gedrückt wird, startet es und wandelt kontinuierlich Farbe in Ton um, bis der Knopf losgelassen wird, um die Batterie abzutrennen. Diese Software ist ebenfalls auf der Webseite des Artikels verfügbar. Der Bau des Systems wird als Übung den interessierten Lesern überlassen.

Im Kurs *Sensor to Report* war der Farbsensor recht beliebt. Eine Studentin, die die Farbe des Himmels über einen längeren Zeitraum überwachen wollte, um die Daten später zu analysieren, verwendete im Wesentlichen den gleichen Aufbau, wobei der Nano über die serielle Leitung mit einem Host-Computer kommunizierte. Sie verwendete ein Python-Skript auf einem Raspberry Pi, um die Daten in einer MySQL-Datenbank zu speichern, die ebenfalls auf dem Pi läuft, und nutzte später Octave, um Plots der Farben zu erstellen, die über etwa eine Woche aufgezeichnet wurden. Die Plots wurden kurz vor Weihnachten aufgenommen und machten uns schmerzlich bewusst, wie kurz die Tage in Uppsala im hohen Norden werden können. Weitere Projekte zur Datenerfassung, die demselben Geist folgen, sind in [8] beschrieben.

210051-02

Haben Sie Fragen oder Kommentare?

Haben Sie technische Fragen zu diesem Artikel? Bitte senden Sie eine E-Mail an die Redaktion unter redaktion@elektor.de.

Ein Beitrag von

Entwurf und Text: **Annika Schlechter, Volker Ziemann**
Redaktion: **Jens Nickel**
Übersetzung: **Rolf Gerstendorf**
Layout: **Sylvia Sopamena**

WEBLINKS

[1] TCS34725 Breakout-Board: https://www.adafruit.com/product/1334
[2] V. Ziemann, Kurs „Sensor to Report“: https://ziemann.web.cern.ch/ziemann/teaching/s2r19/
[3] Arduino-Webseite: https://www.arduino.cc/
[4] Arduino-Tutorials: https://www.arduino.cc/en/Tutorial/HomePage
[5] Python-Website: https://www.python.org/
[6] Datenblatt Farbsensor TCS34725: https://ams.com/tcs34725#tab/documents
[7] Software Download: http://www.elektormagazine.de/210051-01
[8] V. Ziemann, „A Hands-on Course on Sensors using the Arduino and Raspberry Pi“, CRC Press, 2018.

BattLab-One

Betriebsdauer von batteriebetriebenen Geräten messen und optimieren!

Von **Tam Hanna**

Wie erfasst man den Energiebedarf mikrocontroller-gesteuerter Geräte und wie kann man abschätzen, wie lange solch ein Gerät mit einem bestimmten Batterie- oder Akkutyp läuft? Für die Beantwortung von Fragen dieser Art gibt es spezielle Tools. *BattLab One* ist genau für diese Zwecke konzipiert. Das nachfolgende Review beantwortet ein paar Fragen dazu, wie man mit so einem Messgerät umgeht und was uns bei der Inbetriebnahme aufgefallen ist.

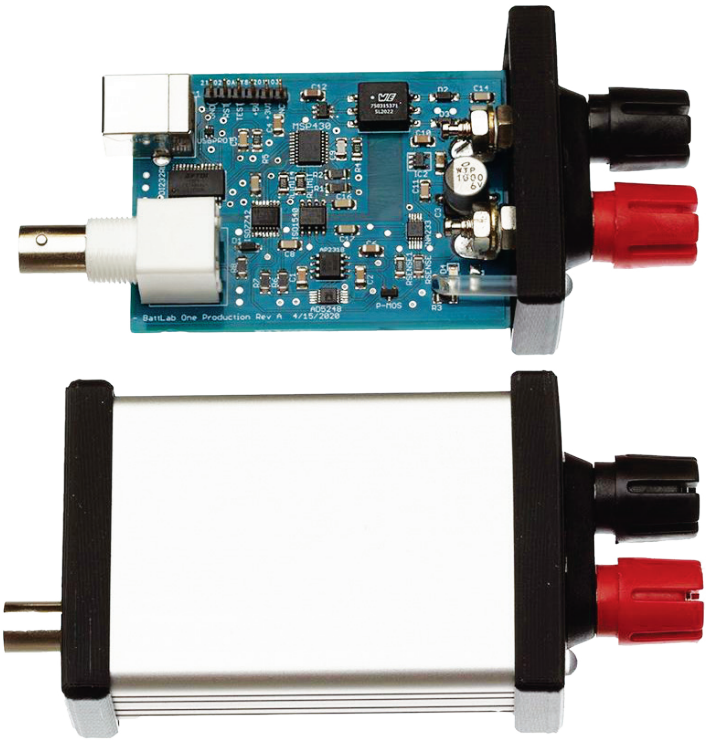


Bild 1. Die Elektronik des BattLab One, vor und nach dem Einbau in ein Alugehäuse.

Die Erfassung des Energieverbrauchs von Geräten auf Mikrocontroller-Basis ist keine triviale Aufgabe. Wie bei jedem Computer-Chip hängt der benötigte Strom stark von der aktuellen Rechenlast und vom Betriebszustand ab. Dabei ist nicht nur an Normalbetrieb und Standby oder Deep-Sleep-States zu denken, bei denen die Stromaufnahme gängiger Mikrocontroller zwischen weniger als 1 µA und mehreren hundert Milliampere variieren kann. Auch eine zu gewissen Zeiten aktive oder inaktive Peripherie haben wechselnde Verbräuche zur Folge. Um diesem dynamischen Geschehen Rechnung zu tragen und eine halbwegs realistische Abschätzung der Betriebsdauer mit verschiedenen Batterie- oder Akkutypen vornehmen zu können, ist ein spezialisiertes Tool wie BattLab One sehr hilfreich.

Was ist BattLab One?

Im Bereich der Stromverbrauchs-Messungen haben sich seit längerer Zeit sogenannte SMUs (Source Measuring Units) bewährt. Darunter versteht man Messsysteme, die einerseits als Netzgerät zur Versorgung des Zielsystems dienen, und andererseits mehr oder weniger aufwändige Messungen des Stromverbrauchs über bestimmte Zeitspannen durchführen. Klassiker in diesem Bereich sind die von Keithley und Keysight gefertigten Batteriesimulatoren.

BattLab One zielt auf eine ähnliche Anwendung, liegt aber mit einem Preis von unter 100 € in einer anderen Liga. Elektor verkauft das Gerät [1] in der im Metallgehäuse eingebauten Variante. In **Bild 1** ist links die USB-Typ-B-Buchse nebst einer BNC-Buchse als Trigger-Eingang zu sehen. Rechts sind zwei 4-mm-Buchsen platziert, an die das DUT (Device Under Test) angeschlossen werden kann. **Bild 2** zeigt, dass sich deren Kunststoffisolation vorschrauben lässt, so dass man leicht Leitungen mit Krokodilklemmen anschließen kann. Ein genauer Blick bringt zum Vorschein, dass die beiden schwarzen Endkappen des Aluprofils wohl per 3D-Druck hergestellt wurden.

BattLab One wird via USB versorgt und ist in der Lage, Geräte mit verschiedenen typischen Spannungen von 1,2 V bis 4,5 V und Strömen bis maximal 450 mA zu versorgen. Mitgeliefert wird eine Gruppe von Entladungsprofilen häufig verwendeter Akkus mit Li-Ion-, $LiFePO_4$-, Alkali-, NiMh- und NiCd-Chemie, die für die Berechnungen verwendet werden. Weitere Feinheiten wie last- und alterungsabhängige Spannungsabfälle berücksichtigt BattLab One nicht. Laut Hersteller bietet das Messgerät eine grundlegende Batterie-Simulation und einen von 10 µA bis 500 mA reichenden Strommessbereich. Integriert ist ein ADC mit 16 bit Auflösung bei einer Abtastrate von 1 kHz.

Bild 2. Anschluss von Kabeln mit Krokodilklemmen an die Bananenbuchsen des BattLab One.

BattLab One und Linux

Der Hersteller Bluebird Labs entwickelte die zum Gerät gehörende Software unter Python. Als ersten Versuch wollte ich das System daher direkt unter Ubuntu 20.04 LTS zum Laufen bringen. Der Quellcode der Software steht bei GitHub [2] bereit und kann mit den üblichen Verfahren kompiliert werden. Allerdings geht Bluebird Labs mit *pipenv* einen Sonderweg. Wer wie ich nur die gewöhnliche Variante *venv* installiert hat, muss *pipenv* daher nachinstallieren. Bei mir ging das wie folgt:

```
tamhan@TAMHAN18:~/BattLabonespace/
   BattLab-One$ sudo apt install pipenv
```

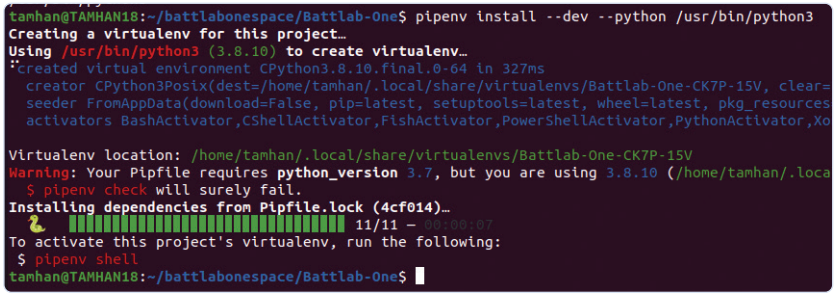


Bild 3. PipEnv-Fehlermeldung.

Bei der Einrichtung des Virtual Environments gab es erste Probleme, da Python in der Version 3.7. vorausgesetzt wird. Will man wie ich eine neuere, zum Beispiel die unter Ubuntu 20.04 LTS vorinstallierte Version 3.8 verwenden, ist noch folgendes notwendig:

```
tamhan@TAMHAN18:~/BattLabonespace/BattLab-One$ which
   python3
/usr/bin/python3
tamhan@TAMHAN18:~/BattLabonespace/BattLab-One$ pipenv
   install --dev --python /usr/bin/python3
Creating a virtualenv for this project…
```

Während der Ausführung der folgenden Kommandos wird sich die Software permanent beschweren (siehe **Bild 3**). Deshalb wird das Virtual Environment nach folgendem Schema aktiviert. Es bleibt wie ein normales *venv*-Environment aktiv, bis das Terminalfenster geschlossen wird:

```
tamhan@TAMHAN18:~/BattLabonespace/BattLab-One$ pipenv
   shell
. . .
```

Eigentlich sollte die Ausführung der drei .py-Dateien mit der Software nun nach folgendem Schema möglich sein:

```
(BattLab-One-CK7P-15V) tamhan@TAMHAN18:~/
   BattLabonespace/BattLab-One$ python3
   BattLab-Release_V1.2.1.py
Traceback (most recent call last):
  File "BattLab-Release_V1.2.1.py", line 59, in
   <module>
    import pkg_resources.py2_warn
ModuleNotFoundError: No module named 'pkg_resources.
   py2_warn'
```

Leider hält sich Blue Bird Labs nicht an die Vorgaben der Setuptools, sondern inkludiert eine Bibliothek explizit, die nur in wenigen Versionen der Setuptools zur Verfügung steht. Daher wird die .py-Datei geöffnet und die folgende Zeile entfernt:

```
import pkg_resources.py2_warn
```

Jetzt hat die Software ein Problem mit den lokalen Pfaden:

```
(BattLab-One-CK7P-15V) tamhan@TAMHAN18:~/
   BattLabonespace/BattLab-One$ python
   BattLab-Release_V1.2.1.py
 . . .
```

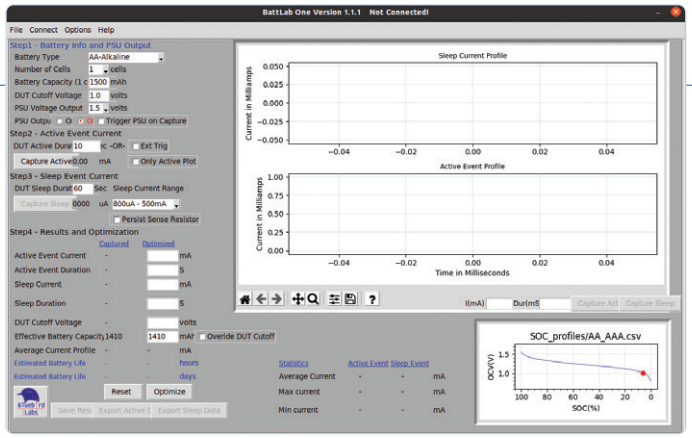


Bild 4. So zeigt sich die Oberfläche der BattLab-Software.

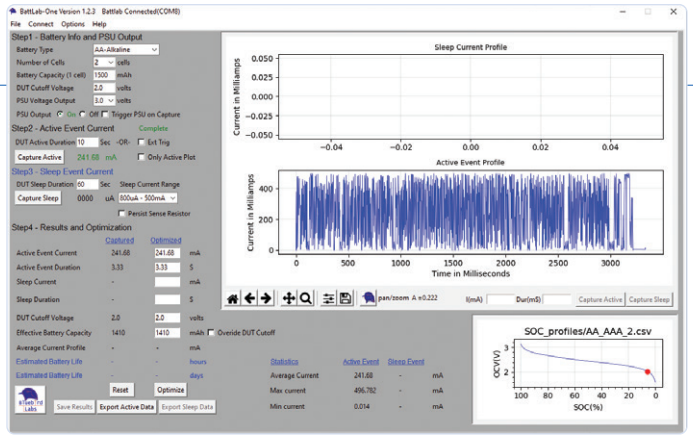


Bild 5. BattLab One erfasst und zeigt den Stromverbrauch eines DUT an.

```
_tkinter.TclError: error reading bitmap file "icons\
    bbirdlogo.xbm"
```

Die Lösung besteht darin, das TK-Inter-GUI durch Entfernung der Zeile `root.iconbitmap(bitmap=GetIconPath())` vom Programmsymbol zu befreien. Bei dieser Gelegenheit sollte man nach folgendem Schema eine Anpassung des Datei-Pfads durchführen:

```
img = PhotoImage(file='/home/tamhan/BattLabonespace/
    BattLab-One/icons/bbirdlogo_png1.png')
```

Der nächste Ausführungsversuch zeigt dann, dass die TK-Inter-Umgebung im falschen Ordner nach der Datei *bblogo.gif* sucht:

```
_tkinter.TclError: couldn't open "/home/tamhan/.
    local/share/virtualenvs/BattLab-One-CK7P-15V/
    lib/python3.8/site-packages/matplotlib/mpl-data/
    images/bblogo.gif": no such file or directory
```

Man kopiere daher eine beliebige GIF-Datei in dieses Verzeichnis und gebe ihr den passenden Namen. Wenn man jetzt noch die Zeile auskommentiert

```
#toolbar.children['!button8'].
    config(command=select_range)
```

kann man das Programm mit leicht eingeschränktem Funktionsumfang starten (siehe **Bild 4**). Die Deaktivierung des Messbereichsknopfs erweist sich in der Praxis als unkritisch, weil sich das MatPlotLib-Diagramm bezüglich Usability sowieso problematisch gibt. So erscheinen viele Labels abgeschnitten und die Größe des Fensters lässt sich nicht anpassen.
Wie noch gezeigt wird, gestaltet sich die Verwendung unter Windows 10 angenehmer. Vor einem Betriebssystemwechsel kann man das angelegte Virtual Environment aber noch beseitigen, um Massenspeicherplatz zu sparen:

```
(BattLab-One-CK7P-15V) tamhan@TAMHAN18:~/
    BattLabonespace/BattLab-One$ exit
exit
tamhan@TAMHAN18:~/BattLabonespace/BattLab-One$
    cd /home/tamhan/.local/share/virtualenvs/
    BattLab-One-CK7P-15V
tamhan@TAMHAN18:~/.local/share/virtualenvs/BattLab-
    One-CK7P-15V$ rm * -rf
tamhan@TAMHAN18:~/.local/share/virtualenvs/BattLab-
    One-CK7P-15V$ ls
```

BattLab One unter Windows 10

Hier ist die Situation insofern einfacher, als fertige .exe-Dateien unter [3] bereitgestellt werden. Das Zip-Archiv enthält unter anderem den Unterordner *SOC_profiles*, der Batterie-Profile in Tabellenform enthält. Man muss das Archiv daher entzippen.
Ein Start der Datei *BattLab-Release_V1.2.3.exe* führt zur schon von Bild 4 bekannten Anzeige. Die Windows-Software verhält sich weitestgehend analog zur Linux-Version, weist aber weniger Rendering-Fehler auf. Als Erstes wählt man oben links den Batterie-Typ aus. Die Daten der Profile dienen zur Anzeige des unten rechts eingeblendeten Diagramms. Rechts neben dem Feld *PSU Output* kann man den im Gerät integrierten Spannungswandler aktivieren, um das an die beiden Buchsen angeschlossene DUT zu versorgen.
Ich arbeite seit längerer Zeit am Projekt *HygroSage*, einem IoT-Sensor, der ob seines Farbdisplays und seines leistungsstarken Prozessors eigentlich ein gutes Messobjekt für BattLab One sein sollte. Also habe ich den Sensor angeschlossen. Dann kam ein Klick auf die Option *Capture Active*, woraufhin die Software einen flackernden Fortschrittsbalken einblendete und sich 10 s lang nichts Sichtbares tat. Unsichtbar erfasst das System in dieser Zeit Informationen und zeigt sie anschließend wie in **Bild 5** zu sehen an.
Die Messwerte kann man sich in den Diagrammen via Cursor anzeigen lassen. In der Praxis funktioniert dies aber nicht ganz so überzeugend, weil die in der Textbox unten eingeblendeten Werte nicht dem Messwert, sondern dem Mauszeiger folgen. Man muss also den Mauszeiger daher auf den Mittelpunkt der Kurve legen. Außerdem gibt es keinen wirklich bequemen Weg, um in das (per MatPlotLib erzeugte) Diagramm zu zoomen. Trotz einer Full-HD-Auflösung meines Bildschirms ließ sich das Programm nicht maximieren, was der Benutzerfreundlichkeit abträglich ist. Außerdem schaltete ich testweise ein kalibriertes und hochwertiges Multimeter in Reihe zum DUT. Dabei ergab sich ein Strom um die 32 mA. BattLab One stimmte damit nicht wirklich überein

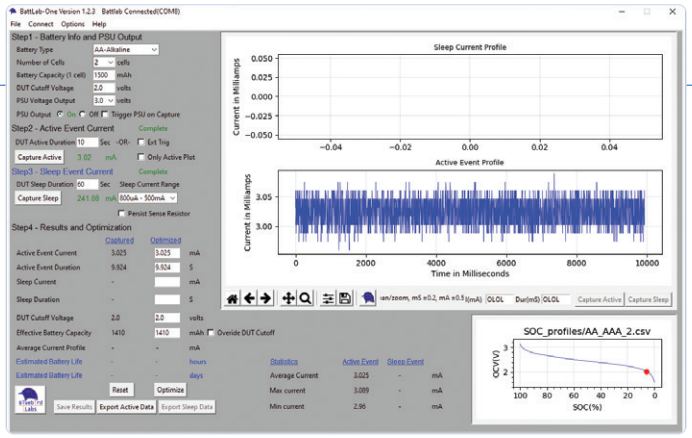


Bild 6. Die Kurve mit dem Strom eines 1-kΩ-Widerstands an 3 V stimmt zwar, ist aber leicht verrauscht.

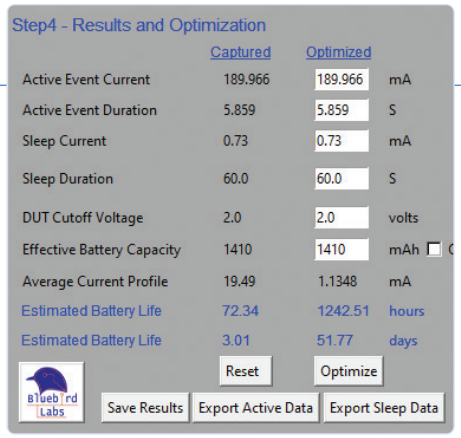


Bild 7. Die Erfassung des Energieverbrauchs im Schlafzustand des DUT ist befriedigend.

– offensichtlich produzierte der Schaltreglers des Typs MCP1640 im DUT Störungen, die BattLab One auch tatsächlich stören. Als zweiter Test wurde BattLab One bei einer Ausgangsspannung von 3 V mit einem 1-kΩ-Widerstand belastet. Es ergab sich die leicht verrauschte Kurve von **Bild 6**. Das Parallelschalten eines 100–µF-Elkos zum HygroSage führte in beiden Fällen nicht zu einer wesentlichen Glättung.

Optimierung des Stand-by-Stromverbrauchs

Systeme mit relativ konstanten Versorgungsströmen sollten sich auch mit preiswerten Multimetern problemlos erfassen lassen. Spannender wird es daher, wenn es um die Analyse des Stand-by-Modus geht. Die zugehörige Option der Software ist *Step3*. Man erfasst das Verhalten im Sleep-Modus manuell durch Anklicken von *Capture Sleep*. Außerdem kann man hier den Strombereich zwischen *10uA - 800uA* oder *800uA - 500mA* umschalten. Im niedrigen Messbereich darf man aber nicht zu viel Strom ziehen. Laut Dokumentation führen Lastströme über 250 mA zur Beschädigung der Hardware.

Ein Test mit Hygrosage zeigt die Resultate von **Bild 7**. In *Step4* zeigt das System dann errechnete Werte an. Dem Hygrosage wurde vom BattLab One eine Batterielebensdauer von nur drei Tagen bescheinigt. Praktische Versuche kamen auf andere Laufzeiten im Bereich von Wochen oder gar Monaten. Ursache ist, dass der im Feld *Active Event Duration* eingestellte Wert quasi in die Ewigkeit interpoliert. Bei vernünftigen Werten fallen die Ergebnisse auch sofort vernünftiger aus. Dabei muss man dann permanent auf *Optimize* klicken, um die Software zur Aktualisierung der Vorhersage-Werte zu animieren.

BattLab One ist aufgrund der vergleichsweise langen möglichen Registrierungsdauer in der Lage, Probleme durch spontanes Aufwachen eines DUTs zu erfassen. Problematisch ist dabei allerdings, dass BattLab One nur einen kleinen Teil der Messzeit tatsächlich registriert. Fällt das Ereignis nicht in eine Registrierungsperiode, so wird es nicht erfasst. Und genau das ist nicht unwahrscheinlich.

Fazit

Die Hardware des BattLab One ist durchaus brauchbar. Leider gilt dies nicht in gleichem Maße für die Software. Ich selbst musste dabei öfter umdenken, da ich professionellere Lösungen gewohnt bin. Andererseits ist es ob des quelloffenen Designs (mit etwas Aufwand) möglich, eine eigene, bessere Software zu kreieren. Bezüglich Hardware macht BattLab One vieles richtig und nur wenig falsch – der Preis ist zudem gemessen an der Leistung wirklich gering.

210473-02

Sie haben Fragen oder Kommentare?

Bei technischen Fragen können Sie sich gern an den Autor wenden unter tamhan@tamoggemon.com oder an die Elektor-Redaktion unter der E-Mail-Adresse redaktion@elektor.de.

Ein Beitrag von

Durchführung und Text: **Tam Hanna**
Redaktion: **Dr. Thomas Scherer**
Layout: **Giel Dols**

PASSENDE PRODUKTE

- **BattLab-One – Batterielebensdauer-Optimierer (SKU 19757)**
  www.elektor.de/19757

WEBLINKS

[1] BattLab One: http://www.elektor.de/19757
[2] Software bei Github: https://github.com/petersdw1/BattLab-One.git
[3] Software für Windows 10: https://github.com/petersdw1/BattLab-One/releases/tag/V1.2.3

Einfacher **Erdschlussprüfer**

Isolationstester für Netzspannungsinstallationen

Von **Elbert Jan van Veldhuizen** (Niederlande)

Fehlerstromschutzschalter sind unverzichtbare Sicherheitseinrichtungen in unserer Hausnetzinstallation, aber manchmal können sie ohne ersichtlichen Grund auslösen. Ein professionelles Erdungswiderstandsmessgerät hilft bei der Lokalisierung eines Lecks, aber der in diesem Artikel vorgestellte, einfach zu bauende Erdschlussprüfer erweist sich als erschwingliche, nützliche und vor allem sichere Alternative zu einem teuren Profi-Gerät. Zur Hochspannungserzeugung kommen dabei handelsübliche CR2032-Lithiumbatterien zum Einsatz.

WARNUNG! Das Arbeiten an der Netzspannung kann lebensgefährlich sein. Bauen Sie die hier beschriebene Schaltung nicht und verwenden Sie sie nicht, wenn Sie keine Erfahrung im Umgang mit Netzspannungen haben!

Fehlerstrom-Schutzschalter haben schon viele Menschen vor tödlichen Stromschlägen bewahrt. Eine Fehlerstrom-Schutzeinrichtung (Residual Current Device, RCD), genauer gesagt, ein Differenzstrom-Schutzschalter (vulgo FI-Schalter) erkennt eine Stromdifferenz zwischen einem stromführenden (L) und dem neutralen (N) Leiter. Beträgt diese Differenz, die am N-Leiter „vorbei" zur Erde fließt, mehr als 30 mA, so schaltet der Fehlerstromschutzschalter ab. Ein Wechselstrom von etwa 30 mA durch den menschlichen Körper kann potenziell einen Herzstillstand oder andere schwere Schäden verursachen, wenn er länger als den Bruchteil einer Sekunde andauert. FI-Schutzschalter sind deshalb so ausgelegt, dass sie die Leitungsadern nicht nur sicher bei 30 mA, sondern auch schnell genug abschalten, um schwere Verletzungen zu verhindern [1].

Ein Fehlerstrom entsteht nicht nur, wenn eine Person den stromführenden Leiter berührt, auch Feuchtigkeit (etwa in einer Außensteckdose oder einer Lampe) kann Leckströme verursachen. Die Quelle eines solchen Problems ist mitunter schwer zu lokalisieren. Eine Methode besteht darin, die verschiedenen Drähte in den Verteilerdosen der Reihe nach ab- und wieder anzuklemmen, um zu sehen, ob der Fehlerstromschutzschalter immer noch auslöst. Dazu muss natürlich jedes Mal der Strom ab- und wieder angeschaltet werden, es besteht das Risiko, Fehler in die Installation einzubauen, versehentlich an stromführenden Teilen zu arbeiten und zu guter Letzt sind Verteilerdosen nicht immer gut erreichbar. Zudem treten manche Erdschlussprobleme nur sporadisch auf und nicht alle Fehler sind leicht reproduzierbar.

Do It Like A Pro!

Was also tun? Der Entwickler dieses Projekts hatte Probleme mit der elektrischen Anlage im Wintergarten seines Elternhauses: Der verantwortliche FI-Schalter löste nur gelegentlich und ohne ersichtlichen Grund aus. Anstatt nun ein Wochenende mit seinen Messgeräten vor dem FI-Schalter zu verbringen, der – so will es Murphy's Gesetz – dann garantiert nicht auslöst, beschloss der Autor, den professionellen Weg zu gehen und den Ableitstrom durch den unerwünschten Strompfad zu messen. Auf diese Art kommt man dem potenziellen Problem viel leichter auf die Schliche, auch wenn der Ableitstrom unterhalb des RCD-Auslösestroms liegt.

Wie man misst

Bei Isolationsproblemen in den in den DACH-Staaten üblichen TN-C-S-Netzen funktioniert die Widerstandsmessung mit einem Standardmultimeter (im höchsten Widerstandsmessbereich) einfach nicht. Jeder, der schon einmal den Widerstand seiner Haut mit einem Multimeter gemessen hat, kennt das Paradoxon: Es zeigt einen Widerstand von etwa 1 MΩ an (je nachdem, wie feucht oder verschwitzt die Hände sind, höher oder niedriger). Das würde bedeuten, dass bei einer Berührung der 230-V-Netzspannung ein Strom von weit unter 1 mA fließen würde, was keineswegs tödlich, sondern nicht einmal spürbar wäre. Der gemessene Widerstandswert ist jedoch sehr stark von der Spannung abhängig, die zur Widerstandsmessung angelegt wird, bei den meisten Multimetern sind es nicht mehr als 9 V. Bei 230 V liegt der Widerstand des menschlichen Körpers jedoch im Kiloohm-Bereich, was zu Strömen von einigen hundert Milliampere führt, die durchaus schädlich oder sogar tödlich sein können. Das verstößt übrigens keinesfalls gegen das Ohmsche Gesetz, denn diese Spannungsabhängigkeit des Widerstandes wird unter anderem durch elektrochemische Reaktionen im menschlichen Körper verursacht.

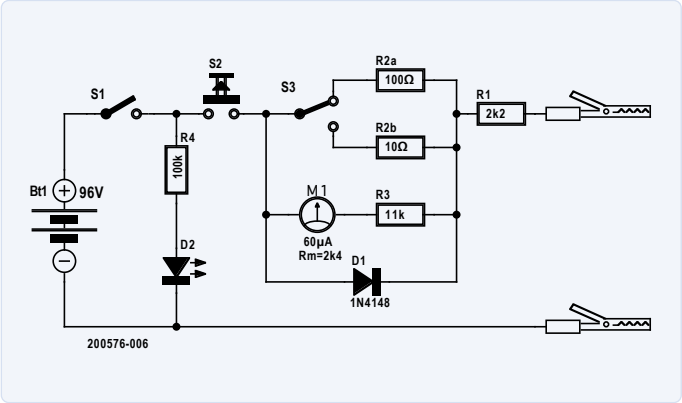


Bild 1. Die einfache Schaltung des Erdschlussprüfers.

Um den Ableitstrom genau zu messen, darf man also nicht wie ein Multimeter mit einer niedrigen Spannung messen, sondern idealerweise mit der ohnehin zur Verfügung stehenden Netzspannung von 230 V. Genau dies tun professionelle Erdungswiderstandsmessgeräte, und genau dies ist auch der Grund für den hohen Preis solcher Messgeräte: die elektrische Sicherheit! Die Herausforderung für den Autor bestand also darin, eine hohe Prüfspannung auf kostengünstige und vor allem sichere Weise zu erzeugen, nämlich mit Batterien.

Praktisch und sicher: die CR2032-Batterie

Der Autor ist einen wohlausgewogenen Kompromiss zwischen einer möglichst hohen Prüfspannung, der Anzahl und des Typs der Batterien eingegangen. Die CR2032 ist eine 3-V-Lithium-Batterie, die, wie der Name schon sagt, einen Durchmesser von 20 mm aufweist, 3,2 mm dick und leicht und sehr preiswert erhältlich ist, da sie häufig in tragbaren Elektrogeräten, Spielzeugen und Gadgets verwendet wird. Diese Batterie besitzt etwa den gleichen Durchmesser wie eine Standard-C-Batterie (Baby-Zelle), die etwa 50 mm lang ist. 16 CR2032-Batterien übereinander gestapelt haben ungefähr den gleichen Formfaktor wie eine C-Batterie und eine Spannung von 48 V. Mit zwei solcher „C-Batterien" in Reihe in einem Standard-Dual-C-Zellen-Batteriehalter erhalten wir mechanisch gut handhabbare 96 V! Diese Spannung sollte ausreichend für eine Messung des Ableitstroms sein. Es sind auch andere Batteriepacks für das gleiche Geld denkbar, etwa 64 LR44s oder acht A23s, aber es gibt keine handelsüblichen Batteriehalter für diese Batteriestapel.

Eine weitere Überlegung für die Verwendung der CR2032 war die Tatsache, dass dieser Batterietyp laut Datenblatt nur einen relativ geringen Strom liefern kann, in der Größenordnung von 10 mA, obwohl es schwierig ist, hierzu schlüssige Daten zu finden. Denn auch eine Spannung von 96 V ist potentiell gefährlich, bei Gleichstrom (DC) liegt der tödliche Strom für den Menschen bei etwa 100 mA. Zwar können laut Datenblatt die CR3032 keinen so hohen Strom liefern, so dass sie theoretisch eine sichere Stromversorgung für den Erdschlussprüfer darstellen, in der Praxis hat der Autor jedoch (Spitzen-) Ströme von einigen hundert Milliampere messen können oder vielmehr müssen, die lange genug aufrechterhalten werden, um potenziell gefährlich zu sein (siehe Textbox über die CR2032). Aus Sicherheitsgründen wurde in den Entwurf ein Strombegrenzungswiderstand R1 eingebaut.

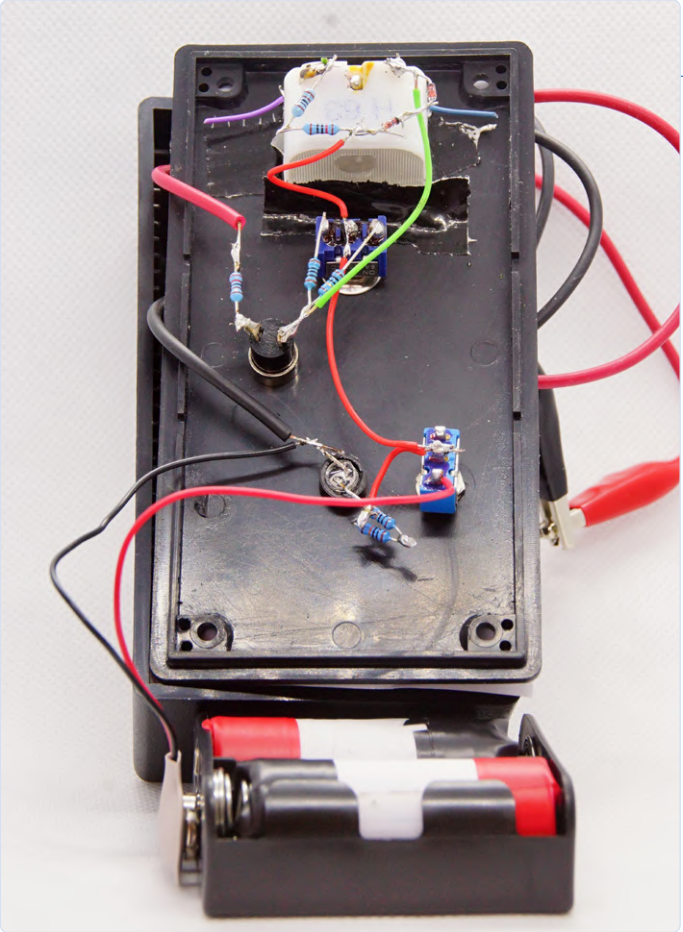


Bild 2. Fliegende Verdrahtung: der Prototyp des Autors.

WIE VIELE CR2032-BATTERIEN GIBT ES?

Die Beliebtheit dieses Batterietyps hat den weniger erfreulichen Nebeneffekt, dass es eine Vielzahl von Herstellern gibt, die CR2032s produzieren. Neben den bekannten Marken gibt es eine unbekannte Anzahl von Herstellern, die Lithiumbatterien mit einer Spannung von 3 V mit den gleichen Abmessungen (20 mm Durchmesser, 3,2 mm dick) herstellen, die die Typennummer bestimmen. Wenn es aber um Daten wie Kapazität, Innenwiderstand und Maximalstrom geht, entpuppt sich der „CR2032-Standard" als weit weniger standardisiert. Und bei den billigeren Typen, die man vorzugsweise kauft, wenn man 32 Batterien braucht, ist es unmöglich herauszufinden, aus welcher Produktion sie stammen, und die Verpackung gibt meist nicht mehr elektrische Daten als „3 V" an. Die vom Autor im Internet gefundene allgemeine Angabe, dass der Kurzschlussstrom dieser Batterien in der Größenordnung von einigen zehn Milliampere liegen würde, erwies sich im Nachhinein in der Praxis als sehr optimistisch, was die Sicherheit betrifft, denn bei Messungen stellte sich heraus, dass die von ihm gekauften CR2032-Batterien einen Kurzschlussstrom von etwa 300 mA besaßen, der nach einigen zehn Sekunden allmählich auf 80 mA abfiel. Aus diesem Grund wäre es eine schlechte Idee, den Widerstand R1 wegzulassen, der den Strom des Erdschlussprüfers begrenzt.

Die Hardware

Für die Messung und Anzeige des Ableitstroms setzte der Autor ein altmodisches Drehspulinstrument ein, gefunden in der Restekiste. Der Vorteil eines solchen analogen Messgeräts ist, dass es schneller reagiert als ein gewöhnliches Digitalmessgerät, so dass auch Stromspitzen beim Einschalten sichtbar sind. Der Widerstand dieses Messgeräts (R_m) beträgt 2,4 kΩ und der Skalenendwert (I_{FS}) beträgt 60 µA. Das Messgerät muss mit Widerständen auf den richtigen Bereich eingestellt werden, es ist aber auch ratsam, es zusätzlich mit einer Diode vor hohen Strömen zu schützen. Eine Diode begrenzt die Spannung über dem Instrument samt Vorwiderstand R3 auf 0,7 V.

Bild 1 zeigt das Schaltbild dieses Erdschluss-Messgeräts. R1 ist der besagte strombegrenzende Sicherheitswiderstand, der, um auf der sicheren Seite zu sein, einen Widerstand von 2,2 kΩ besitzt und den Strom auf etwa 45 mA begrenzt. Wenn dieser Maximalstrom fließt, werden im Widerstand R2 4 W verheizt, so dass entweder ein Leistungswiderstand eingesetzt oder die Prüfdauer immer sehr kurz gehalten werden muss. Mit dem Shunt-Widerstand R2, dem Widerstand R3 und dem Innenwiderstand R_m des Messgeräts können wir (unter der Bedingung, dass R2 viel kleiner ist als R3+R_m) mit dem Ohmschen Gesetz den korrekten Bereich des Messgeräts berechnen:

$$R3 \le \frac{V_{diode}}{I_{FS}} - R_m \qquad R2 = \frac{(R_m + R3) \times I_{FS}}{I_{scale}}$$

wobei I_{scale} der tatsächliche maximale Ausgangsstrom der Schaltung ist. Wir können dann die Widerstandswerte so anpassen, dass sich Standardwiderstände aus der E12-Reihe verwenden lassen. Der Autor hat die folgenden Widerstände für sein Messgerät ermittelt:

- R3 ist 11 kΩ (10 kΩ und 1 kΩ in Reihe). Dies begrenzt die Anzeige auf den Wert 7 bei einem Skalenendwert 8. Die Diode begrenzt die Spannung über R3 und das Instrument auf 0,7 V, was zu einem maximalen Strom von etwa 53 µA führt, während der Skalenendwert ja 60 µA entspricht. Der Autor verwendet daher zwei Dioden in Reihe mit einem Spannungsabfall von 1,4 V, so dass der volle Skalenbereich erreicht wird, während der Strom durch das Instrument innerhalb der erlaubten Grenzen bleibt.
- R2a ist 100 Ω (für 8 mA Vollausschlag)
- R2b ist 10 Ω (für 80 mA Vollausschlag)

Die Schaltung ist so einfach, dass keine Platine erforderlich ist, sondern die wenigen Bauteile fliegend verdrahtet werden können. **Bild 2** zeigt das Innere des vom Autor gebauten Erdschlussprüfers. Unten sehen Sie die beiden mit Klebeband umwickelten Stapel der CR2032-Batterien in einem Standard-Doppel-C-Zellen-Batteriehalter. Die Plus- und Minuspole dieser selbstgebauten 48-V-Batterien sind mit rotem und schwarzem Klebeband markiert.

Messung des Ableitstroms

Die Anwendung dieses Erdschlussprüfers ist einfach. Wenn das Gerät mit S1 eingeschaltet wird, leuchtet die Power-LED D2. Wenn der Taster S2 gedrückt wird, fließt der Messstrom zu den Klemmen, wobei die Messleitungen vorzugsweise mit (isolierten) Krokodilklemmen an die Leitungen des zu prüfenden Stromkreises angeschlossen werden.

Der Strom kann am Messgerät abgelesen werden. Auf diese Weise können wir alle Kabel in den Verteilerkästen und -dosen auf Ableitströme prüfen.
Bei der Prüfspannung aus den Batterien von 96 V ist der gemessene Strom natürlich geringer als der Strom, der bei 230-V-Netzspannung tatsächlich durch den Erdschluss fließen würde. Das heißt, wenn wir davon ausgehen, dass sich der Widerstand der Ableitung zwischen 96 V und 230 V nur geringfügig ändert, fließt bei einer höheren Spannung ein proportional höherer Strom. Zusätzlich reduziert der Innenwiderstand der Batterie, des Shunts und des Sicherheitswiderstands R_{int} den Strom. Der Innenwiderstand des Geräts lässt sich leicht durch Messung des Kurzschlussstroms des Erdschlussprüfgeräts mit einem Multimeter ermitteln. Beim Autor sind dies - mit einem 10-Ω-Shunt (R2b) – 38 mA, was einem Innenwiderstand von 2526 Ω entspricht. Aus dem gemessenen Strom I_m kann dann der tatsächliche Ableitstrom I_{actual} bei Netzspannung berechnet werden:

$$I_{actual} = \frac{V_{mains}}{\frac{V_{batt}}{I_m} - R_{int}}$$

In einer Tabelle eingetragen ergibt sich:

I_m [mA]	I_{actual} [mA]
0	0
1	2
2	5
3	8
4	11
5	14
6	17
7	20
8	24
10	31
20	100
30	>200

Und was war das Problem bei der Elektroinstallation im Wintergarten meiner Eltern? Ein Kabel zu einer Außenlampe hatte einen Isolationsfehler. Der Erdschlussprüfer zeigte einen Ableitstrom von 4 mA an, was in der Praxis bei eingeschalteter Netzspannung etwa 11 mA entsprechen würde. Wahrscheinlich wurde die Leckage bei Regenwetter schlimmer. Die einfachste Möglichkeit, das Problem zu lösen, war der Austausch des Kabels.

200576-02

WEBLINK

[1] **Fehlerstrom-Schutzschalter:** https://de.wikipedia.org/wiki/Fehlerstrom-Schutzschalter

Ein Beitrag von
Entwurf und Text: **Elbert Jan van Veldhuizen**
Redaktion: **Luc Lemmens**
Illustrationen: **Elbert Jan van Veldhuizen, Patrick Wielders**
Übersetzung: **Rolf Gerstendorf**
Layout: **Giel Dols**

Haben Sie Fragen oder Kommentare?
Haben Sie technische Fragen oder Kommentare zu diesem Artikel? Schicken Sie eine E-Mail an den Redakteur unter luc.lemmens@elektor.com oder kontaktieren Sie Elektor unter editor@elektor.com.

PASSENDE PRODUKTE

PeakTech 2710 Digital RCD Tester (SKU 19318)
www.elektor.de/peaktech-2710-digital-rcd-tester

PeakTech 2715 Digital Loop / PSC Tester (SKU 19078)
www.elektor.de/peaktech-2715-digital-loop-psc-tester

VORSICHT HOCHSPANNUNG!

Arbeiten an Stromkreisen oder Anlagen, die (potentiell) an die Netzspannung angeschlossen sind, bergen immer ein Sicherheitsrisiko. Elektor und der Entwickler dieses Projekts übernehmen keine Haftung für Schäden (in welcher Form auch immer), die durch die Verwendung dieses Fehlerstrom-Schutzschalters entstehen. Wie bei allen Projekten, die mit Netzstrom zu tun haben, geben wir den folgenden aufrichtigen Rat: Wenn Sie nicht wissen, was Sie tun, tun Sie es nicht!

ARMUT und Elektronik

1. Ziel für nachhaltige Entwicklung

Von **Priscilla Haring-Kuipers** (Niederlande)

Elektronik berührt die Armut in vielerlei Hinsicht. Technologien können eine Ressource sein, um Menschen aus der Armut zu helfen, oder sie können die Ursache dafür sein.

„Die Armut in all ihren Formen und überall zu beenden" ist das erste der von den Vereinten Nationen in der Agenda 2030 definierten Ziele für nachhaltige Entwicklung (Sustainable Development Goals, SDG). Die Elektronik spielt bei der Armutsbekämpfung oft eine unterstützende Rolle, indem sie Kommunikationsmittel, Inhalte oder Bemessungen in verschiedenen Projekten wie Mikrofinanzierung bereitstellt, und zur Verbesserung von Anbaumethoden und medizinischer Grundversorgung beiträgt. [1]

Die Familie Chowdhury lebt in Indien von 30 Dollar im Monat. Ihr wichtigster Haushaltsgegenstand ist dieses Mobiltelefon. (Quelle: Zoriah Miller, Dollar Street 2015 [2], CC BY 4.0)

Ziele der Armutsbekämpfung

Das erste Teilziel ist die Beseitigung der extremen Armut, definiert als der Anteil der Menschen, die mit weniger als 1,25 Dollar pro Tag auskommen müssen. Wenn Sie wissen wollen, wie das aussieht, gehen Sie zur Dollar Street [2] und schauen Sie sich Familien mit einem Einkommen von bis zu 38 Dollar pro Monat an. Traurigerweise ist die extreme Armut aufgrund von Covid-19 zum ersten Mal in unserer Generation wieder angestiegen. Die Auswirkungen der Pandemie haben im Jahr 2020 rund 120 Millionen Menschen wieder in diese unterste Kategorie befördert. Weitere SDG-Teilziele sind die Halbierung des Anteils der Menschen, die unterhalb der nationalen Armutsgrenze leben, der Zugang armer Menschen zu grundlegenden Diensten, das Recht auf Grundeigentum und die Verringerung der Gefährdung durch klimabedingte Wetterextreme.

Tiefergehende Probleme

Die schlimmste Verflechtung zwischen unserer Elektronik und der Armut besteht im Abbau von Rohstoffen. Der Kobaltabbau im Kongo erfolgt meist unter sehr harten Bedingungen, wobei den Arbeitern - oft Kindern – nur ein Lohn gezahlt wird, der sie weiterhin in extremer Armut leben lässt. Beim Gold- und Kupferabbau auf der ganzen Welt

herrschen „ungesunde Arbeitsbedingungen“ bei sehr niedrigen Löhnen. Der Rohstoffabbau boomt, aber dieser wirtschaftliche Vorteil kommt nicht bei den Händen, die die Arbeit machen, an. Mehrere große Hersteller (Samsung, BASF, Tesla und andere) haben Initiativen zur Förderung des ethischen Tagebaus gestartet, aber diese Projekte scheinen sich bestenfalls im Stadium der guten Vorsätze zu befinden.

Die Familie Jacques lebt in Haiti von 39 Dollar im Monat. Dies sind die Gewürze, die sie zum Kochen verwenden. (Quelle: Zoriah Miller, Dollar Street 2015 [2], CC BY 4.0)

Wir entscheiden

Viele der Entscheidungen, die in Ländern mit hohem Einkommen getroffen werden, bestimmen die wirtschaftlichen Möglichkeiten der Menschen in Ländern mit niedrigem Einkommen. Welche Art von Regulierungen legen wir für die in der Elektronik verwendeten Materialien fest? Lassen unsere Designs Reparaturen zu? Wie verarbeiten wir unseren Elektroschrott? Die Menschen in Ländern mit niedrigem Einkommen können die von uns weggeworfene Elektronik nutzen, indem sie sie reparieren und wiederaufbereitet in ihre eigene Wirtschaft integrieren und so einen Mehrwert für sich selbst schaffen. Ansonsten müssen sie ein sehr ungesundes und unterbezahltes Leben führen, indem sie unsere Elektronik als Abfall sortieren. Unsere Designentscheidungen bestimmen also ihre Möglichkeiten.

Technik als Retterin

Manche glauben, dass die Technologie den Weg aus der Armut weist: „Die moderne Technik ist eine der wirksamsten Lösungen für die Armut [3].“ Die mobile Revolution hat es Menschen, die in Armut leben, ermöglicht, mobile Bankgeschäfte zu tätigen und lebenswichtige Gesundheitsinformationen zu erhalten. Landwirte können bei den Erntepreisen kooperieren und Wettervorhersagen erhalten. Mobiltelefone fungieren auch als Warnsysteme im Falle von Naturkatastrophen, was für Menschen in abgelegenen Gebieten besonders wichtig ist. Pumpen und Filteranlagen mit Solarenergie sorgen für sauberes Wasser und zuverlässige Stromversorgung. Diese und andere Technologien, die in der Bildung und der politischen Zusammenarbeit eingesetzt werden, können die Menschen dabei unterstützen, ihre Ressourcen und Möglichkeiten zu erweitern und ihren Weg aus der extremen Armut zu finden.

Die mobile Revolution ermöglicht es Menschen, die in Armut leben, auf mobiles Banking zuzugreifen und wichtige Gesundheitsinformationen zu erhalten.

210561-02

WEBLINKS

[1] UN Department of Economic and Social Affairs, „SDG Good Practices - A Compilation of Success Stories and Lessons Learned in SDG Implementation“, 1. Auflage, 2020: https://sdgs.un.org/publications/sdg-good-practices-2020
[2] Dollar Street: https://www.gapminder.org/dollar-street
[3] The Borgen Project „10 Technological Solutions to Poverty“, 2014: https://borgenproject.org/10-technological-solutions-poverty

elektorstore

www.elektor.de

Der Elektor Store

Nie teuer, immer überraschend!

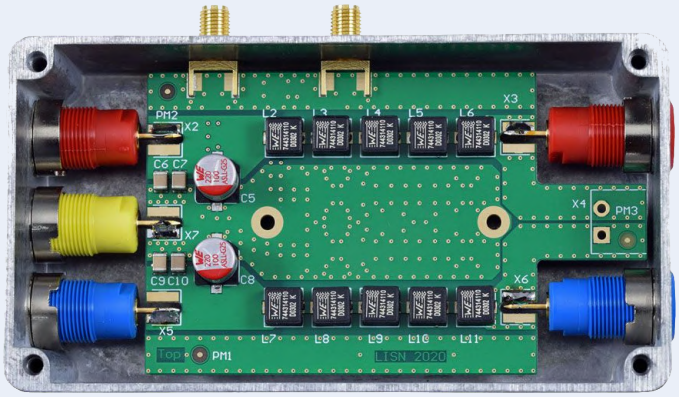
Der Elektor Store hat sich vom Community-Store für Elektor-eigene Produkte wie Bücher, Zeitschriften, Bausätze und Module zu einem umfassenden Webshop entwickelt, der einen großen Wert auf überraschende Elektronik legt. Wir bieten die Produkte an, von denen wir selbst begeistert sind oder die wir einfach ausprobieren wollen. Wenn Sie einen Produktvorschlag haben, sind wir hier erreichbar (sale@elektor.com). Unsere Bedingungen:
Nie teuer, immer überraschend!

Creality HALOT-SKY Resin 3D-Drucker

Preis: 849,00 €
Mitgliederpreis: 764,10 €

www.elektor.de/19745

Elektor Dual DC LISN 150 kHz – 200 MHz



Preis: 109,95 €
Mitgliederpreis: 98,96 €

www.elektor.de/19869

Cytron Maker Pi RP2040 – Robotik mit Raspberry Pi RP2040

Preis: 16,95 €

Mitgliederpreis: 15,26 €

www.elektor.de/19926

Pimoroni Raspberry Pi Pico Wireless Pack

Preis: 17,95 €

Mitgliederpreis: 16,16 €

www.elektor.de/19916

MonkMakes Luftqualitäts-Kit für Raspberry Pi

Preis: 29,95 €

Mitgliederpreis: 26,96 €

www.elektor.de/19913

WIZnet W5100S-EVB-Pico RP2040-basiertes Evaluierungsboard

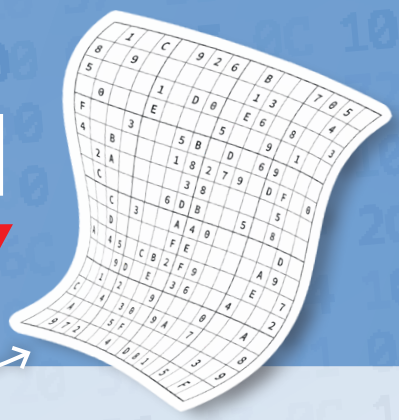
Preis: 14,95 €

Mitgliederpreis: 13,46 €

www.elektor.de/19971

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 15. Februar 2022.

DIE GEWINNER DES HEXADOKUS AUS DER AUSGABE NOVEMBER/DEZEMBER STEHEN FEST!

Die richtige Lösung ist: **53974**

Aus allen Einsendungen mit der richtigen Lösung haben wir die fünf Gewinner eines Elektor-Wertgutscheins über je 50 € gezogen.

Die Namen der Gewinner werden unter www.elektormagazine.de/hexadoku bekannt gegeben.

Herzlichen Glückwunsch!

	4	7		3		B			5		0		6	8	
	1	D				0	2	B	7				A	4	
B	0		5				6	D				7		E	2
C	A		6		7	8	E	9	1	2		F		D	0
				1	8					5	9				
			A		3	7			C	B		8			
4		8	B				D	F				A	2		C
6	D	E											7	1	9
			3		A	F	9	C	4	0		D			
A						E			2						4
			2	8							A	C			
					5	6			9	D					
5	3	2	D	9	B	4			6	7	C	1	8	F	E
	B	A		0		5			3		E		D	C	
0			9	E							5	2			A
7	E	6		F	1	C			D	A	2		9	5	B

4	9	D	C	3	E	5	F	6	A	8	0	B	2	7	1
E	5	6	3	2	B	0	7	1	9	C	D	F	4	8	A
B	F	A	8	C	1	4	6	E	2	7	3	5	D	0	9
0	1	2	7	8	D	9	A	B	5	F	4	6	3	C	E
C	B	E	9	6	7	D	5	A	8	3	F	2	1	4	0
F	A	0	1	E	8	B	4	2	D	9	7	3	5	6	C
6	7	3	4	9	F	1	2	C	E	0	5	D	8	A	B
D	2	8	5	A	3	C	0	4	B	1	6	E	F	9	7
1	E	B	2	F	9	7	8	D	6	A	C	4	0	3	5
3	6	5	D	4	C	E	1	F	0	B	9	A	7	2	8
7	4	9	A	5	0	6	D	8	3	2	1	C	E	B	F
8	C	F	0	B	2	A	3	5	7	4	E	9	6	1	D
2	8	1	6	7	4	F	E	9	C	5	B	0	A	D	3
5	3	4	B	D	6	8	C	0	1	E	A	7	9	F	2
9	D	7	E	0	A	2	B	3	F	6	8	1	C	5	4
A	0	C	F	1	5	3	9	7	4	D	2	8	B	E	6

Der Rechtsweg ist ausgeschlossen. Mitarbeiter der in der Unternehmensgruppe Elektor International Media B.V. zusammengeschlossenen Verlage und deren Angehörige sind von der Teilnahme ausgeschlossen.

elektor e-zine

Your dose of electronics

Jede Woche, in der Sie den Elektor e-zine Newsletter nicht abonnieren, ist eine Woche mit großartigen Artikeln und Projekten zum Thema Elektronik, die Sie verpassen!

Also, worauf warten Sie noch? Melden Sie sich heute für unseren Elektor e-zine Newsletter unter www.elektor.de/ezine an und erhalten Sie zusätzlich ein kostenloses Raspberry Pi Projektbuch!

Was können Sie erwarten?

Redaktioneller Elektor-Newsletter
Jeden Freitag erhalten Sie die wichtigsten Artikel und Projekte der Woche. Wir zeigen MCU-basierte Projekte, IoT, Programmierung, KI und mehr!

Elektor-Newsletter mit exklusiven Angeboten
Verpassen Sie nicht unsere Shop-Angebote, jeden Dienstag und Donnerstag haben wir eine besondere Aktion für Sie.

Mailing von externen Partnern
Sie wollen über die laufenden Aktivitäten in der Branche informiert bleiben? Dann gibt Ihnen diese E-Mail die besten Einblicke. Unregelmäßig, aber immer mittwochs.

elektor
design > share > sell

Treten Sie jetzt der Elektor Community bei!

Jetzt elektor GOLD membership 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 elektor GREEN membership 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

elektor design › share › sell